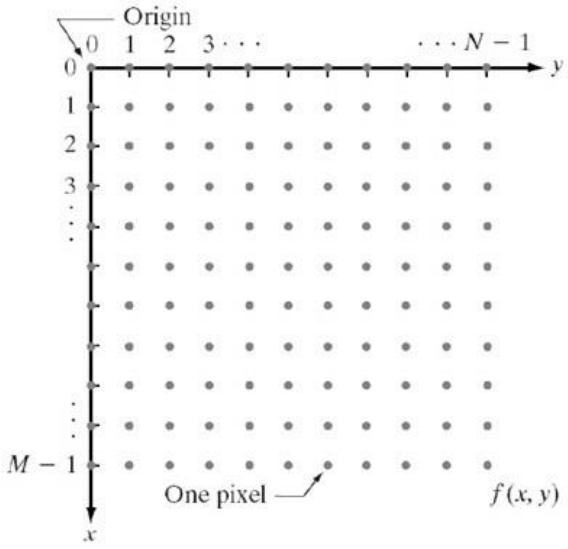
BAB 2
LANDASAN TEORI

2.1 Citra Digital

Citra adalah gambar pada bidang dua dimensi, maka sebuah citra merupakan dimensi special atau bidang yang berisi informasi warna yang tidak bergantung waktu [8]. Citra digital merupakan suatu gambar yang tersusun dari piksel, dimana tiap piksel merepresentasikan warna (tingkat keabuan untuk gambar hitam putih) pada suatu titik di gambar [2]. Secara umum, pengolahan citra digital menunjukkan pemroresan gambar 2 dimensi menggunakan computer. Citra digital merupakan sebuah larik (*array*) yang berisi nilai – nilai real maupun komplek yang direpresentasikan dengan deretan bit tertentu.

Sebuah citra dapat didefinisikan sebagai fungsi *f(x,y)* berukuran M baris dan N kolom, dengan *x* dan *y* adalah koordinat spasial, dan amplitudo *f* di titik koordinat *(x,y)* dinamakan intensitas atau tingkat keabuan dari citra pada titik tersebut. Apabila nilai *x,y* dan nilai amplitudo *f* secara keseluruhan berhingga *(finite)* dan bernilai diskrit maka dapat dikatakan bahwa citra tersebut adalah citra digital.



Gambar 2.1 Koordinat Citra Digital

2.1.1 Elemen Citra Digital

Citra digital mengandung sejumlah elemen-elemen dasar. Elemen-elemen dasar tersebut dimanipulasi dalam pengolahan citra dan dieksploitasi lebih lanjut dalam computer vision. Elemen-elemen dasar yang penting diantaranya adalah :

1. Kecerahan (*brightness*)

   Kecerahan adalah kata lain untuk intensitas cahaya. Kecerahan pada sebuah titik (*piksel*) di dalam citra bukanlah intensitas yang real, tetapi sebenarnya adalah intensitas rata – rata dari suatu area yang melingkupinya. Sistem visual manusia mampu menyesuaikan dirinya dengan tingkat kecerahan (*brightness level*) mulai dari yang paling rendah sampai yang paling tinggi dengan jangkauan sebesar 10^{10}.

2. Kontras (*contrast*)

   Kontras menyatakan sebarang terang (*lightness*) dan gelap (*darkness*) di dalam sebuah gambar. Citra dengan kontras rendah dicirikan oleh sebagian besar komposisi citranya adalah terang atau sebagian besar gelap. Pada citra dengan kontras yang baik, komposisi gelap dan terang tersebar secara merata.

3. Kontur (*contour*)

   Kontur adalah keadaan yang ditimbulkan oleh perubahan intensitas pada piksel – piksel yang bertetangga. Karena adanya perubahan intensitas inilah mata kita mampu mendeteksi tepi – tepi (*edge*) objek di dalam citra.

4. Warna (*color*)

   Warna adalah persepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang cahaya yang dipantulkan oleh objek. Setiap warna mempunyai panjang gelombang (λ) yang berbeda. Warna merah mempunyai panjang gelombang paling tinggi, sedangkan warna ungu mempunyai panjang gelombang paling rendah. Warna – warna yang diterima oleh mata (sistem visual manusia) merupakan hasil kombinasi cahaya dengan panjang gelombang berbeda.

5. Bentuk (*shape*)

   Shape adalah properti intrinsik dari objek tiga dimensi, dengan pengertian bahwa shape merupakan properti intrinsik utama untuk sistem visual manusia. Manusia lebih sering mengasosiasikan objek dengan bentuknya daripada elemen lainnya. Pada umumnya, citra yang dibentuk oleh mata merupakan citra dwimatra (2 dimensi), sedangkan objek yang dilihat umumnya berbentuk trimatra (3 dimensi).

6. Tekstur (*texture*)

   Tekstur dicirikan sebagai distribusi spasial dari derajat keabuan di dalam sekumpulan piksel – piksel yang bertetangga. Tekstur tidak dapat didefinisikan untuk sebuah piksel. Sistem visual manusia pada hakikatnya tidak menerima informasi citra secara independen pada setiap piksel, melainkan suatu citra dianggap sebagai suatu kesatuan.

2.1.2 Resolusi Citra

Resolusi citra merupakan tingkat detail suatu citra. Semakin tinggi resolusi citra maka akan semakin tinggi pula tingkat detail dari citra tersebut. Satuan dalam pengukuran resolusi citra dapat berupa ukuran fisik (jumlah garis per tinggi citra). Resolusi sebuah citra dapat diukur dengan berbagai cara.

1. Resolusi Piksel

   Resolusi piksel merupakan perhitungan jumlah piksel dalam sebuah citra digital. Sebuah citra dengan tinggi N dan lebar M piksel berarti memiliki resolusi sebesar M x N. Resolusi piksel akan memberikan dua buah angka integer yang secara berurutan akan mewakili jumlah piksel lebar dan jumlah piksel tinggi dari citra tersebut.

2. Resolusi Spasial

   Resolusi spasial menunjukkan seberapa dekat jarak setiap garis pada citra. Jarak tersbut tergantung dari sistem yang menciptakan citra tersebut. Resolusi spasial menghasilkan jumlah piksel per satuan panjang.

3. Resolusi Spektral

   Sebuah citra digital membedakan intensitas ke dalam beberapa spectrum. Citra multi spectrum akan memberikan spectrum atau panjang gelombang yang lebih baik yang akan digunakan untuk menampilkan warna.

4. Resolusi Temporal

   Resolusi temporal berkaitan dengan video. Suatu video merupakan kumpulan frame statis yang berupa citra yang berurutan dan ditampilkan secara cepat. Resolusi temporal memberikan jumlah frame yang dapat ditampilkan setiap satuan *frame per second* (fps).

5. Resolusi Radiometrik

   Resolusi ini memberikan nilai atau tingkat kehalusan citra yang dapat ditampilkan dan biasanya ditampilkan dalam satuan bit, contohnya citra 8 bit dan citra 256 bit. Semakin tinggi resolusi radiometrik ini maka semakin baik perbedaan intensitas yang ditampilkan.

2.1.3 Jenis Citra

Nilai suatu piksel memiliki nilai dalam rentang tertentu, dari nilai minimum sampai nilai maksimum. Jangkauan yang digunakan berbeda-beda tergantung dari jenis warnanya. Namun secara umum jangkauannya adalah 0 – 255. Citra dengan penggambaran seperti ini digolongkan ke dalam citra integer [5].

1. Citra Biner

   Citra biner adalah citra yang hanya memiliki dua kemungkinan nilai piksel yaitu hitam dan putih. Citra biner juga disebut sebagai citra B & W (black and white) atau citra monokrom. Hanya dibutuhkan 1 bit untuk mewakili nilai setiap piksel dari citra biner.

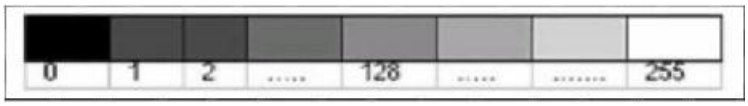
   Citra biner sering kali muncul sebagai hasil dari proses pengolahan seperti segmentasi, pengembangan, morfologi, ataupun dithering.



Gambar 2.2 Contoh Citra Biner

2. Citra *Grayscale*

Grayscale adalah warna-warna piksel yang berada dalam rentang gradasi warna hitam dan putih. Intensitas *grayscale* biasanya disimpan sebagai data citra 8 bit atau 256 intensitas warna *gray* dari nilai 0 (hitam) dan 255 (putih). Citra *grayscale* merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap pikselnya, dengan kata lain nilai bagian *RED* = *GREEN* = *BLUE*. Nilai tersebut digunakan untuk menunjukan tingkat intensitas. Warna yang dimiliki adalah warna dari hitam, keabuan, dan putih. Tingkatan keabuan disini merupakan warna abu dengan berbagai tingkatan dari hitam hingga mendekati putih [13]. Model warna ini menyimpan informasi warna sebesar 8 bit. Jadi warna yang bisa ditampilkan pada mode warna *greyscale* berjumlah sampai 256. Skala perhitungan numeriknya bisa dengan pendekatan nilai *brightness* yang berskala 0 sampai 255 seperti pada Gambar 2-3.



Gambar 2.3 Komposisi Warna Grayscale

Untuk mengkonversikan citra yang memiliki warna RGB ke derajat keabuan bisa menggunakan rumus 2.1 :

$G = 0.3.R + 0.59.G + 0.11.B$ .....(2.1)

Dimana :

G = Citra Greyscale

R = Red

G = Green

B = Blue

R = nilai red * 15 / 255

G = nilai green * 15 / 255

B = nilai blue * 15 / 255

Contoh dari Hasil dari proses *grayscaling* ditunjukkan pada Gambar 2.3 dibawah ini :



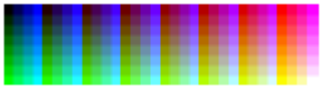
Gambar 2.4 Contoh Citra *Grayscale*

3. Citra Warna (8 bit)

Setipa piksel dari citra warna (8 bit) hanya diwakili oleh 8 bit dengan jumlah warna maksimum yang dapat digunakan adalah 256 warna. Ada dua jenis citra warna 8 bit. Pertama, citra warna 8 bit dengan menggunakan palet warna 256 dengan setiap paletnya memiliki pemetaan nilai (colormap) RGB tertentu. Model ini lebih sering digunakan. Kedua, setiap piksel memiliki format 8 bit sebagai berikut.

Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0
R	R	R	G	B	G	B	

Bentuk kedua dinamakan 8 bit true color. Berikut ini adalah warna-warnanya.



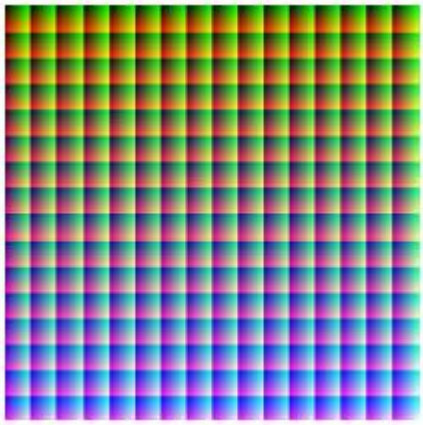
Gambar 2.5 Komposisi warna 8 bit true color

4. Citra Warna (16 bit)

   Citra warna 16 bit (biasanya disebut sebagai citra highcolor dengan setiap pikselnya diwakili dengan 2 byte memory 916 bit). Warna 16 bit memiliki 65.536 warna. Dalam formasi bitnya, nilai merah dan biru mengambil tempat di 5 bit di kanan dan kiri. Komponen hijau memiliki 5 bit ditambah 1 bit ekstra. Pemilihan komponen hijau dengan deret 6 bit dikarenakan penglihatan manusia lebih sensitive terhadap warna hijau.

Bit-15	Bit-14	Bit-13	Bit-12	Bit-11	Bit-10	Bit-9	Bit-8	Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0
R	R	R	R	R	G	G	G	G	G	G	B	B	B	B	B

   Berikut ini adalah deret warna yang dihasilkan dari warna 16 bit.



Gambar 2.6 Komposisi warna 16 bit truecoolor

5. Citra Warna (24 bit)

   Setiap piksel dari citra warna 24 bit diwakili dengan 24 bit sehingga total 16.777.216 variasi warna. Variasi ini sudah lebih dari cukup untuk memvisualisasikan seluruh warna yang dapat dilihat oleh penglihatan manusia. Penglihatan manusia dipercaya hanya dapat membedakan hingga 10 juta warna saja. Setiap poin informasi piksel (RGB) dismpan ke dalam 1 byte data. 8 bit pertama menyimpan nilai biru, kemudian diikuti dengan nilai hijau pada bit kedua dan pada 8 bit terakhir merupakan warna merah.

2.1.4 Format File Citra

Format file citra standar yang digunakan saat ini terdiri dari beberapa Janis. Format-format ini digunakan dalam menyimpan citra dalam sebuah file. Setiap format memiliki karakteristik masing-masing.

1. Bitmap (.bmp)

   Format .bmp adalah format penyimpanan standar tanpa kompresi yang umum dapat digunakan untuk menyimpan citra biner hingga citra warna. Format ini terdiri dari beberapa jenis yang setiap jenisnya ditentukan dengan jumlah nilai bit yang digunakan untuk menyimpan sebuah nilai piksel. Saat ini format BMP kurang diminati karena ukuran berkasnya relatuf lebih besar daripada berkas JPG. Namun, format ini memiliki kelebihan yaitu kualitas yang bagus karena citranya tidak dimampatkan sehingga tidak ada informasi yang hilang.

2. Tagged Image Format (.tif, .tiff)

   Format .tif merupakan format penyimpanan citra yang dapat digunakan untuk menyimpan citra bitmap hingga citra dengan warna palet terkompresi. Format ini dapat digunakan untuk menyimpan citra yang tidak terkompresi dan juga citra terkompresi.

3. Portable Network Graphics (.png)

   Format .png adalah format penyimpanan citra terkompresi. Format ini dapat digunakan pada citra *grayscale*, citra dengan paler warna, dan juga citra fullcolor. Format .png juga mampu menyimpan informasi hingga kanal alpha dengan penyimpanan sebesar 1 hingga 16 bit per kanal.

4. JPEG (.jpg)

   Format .jpg adalah format yang sangat umum digunakan saat ini khususnya untuk transmisi citra. Format ini digunakan untuk menyimpan citra hasil kompresi dengan metode JPEG.

5. MPEG (.mpg)

   Format ini digunakan di dunia internet dan dieperuntukkan sebagai format penyimpanan citra bergerak (video). Format ini mendukung video dengan kompresi ber-rugi.

6. Graphics Interchange Format (.gif)

   Format ini dapat digunakan pada citra warna dengan palet 8 bit. Penggunaan umumnya pada aplikasi web. Kualitas yang rendah menyebabkan format ini tidak terlalu popular di kalangan peneliti pengolahan citra digital.

7. Portable Image File Format

   Format ini memiliki beberapa bagian diantaranya adalah *portable bitmap, portable graymap, portable pixmap,* dan *portable network map* dengan format berturut-turut adalah .pbm, .pgm, .ppm dan .pnm. Format ini baik digunakan untuk menyimpan dan membaca kembali data citra.

2.2 Pengolahan Citra Digital

Pengolahan citra adalah pemrosesan citra, khususnya dengan menggunakan computer, menjadi citra yang kualitasnya lebih baik. Pengolahan citra bertujuan untuk memperbaiki kualitas citra agar mudah diinterpretasikan oleh manusia atau mesin. Teknik – teknik pengolahan citra mentransformasikan citra menjadi citra lain, masukannya adalah citra dan keluarannya juga citra. Namun, citra keluaran mempunyai kualitas lebih baik daripada citra masukan. Pengolahan citra digital dapat dikelompokkan dalam dua jenis kegiatan :

1. Memperbaiki kualitas suatu gambar, sehingga dapat lebih mudah diinterpretasi oleh mata manusia.
2. Mengolah informasi yang terdapat pada suatu gambar untuk keperluan pengenalan objek secara otomatis.

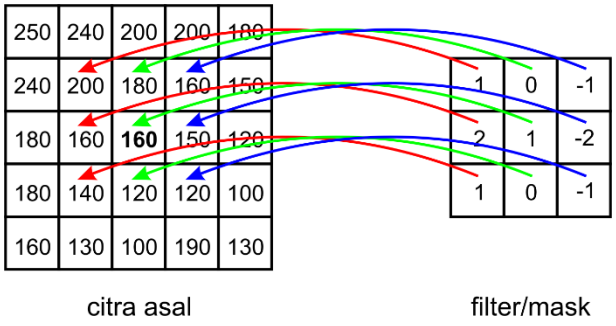
Secara umum tahapan pengolahan citra digital meliputi akusisi citra, peningkatan kualitas citra, segmentasi citra, representasi dan uraian, pengenalan dan interpretasi [13].

2.2.1 Konvolusi

Konvolusi (*convolution*) adalah sebuah proses dimana citra dimanipulasi dengan menggunakan eksternal mask / subwindows untuk menghasilkan citra yang baru. Sedangkan *filtering* tanpa menggunakan eksternal mask tetapi hanya

menggunakan piksel tetangga untuk mendapatkan piksel baru. Konvolusi banyak digunakan dalam berbagai pengolahan citra untuk *smoothing, crispening, edge detection,* serta efek lainnya. Proses konvolusi dapat dijelaskan sebagai berikut. Kernel (*sliding window*) diletakkan pada setiap piksel dari citra input dan menghasilkan piksel baru. Nilai piksel baru dihitung dengan mengalikan setiap nilai piksel tetangga dengan bobot yang berhubungan pada *kernel* dan kenudian menjumlah hasil dari perkalian tersebut.

Untuk pengolahan citra, operasi yang dilakukan adalah operasi diskrit karena nilai koordinat piksel merupakan nilai yang diskrit. Selanjutnya filter atau *mask* yang digunakan dalam pengolahan citra biasanya berukuran terbatas, dalam arti bahwa bobot atau pengaruh dari titik-titik yang cukup jauh sudah tidak signifikan sehingga dapat diabaikan (dianggap nol). Ilustrasi proses konvolusi ditunjukkan pada gambar (2.7) dibawah ini [5].



Gambar 2.7 Contoh perhitungan nilai keabuan titik hasil

2.2.2 Kernel

Kernel adalah matrik yang pada umumnya berukuran kecil dengan elemen-elemennya adalah berupa bilangan. Kernel digunakan pada proses *konvolusi.* Oleh Karena itu, kernel juga disebut dengan *convolution window* (jendela konvolusi). Ukuran kernel dapat berbeda-beda, seperti 2 x 2, 3 x 3, 5 x 5, dan sebagainya. Elemen – elemen kernel yang juga disebut bobot (weight) merupakan bilangan-bilangan yang membentuk pola -pola tertentu. Kernel biasa juga disebut dengan tapis (*filter*), *template, mask,* serta *sliding window*. Dalam konsep morphologi kernel disebut juga dengan *structuring element* [5].

2.2.3 Teknik – Teknik Pengolahan Citra

Secara umum teknik pengolahan citra dibagi menjadi 3 tingkat pengolahan, yaitu sebagai berikut :

1. *Low-Level Processing*, pengolahan ini merupakan operasi dasar dalam pengolahan citra, seperti pengurangan *noise*, perbaikan citra dan restorasi citra.
2. *Mid-Level Processing*, meliputi segmentasi pada citra, deskripsi objek, dan klasifikasi objek secara terpisah.
3. *High-Level Processing*, meliputi analisis citra.

Dari ketiga tingkat pengolahan citra diatas tersebut, dapat dinyatakan suatu gambaran mengenai teknik – teknik pengolahan citra digital dan macam – macamnya antara lain sebagai berikut :

1. *Image Enhancement*, berupa proses perbaiakan citra dengan meningkatkan kualitas itra baik kontras maupun kecerahan.
2. *Image Restoration*, proses memperbaiki model citra, biasanya berhubungan dengan bentuk citra yang sesuai.
3. *Color Image Processing*, suatu proses yang melibatkan citra berwarna, baik berupa image enhancement, image restoration atau yang lainnya.
4. *Wavelet dan Multiresolution Processing*, merupakam suatu proses yang menyatakan citra dalam beberapa resolusi.
5. *Image Compression*, merupakan proses yang digunakan untuk mengubah ukuran data pada citra.
6. *Morpholohical Processing*, proses untuk mendapatkan informasi yang menyatakan deskripsi dari suatu bentuk pada citra.
7. *Segmentataion*, proses untuk membedakan atau memisahkan objek– objek yang ada alam suatu citra, seperti objek dengan latar belakangnya.
8. *Object Recognition*, suatu proses yang dialakukan untuk mengenali objek objek apa saja yang ada dalam suatu citra.

2.3 Metode Ektraksi *Gabor Filter*

Metode *gabor filter* merupakan metode pengenalan objek yang bertujuan untuk memunculkan ciri-ciri tekstur khusus dari citra yang telah dikonvolusi terhadap kernel. Sebagai filter digunakan *gabor filter kernel* 2D yang diperoleh dengan memodulasi gelombang *sinus* 2D pada frekuensi dan orientasi tertentu dengan *Gaussian envelope*. Persamaan dasar fungis *gabor filter* 2D ditunjukkan pada persamaan (2.1), dimana σ adalah standar deviasi dari *Gaussian envelope* pada dimensi *x* dan *y*. Lamda (λ) dan tetha (θ) adalah panjang dan orientasi dari gelombang sinus 2D. Penyebaran dari *Gaussian envelope* didefinisikan dalam bentuk dari gelombang *sinus* λ. Rotasi dari $x - y$ sebesar sudut θ menghasilkan *gabor filter* pada orientasi θ [20].

$$g(x, y; \lambda, \theta, \varphi, \sigma, \gamma) = \exp\left(-\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}\right) \cos\left(2\pi \frac{x'}{\lambda} + \psi\right) \quad \text{.....(2.2)}$$

Dimana :

$x' = x \cos\theta + y \sin\theta$

$y' = -x \sin\theta + y \cos\theta$

σ = standar deviasi *Gaussian envelope*

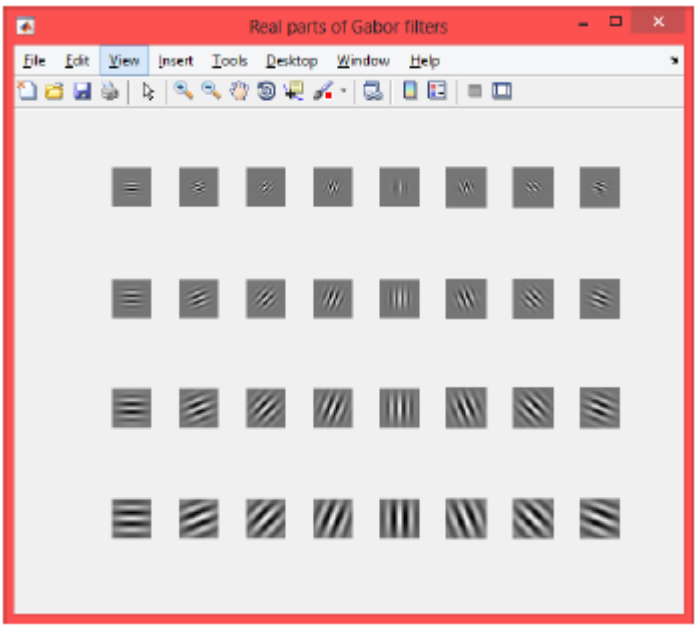
λ = panjang gelombang *sinus* pada 2D dimensi

θ = derajat orientasi gelombang *sinus* pada 2D dimensi

γ = aspek ratio spasial

ψ = *phase offset* (sebagai faktor kosinus pada fungsi gabor)

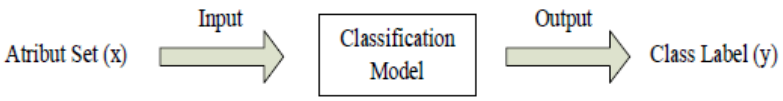
Psi (ψ) memiliki nilai derajat antara -180, -90, 0, 90, 180. Nilai gamma (γ) yaitu 0,5. Sigma (σ) tidak memiliki nilai yang dapat ditetapkan, tetapi dapat diganti dengan nilai bandwidth (b) yaitu 1. Panjang gelombang (λ) dan derajat orientasi (θ) berbeda diterapkan pada satu titik tertentu (x,y), maka didapatkan banyak respon *filter* untuk titik tersebut, missal : digunakan empat panjang gelombang dan delapan orientasi, maka akan dihasilkan 32 respon *filter* untuk tiap titik citra yang dikomvolusikan dengan *filter* tersebut.



Gambar 2.8 Contoh Real Part G*abor Filter*

2.4 Metode Klasifikasi

Klasifikasi adalah proses untuk menemukan model atau fungsi yang menjelasakan atau membedakan konsep atau kelas data, dengan tujuan untuk dapat memperkirakan kelas dari suatu objek yang labelnya tidak diketahui. Dalam mencapai tujuan tersebut, proses klasifikasi membentuk satu model yang mampu membedakan data kedalam kelas – kelas yang berbeda berdasarkan aturan atau fungsi tertentu. Model itu tersendiri bisa berupa aturan "jika-maka", berupa pohon keputusan, atau formula matematis [19].



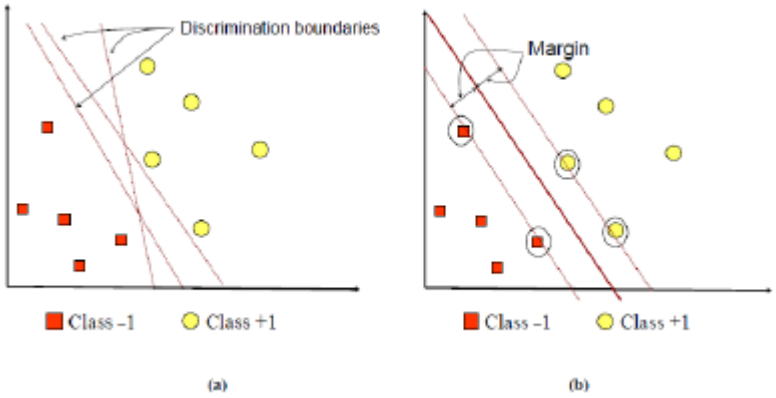
Gambar 2.9 Blok Diagram Model Klasifikasi

2.4.1 Support Vector Machines

Support Vector Machines (SVM) merupakan metode klasifikasi *supervised* karena ketika proses pelatihan, diperlukan target pembelajaran tertentu. SVM juga merupakan algoritma yang bekerja menggunakan pemetaan nonlinear untuk mengubah data pelatihan asli ke dimensi yang lebih tinggi. Dalam hal ini dimensi

baru, akan mencari *hyperplane* untuk memisahkan secara linear dan dengan pemetaan nonlinear yang tepat ke dimensi yang cukup tinggi, data dari dua kelas selalu dapat dipisahkan dengan *hyperplane* tersebut. SVM menemukan *hyperplane* ini menggunakan *support vector* dan *margin* [21].

SVM muncul pertama kali pada tahun 1992 oleh **Vladimir Vapnik** bersama rekannya **Berhand Boser** dan **Isabelle Guyon.** Dasar untuk SVM sudah sejak tahun 1960-an (termasuk karya awal oleh Vapnik dan Alexei Chervonenkis pada teori belajar statistik).



Gambar 2.10 Proses menemukan *hyperplane* oleh metode SVM dengan memisahkan kedua class -1 dan +1

Hyperplane pemisah terbaik antara kedua class dapat ditemukan dengan mengukur *margin hyperplane* tersebut dan mencari titik maksimalnya. *Margin* adalah jarak antara *hyperplane* tersebut dengan *pattern* terdekat dari masing-masing *class. Pattern* yang paling dekat ini disebut sebagai *support vector.* Garis *solid* pada gambar 2.10-b menunjukkan *hyperplane* yang terbaik, yaitu yang terletak tepat pada tengah-tengah kedua *class*, sedangkan titik merah dan kuning yang berada dalam lingkaran hitam adalah hitam adalah *support vector*. Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pembelajaran pada SVM [22].

Konsep SVM dapat dijelaskan secara sederhana sebagai usaha mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua buah *class* pada input *space.* Gambar 2.10-a memperlihatkan beberapa *pattern* yang merupakan anggota

dari dua buah *class* +1 dan -1. *Pattern* yang tergabung pada *class* -1 disimbolkan dengan warna merah (kotak), sedangkan *pattern* +1 disimbolkan dengan warna kuning (lingkaran). Problem klasifikasi dapat diterjemahkan dengan usaha menemukan garis (*hyperplane*) yang memisahkan antara kedua kelompok tersebut. Dalam linear Support Vector Machine permisah merupakan fungsi linear. Data latih dinyatakan oleh (x_i, y_i) dan x_i = { x_1, x_2, ... x_{iq}} merupakan atribut (fitur) set untuk data latih kelas ke-i. Untuk yi € {-1,1} menyatakan label kelas. Pendefinisian persamaan suatu hyperplane pemisah yang dituliskan dengan:

$$w * x_i + b = 0 \quad \text{.....(2.3)}$$

Data xi yang terbagi ke dalam dua kelas, yang termasuk kelas -1 (sampel negatif) didefinisikan sebagai vektor yang memenuhi pertidaksamaan (2.4) berikut.

$$w * x_i + b < 0 \text{ untuk } y_i = -1 \quad \text{.....(2.4)}$$

Sedangkan yang termasuk kelas +1 (sampel positif) memenuhi pertidaksamaan (2.5) berikut.

$$w * x_i + b >= 0 \text{ untuk } y_i = +1 \quad \text{.....(2.5)}$$

Dimana:

x_i = data input

y_i = label yang diberikan

w = nilai dari bidang normal

b = posisi bidang relatif terhadap pusat koordinat

Parameter w dan b adalah parameter yang akan dicari nilainya. Bila label data yi = -1, maka pembatas menjadi persamaan (2.6) berikut:

$$w * x_i + b \leq -1 \quad \text{.....(2.6)}$$

Bila label data yi = +1, maka pembatas menjadi persamaan (2.7) berikut:

$$w * x_i + b \geq +1 \quad \text{.....(2.7)}$$

Margin terbesar dapat dicari dengan cara memaksimalkan jarak antar bidang pembatas kedua kelas dan titik terdekatnya, yaitu 2/|w|. Hal ini dirumuskan sebagai permasalahan *quadratic programming* (QP) *problem* yaitu mencari titik minimal persamaan (2.8) dengan memperhatikan persamaan (2.9) berikut:

$$min\ \tau(w) = 1/2||w||^2 \quad \text{.....(2.8)}$$

$$yi(w * xi + b) - 1 \geq 0, (i=1,..,n) \quad \text{.....(2.9)}$$

Permasalahan ini ini dapat dipecahkan dengan berbagi teknik komputasi. Lebih mudah diselesaikan dengan mengubah persamaan (2.8) ke dalam fungsi *Lagrangian* pada persamaan (2.10), dan menyederhanakannya menjadi persamaan (2.11) berikut:

$$L(w,b,a) = 1/2\|w\|^2 - \sum_{i=1}^{n} a_i(y_i((\mathrm{w}^T\mathrm{x}_i + \mathrm{b}) - 1)) \quad \text{.....(2.10)}$$

$$L(w,b,a) = 1/2\|w\|^2 - \sum_{i=1}^{n} a_i y_i((\mathrm{w}^T\mathrm{x}_i + \mathrm{b}) - 1)) + \sum_{i=1}^{n} a_i \quad \text{.....(2.11)}$$

Dimana a_i adalah *lagrange multiplier* yang bernilai nol atau positif ($a_i \geq 0$). Nilai optimal dari persamaan (2.11) dapat dihitung dengan meminimalkan L terhadap *w*, *b* dan *a*. Dapat dilihat pada persamaan (2.12) sampai (2.14) berikut:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} a_i y_i x_i = 0 \quad \text{.....(2.12)}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} a_i y_i = 0 \quad \text{.....(2.13)}$$

$$\frac{\partial L}{\partial a} = \sum_{i=1}^{n} a_i y_i(\mathrm{w}^T\mathrm{x}_i + \mathrm{b}) - \sum_{i=1}^{n} a_i = 0 \quad \text{.....(2.14)}$$

Maka masalah *Lagrange* untuk klasifikasi dapat dinyatakan pada persamaan (2.15) berikut:

$$Min\ L(w,b,a) = 1/2\|w\|^2 - \sum_{i=1}^{n} a_i y_i(\mathrm{w}^T\mathrm{x}_i + \mathrm{b}) - \sum_{i=1}^{n} a_i \quad \text{.....(2.15)}$$

Dengan memperhatikan persamaan (2.16) dan (2.17) berikut :

$$w - \sum_{i=1}^{n} a_i y_i x_i = 0 \quad \text{.....(2.16)}$$

$$\sum_{i=1}^{n} a_i y_i = 0 \quad \text{.....(2.17)}$$

Model persamaan (2.15) diatas merupakan model primal *Lagrange.* Sedangkan dengan memaksimalkan L terhadap a_i, persamannya menjadi persamaan (2.18) berikut:

$$Max \sum_{i=1}^{n} a_i a_j y_i y_j^T x_i x_j^T \quad \text{.....(2.18)}$$

Dengan memperhatikan persamaan (2.19) berikut:

$$\sum_{i=1}^{n} a_i y_i = 0\,, a_i \geq 0\ (i,j = 1, \dots, n) \quad \text{.....(2.19)}$$

Untuk mencari nilai x_i dan y_i dapat dilakukan ketika sudah didapatkan nilai ekstraksi ciri dari pembobotan dan inisialisasi kelas . Hasil dari pembobotan ekstraksi diubah ke dalam bentuk format data svm, sedangkan data kelas menjadi label data svm.

Untuk mendapatkan nilai a_i, langkah pertama adalah mengubah setiap abstrak menjadi nilai vektor (*support vector*) = $\binom{x}{y}$. Kemudian nilai vektor dari setiap bobot ekstraksi dimasukkan ke persamaan (2.20) kernel *trick phi* berikut.

$$\phi \begin{bmatrix} x \\ y \end{bmatrix} = \begin{cases} \sqrt{x_n^2 + y_n^2} > 2 \; maka \begin{bmatrix} \sqrt{x_n^2 + y_n^2} - x + |x-y| \\ \sqrt{x_n^2 + y_n^2} - y + |x-y| \end{bmatrix} \\ \sqrt{x_n^2 + y_n^2} \leq 2 \; maka \begin{bmatrix} x \\ y \end{bmatrix} \end{cases} \quad \text{.........................(2.20)}$$

Nilai *x* didapatkan dari persamaan (2.21) kernel linear untuk *x* berikut :

$$\sum_{i=1,j=1}^{n} x_i x_j^T , (i,j = 1, \ldots, n) \quad \text{...(2.21)}$$

Nilai *y* didapatkan dari persamaan (2.22) kernel linear untuk *y* berikut :

$$\sum_{i=1,j=1}^{1} y_i y_j^T , (i,j = 1, \ldots, n) \quad \text{...(2.22)}$$

Untuk mendapatkan jarak tegak lurus yang optimal dengan mempertimbangkan vektor positif, maka hasil perhitungan dari substitusi nilai *x* dan nilai *y* ke persamaan (2.20) diberi nilai bias = 1 . Kemudian cari parameter a_i, dengan terlebih dahulu mencari nilai fungsi setiap abstrak menggunakan persamaan (2.23), lalu mencari nilai a_i pada persamaan linear menggunakan persamaan (2.24) dengan memperhatikan i,j = 1,..,n berikut:

$$\sum_{i=1,j=1}^{n} a_i S_i^T S_j \quad \text{...(2.23)}$$

$$\sum_{i=1,j=1}^{n} a_i S_i^T S_j = y_i \quad \text{..(2.24)}$$

Setelah parameter a_i didapatakan, kemudian masukkan ke persamaan (2.25) berikut:

$$W = \sum_{i=1,j=1}^{n} a_i S_i \quad \text{...(2.25)}$$

Hasil yang didapatkan menggunakan persamaan (2.25), selanjutnya digunakan persamaan (2.26) untuk mendapatkan nilai *w* dan *b.*

$$y = wx + b \quad \text{..(2.26)}$$

Sedemikian sehingga didapatkanlah nilai *w* dan *b* atau nilai *hyperplane* untuk mengklasifikasikan kedua kelas. Sebuah fungsi bisa menjasi fungsi kernel jika memenuhi Teorema Mercer, yang menyatakan bahwa matriks kernel yang dihasilkan harus bersifat *positive* semi *definite* [31]. Berikut ini adalah beberapa fungsi kernel yang umum digunakan yaitu :

a. Kernel Linear

$K(x_i,x) = x_i^T x$

b. Polynomial

$K(x_i,x) = (\gamma x_i^T x + r)^p$, $\gamma > 0$

c. Radial Basis Function

$K(x_i,x) = exp(-\gamma | x_i - x |^2)$, $\gamma > 0$

d. Sigmoid Kernel

$K(x_i,x) = tanh(\gamma x_i^T + r)$.

Support vector machines (SVM) merupakan bagian dari metode pembelajaran yang digunakan untuk klasifikasi. SVM memetakan vector input ke sebuah ruang dimensi yang lebih tinggi dimana *hyperplane* pemisah dibangun. Untuk penerapan metode *support vector machines* (SVM) digunakan fungsi-fungsi dalam *LibSVM*. *LibSVM* merupakan *package program* yang digunakan untuk mendapatkan solusi optimal dari *dualitas lagrange* pada SVM. Penggunaan paket program *LibSVM* mengukuti 2 tahapan, yaitu [27] :

2. Melakukan pelatihan terhadap set data untuk mendapatkan sebuah model
3. Menggunakan model yang telah didapatkan untuk memprediksi informasi dalam sebuah set data pelatihan.

Fitur-fitur utama dari LibSVM diantaranya :

- Formulasi SVM yang berbeda
- Efisiensi klasifikasi multi-class
- Cross validation untuk pemilihan model
- Estimasi probabilitas
- Macam-macam kernel (termasuk matriks kernel precomputed)
- Pemilihan model otomatis yang dapat menghasilkan akurasi cross validation.

Class yang terdapat pada *LibSVM* yang digunakan pada penelitian adalah *svm_model, svmTrain.*.

1. *svm_model*, merupakan class method yang terdapat pada library svm yang bertujuan sebagai method penyimpan hasil dari perhitungan data training

yang nantinya akan digunakan dalam klasifikasi untuk menghasilkan prediksi.

2. *svmTrain*, adalah class method pada library svm yang berfungsi untuk melakukan proses perhitungan pada data training dengan menggunakan parameter – parameter yang terdapat pada SVM, seperti : tipe kernel, C, dan tipe svm.

2.5 Mobile Android

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi (Nazruddin, 2012:1) dan menurut Firdan (2011:1) android adalah sistem operasi untuk telepon seluler yang berbasiskan linux. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri sehingga dapat digunakan oleh bermacam peranti penggerak.

Berdasarkan pemaparan pendapat oleh Nazruddin dan Firdan dapat diambil kesimpulan, android adalah suatu sistem operasi untuk *mobile* berbasis linux yang bersifat *open source* sehingga memberikan pengembang untuk melakukan pengembangan sesuai dengan yang diharapkannya sehingga sangat cepat perkembangan sistem operasi ini.

2.5.1 Sejarah Android

Awalnya, Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat piranti lunak untuk ponsel/*smartphone.* Untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, yaitu gabungan dari 34 perusahaan piranti keras, piranti lunak, dan telekomunikasi. Pada saat perilisan perdana Android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan *open source* pada perangkat *mobile*. Di pihak lain, Google merilis kode-kode Android di bawah lisensi Apache, sebuah lisensi perangkat lunak dan *open platform* perangkat seluler. 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) (Nazruddin, 2012: 2).

Saat ini sebagian besar *vendor-vendor smartphone* sudah memproduksi *smartphone* berbasis android. Tidak hanya menjadi sistem operasi di *smartphone*, saat ini Android menjadi pesaing utama dari Apple pada sistem operasi Tablet PC. Pesatnya pertumbuhan Android dikarenakan Android itu sendiri adalah *platform* yang sangat lengkap baik itu dari sistem operasi, aplikasi, alat pengembangan, serta dukungan yang sangat tinggi dari komunitas *open source* di dunia. Berikut ini merupakan perkembangan OS Android sampai saat ini.

Tabel 2.1 Perkembangan Sistem Operasi Android (Nazruddin, 2012:10)

Sistem Operasi	Keterangan
Android Beta	Pertama kali dirilis pada tanggal 5 November 2007, kemudian pada 12 November 2007 Software Development Kit (SDK) dirilis oleh Google.
Android 1.0 (Astro)	Pertama kali dirilis pada 23 September 2008. Sebenarnya Android versi pertama ini akan dinamai dengan nama "Astro" tapi karena alasan hak cipta dan trademark nama"Astro" tidak jadi disematkan pada versi pertama dari OS Android ini. HTC Dream adalah ponsel pertama yang menggunakan OS ini.
Android 1.1 (Bender)	Versi ini dirilis pada 9 Maret 2009, yang muncul dengan pembaruan terhadap estetika pada aplikasi, jam alarm, voice search (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan email. Namun versi Android kedua ini juga mengalami masalah penamaan yang sama dengan versi pertamanya.
Android 1.5 (Cupcake)	Pertama kali dirilis pada 30 April 2009. Cupcake atau dalam terjemahan bebas berarti kue cawan. Mulai versi Android ini penamaan menggunakan nama makan pencuci mulut (dessert) mulai digunakan, karena ini merupakan versi yang ketiga maka penamaan diawali dengan huruf "C" dan jadilah "Cupcake" menjadi nama resmi dari versi OS Android ketiga ini.
Android 1.6 (Donut)	Versi ini dirilis pada September 2009.Donut dikenal juga dengan kue berlubang alias dari versi Android 1.6.
Android 2.0/2.1 (Enclair)	Versi ini dirilis pada 9 Desember 2009. Eclair adalah kue sus yang berbentuk panjang dengan toping coklat diatasnya.
Android 2.2 (Froyo)	Versi ini dirilis pertamakali pada 20 Mei 2010. Froyo (Frozen yoghurt) adalah yoghurt (susu yang dibuat melalui fermentasi bakteri) yang dibekukan mirip seperti

Sistem Operasi	Keterangan
	ice cream. Froyo dipakai sebagai nama alias dari sistem operasi Android versi 2.2.
Android 2.3 (Gingerbread)	Versi ini dirilis pada 6 Desember 2010. Gingerbread adalah kue yang terbuat dari jahe, biasanya berbentuk boneka sering disajikan sebangai teman minum kopi. Gingerbread dipakai sebagai nama alias dari sistem operasi Android versi 2.3
Android 4.0 (Ice Cream Sandwich)	Versi dirilis pada 19 Oktober 2011 .Ice Cream, tentu saja kita tahu karena ini adalah minuman atau tapatnya makanan yang sangat disukai terutama oleh anak kecil. Ice Cream dipakai sebagai nama alias dari Android versi 4.0. Smartphone yang pertama kali mengunakan OS Android ini adalah Samsung Galaxy Nexus.
Android 4.1 (Jelly Bean)	Android Jelly Bean yaang diluncurkan pada acara Google I/O lalu membawa sejumlah keunggulan dan fitur baru.
Android 4.4 (Kitkat)	Dirilis pada 14 oktober 2013 lalu . dengan mengusung konsep sebuah coklat yang sering disajikan untuk para karyawan nya di kantor google. Muncul sebuah ide untuk mengusung nama "KitKat" menjadi nama OS android terbaru setelah 4.1.
Android 5.0/5.1 (Lollipop)	Diresmikan pada 25 Juni 2014 saat Google I / O, dan tersedia secara resmi melalui *over-the-air* (OTA) update pada tanggal 12 November 2014, untuk memilih perangkat yang menjalankan distribusi Android dilayani oleh Google (seperti perangkat Nexus dan Google Play edition).
Android 6.0 (Marshmallow)	Pratayang pengembang Android "M" dirilis pada tanggal 28 Mei 2015, untuk telepon genggam Nexus 5 dan Nexus 6, tablet Nexus 9, dan *set-top box* Nexus Player, di bawah nomor bentukan MPZ44Q. Pratayang pengembang ketiga (MPA44G) dirilis pada tanggal 17 Agustus 2015 untuk Nexus 5, Nexus 6, Nexus 9 dan perangkat Nexus Player, dan diperbarui ke MPA44I yang membawa perbaikan yang berhubungan dengan profil Android *for Work*.

2.5.2 Arsitektur Android

Secara garis besar Arsitektur Android memiliki 5 lapisan, yaitu *Application and Widgets, Application Frameworks, Libraries, Android Run Time,* dan Linux *Kernel* (Nazruddin, 2012: 6).

1. *Application and Widgets*

Application adn Widgets ini adalah lapisan dimana kita berhubungan dengan aplikasi saja, di mana biasanya kita mengunduh aplikasi kemudian menginstalasi dan menjalankan aplikasi tersebut.

2. *Application Frameworks*

   A*pplication Frameworks* ini adalah lapisan di mana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem operasi android, karena pada lapisan inilah aplikasi dapat dirancang dan dibuat.

3. *Libraries*

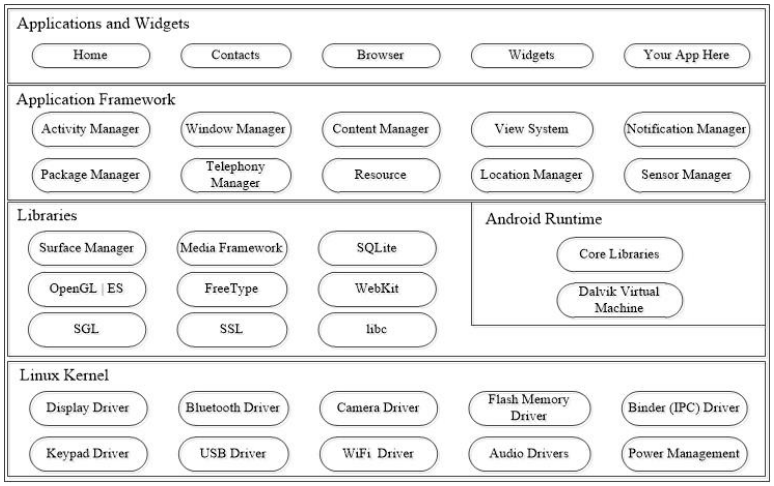
   Libraries ini adalah lapisan di mana fitur-fitur android berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya.

4. *Android Run Time*

   Lapisan inilah yang membuat aplikasi android dapat dijalankan di mana dalam prosesnya menggunakan implementasi linux.

5. Linux *Kernel*

   Linux *kernel* ini adalah lapisan di mana inti dari sistem operasi android itu berada. Berisi file-file sistem yang mengatur sistem *processing*, *memory*, *resource*, *drivers*, dan sistem-sistem operasi android lainnya.



Gambar 2.11 Arsitektur Android (Nazruddin, 2012: 8)

2.5.3 Kelebihan Android

Kelebihan yang dimiliki android adalah sebagai berikut :

1. *User Friendly*

   Kalimat ini sangat melekat pada sistem operasi Windows milikinya Microsoft, ibaratnya kita dengan sangat mudah mengoperasikan komputer hanya dengan belajar beberapa hari bahkan beberapa jam saja, dan ini juga melekat pada Android yang berjalan pada *Smartphone*.

2. *Notifications*

   sangat mudah mendapatkan notifikasi dari smartphone android dengan mengatur beberapa akun E-mail, SMS, Voice Dial, Update dan lain sebagainya.

3. *Tampilan*

   Dari segi tampilan, Android tidak kalah bagusnya dari IOS milik Apple, karena memang dari awal android hampir mengusung teknologi IOS, hanya saja ini versi murahnya.

4. *Open Source*

   Operating system ini memang dibuat opern source oleh penciptanya, karena memang berbasis kernel Linux.

5. *Aplikasi*

   Untuk aplikasi anda akan disajikan jutaan pilihan aplikasi yang menarik dari yang gratis hingga berbayar, dan bias mendownloadnya di Google Play.

2.5.4 Kekurangan Android

Kekurangan dari android sendiri adalah sebagai berikut :

1. *Update System*

   Untuk melakukan update system tidaklah mudah, harus menunggu dari masing-masing vendor untuk merilis update versi yang terbaru.

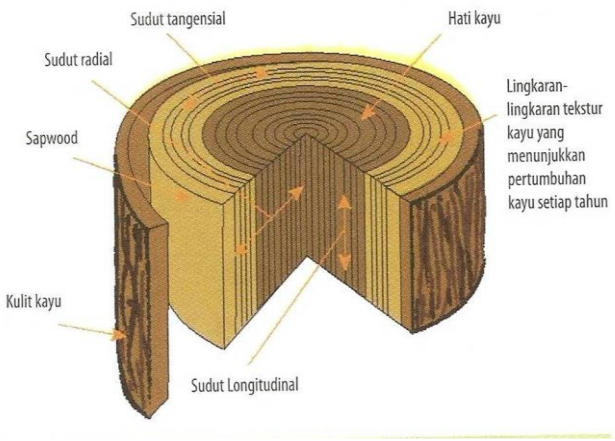
2. *Batterai Cepat Habis*

Ini sering terjadi jika menyalakan paket data dan menggunakan widget serta aplikasi yang berjalan secara berlebihan, untuk mengatasinya pengguna harus mengurangi waktu aktifitas aplikasi pada smartphone.

3. *Lag*

   Kekurangan ini berkaitan dengan spesifikasi dari masing-masing perangkat, namun ada kalanya android ini tidak bersahabat dengan beberapa aplikasi dikarenakan RAM ataupun Processornya yang kurang memadai.

2.6 Pengertian Kayu



Gambar 2.12 Anatomi Kayu

Kayu merupakan hasil hutan dari kekayaan alam, merupakan bahan mentah yang mudah diproses untuk dijadikan barang sesuai kemajuan teknologi. Kayu memiliki beberapa sifat sekaligus, yang tidak dapat ditiru oleh bahan-bahan lain. Pengertian kayu disini ialah sesuatu bahan, yang diperoleh dari hasil pemungutan pohon-pohon di hutan, yang merupakan bagian dari pohon tersebut, setelah diperhitungkan bagian-bagian mana yang lebih banyak dimanfaatkan untuk sesuatu tujuan penggunaan. Baik berbentuk kayu pertukangan, kayu industri maupun kayu bakar.

Selama periode prasejarah dan sesudahnya kayu tidak hanya digunakan untuk bahan bangunan tetapi juga semakin penting sebagai bahan mentah kimia untuk pembuatan arang (digunakan dalam peleburan besi), ter dan getah (digunakan untuk mengawetkan dan melapisi lambung kapal), dan kalium (digunakan dalam pembuatan gelas dan sebagai bahan pemucat kain dan tekstil

kapas). Namun disisi lain kayu merupakan bahan dasar yang sangat modern. Kubah-kubah kayu yang besar dan perabot rumah yang indah membuktikan kagunaan dan keindahannya.

2.6.1 Jenis - Jenis Kayu

Indonesia memiliki sekitar 4.000 jenis pohon, yang berpotensi untuk digunakan sebagai kayu bangunan. Akan tetapi, hingga saat ini hanya sekitar 400 jenis (10%) yang memiliki nilai ekonomi dan lebih sedikit lagi, 260 jenis, yang telah digolongkan sebagai kayu perdagangan [14]. Berikut ini adalah daftar nama-nama kayu menurut nama perdagangannya, sesuai dengan Lampiran Keputusan Menteri Kehutanan Nomor : 163/Kpts-II/2003 tanggal 26 Mei 2003 tentang *Pengelompokan Jenis Kayu Sebagai Dasar Pengenaan Iuran Kehutanan;* dengan beberapa penyesuaian, yaitu Kelompok Jenis Meranti / Kelompok Komersial Satu, Kelompok Jenis Kayu Rimba Campuran / Kelompok Komersial Dua, Kelompok Jenis Kayu Eboni / Kelompok Indah Satu, dan Kelompok Jenis Kayu Indah / Kelompok Indah Dua.

1. Kelompok Jenis Meranti / Kelompok Komersial Satu

Tabel 2.2 Kelompok Kayu Jenis Meranti

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
1.	Agatis	*Agathis* spp.	Damar (Jw.), dama (Slw.), damar bindang (Klm.), damar sigi (Smt.). (Ingg.): *kauri pine*.
2.	Balau	*Shorea* spp. (misalnya *S. materialis* Ridl., *S. maxwelliana* King, *S. scrobiculata* Burck); *Parashorea* spp.	Damar laut (Smt.), semantok (Aceh), amperok, anggelam, selangan batu (Klm.)
3.	Balau merah	*Shorea* spp. (mis. *S. collina* Ridl., *S. guiso* (Blanco) Bl.)	Balau laut, damar laut merah, batu tuyang, putang, lempung abang. Ingg.: *red selangan*.

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
4.	Bangkirai	*Shorea* spp. (mis. *S. kunstleri* King, *S. laevis* Ridley, *S. laevifolia*Endert); *Hopea* spp. (mis. *H. celebica* Burck, *H. semicuneata*Sym.)	Benuas, balau mata kucing, hulo dereh, puguh, jangkang putih, kerangan (Smt.), bubuh (Bk.)
5.	Damar	*Araucaria* spp. (mis. *A. cunninghamii* D. Don, *A. hunsteinii*K.Schum.)	Alloa, ningwik, pien (Pap.). Ingg.: *araucaria*.
6.	Durian	*Durio* spp. (terutama *Durio carinatus* Mast.); *Coelostegia* spp.	Durian burung, lahong, layung, apun, begurah, punggai, durian hantu, enggang
7.	Gia	*Homalium tomentosum* (Roxb.) Benth., *Homalium foetidum*(Roxb.) Benth.	Delingsem (Jw.), kayu batu, melunas, kayu kerbau, momala (Slw.)
8.	Giam[2]	*Cotylelobium* spp. (mis. *C. burckii* Heim, *C. lanceolatum* Craib, *C. melanoxylon* Pierre	Giam durian, resak bukit tembaga; giam padi, resak daun kecil, resak batu; giam tembaga, resak daun lebar; resak gunung
9.	Jelutung	*Dyera* spp.	Pulai nasi, pantung gunung, melabuai
10.	Kapur	*Dryobalanops* spp. (di antaranya *D. oblongifolia* Dyer, *D. sumatrensis* (Gmelin) Kosterm.)	Kamper (kayu), kayu kayatan, empedu, keladan
11.	Kapur petanang	*Dryobalanops oblongifolia* Dyer	Kapur guras (Smt.), kapur paya (Mly.), kelansau (Swk.)
12.	Kenari	*Canarium* spp., *Dacryodes* spp. , *Santiria* spp., *Trioma* spp.	Kerantai, ki tuwak, binjau, asam-asam, kedondong (kedundung), resung, bayung, ranggorai, mertukul

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
13.	Keruing	*Dipterocarpus* spp. (mis. *D. applanatus* V.Sl., *D. baudii* Korth., *D. elongatus* Korth. dll.)	Keruing arong, kekalup; Lagan sanduk, mara keluang; Keruing tempudau; tempurau, merkurang, kawang, apitong
14.	Kulim	*Scorodocarpus borneensis* Becc.	Kayu bawang hutan (Klm.)
15.	Malapari	*Pongamia pinnata* (L.) Pierre	Malapari
16.	Matoa	*Pometia* spp.; mis. *P. pinnata* Forster & Forster, *P. ridleyi* King	Kasai, taun, kungki, hatobu, kayu sapi (Jw.), tawan (Mlku.), ihi mendek (Irian Jaya)
17.	Medang	*Cinnamomum* spp.	Sintuk, sintok lancing, ki teja, ki tuha, ki sereh, selasihan
18.	Meranti kuning	*Shorea* spp. (di antaranya: *S. acuminatissima* Sym., *S. balanocarpoides* Sym., *S. faguetiana* Heim, *S. gibbosa* Brandis,*Shorea scollaris* V.Sl.;	Damar hitam, damar kalepek; Damar hitam katup; Bangkirai guruk, karamuku; Damar buah, mereng-kuyung; Damar tanduk. Ingg.: *yellow seraya*.
19.	Meranti merah	*Shorea* spp. (di antaranya: *S. johorensis* Foxw., *S. lepidota* BI., *S. leprosula* Miq., *S. ovalis* BI., *S. palembanica* Miq., *S. platyclados*V.Sl. ex Foxw., *S. leptoclados* Sym., dll.)	Majau, meranti merkuyung; Meranti ketrahan; Meranti tembaga, kontoi bayor; Meranti kelungkung; Tengkawang majau; Banio, ketir; Seraya merah, campaga, lempong, kumbang, meranti ketuko, cupang. Ingg.: *red seraya*, *red lauan*.
20.	Meranti putih	*Shorea* spp. (di antaranya: *S. assamica* Dyer, *S. bracteolata* Dyer,*S. javanica* K. et. Val., *S. lamellata* Foxw., *S.*	Damar mesegar; Bunyau, damar kedontang; Damar mata kucing, damar kaca,

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
		ochracea Sym., *S. retinodes* V.SI., *S. virescens* Parijs, *S. koordersi* Brandis, dll.)	damar kucing; Damar tunam, damar pakit; Damar kebaong, baong, bayong, baung, belobungo, kontoi tembaga; Balamsarai, damar mansarai; Damar maja, kontoi sabang; Kikir, udang, udang ulang, damar hutan, anggelam tikus, maharam potong, pongin, awan punuk, mehing (Smt., Kal.); Damar lari-lari, lalari, temungku, tambia putih (Slw.), Damar tenang putih, hili, honi (Mlku.). Ingg.:*white meranti*.
21.	Merawan	*Hopea* spp. (mis. *H. dasyrrachis* V.Sl., *H. dyeri* Heim, *H. sangal*Korth., dll.)	Tekam, tekam rayap; Bangkirai tanduk, emang, amang besi; Cengal, merawan telor; Ngerawan, cengal balau
22.	Merbau	*Intsia* spp. (terutama *I. bijuga* O.K., *I. palembanica* Miq.)	Merbau asam, ipi (NT.), kayu besi (Papua); Ipil, anglai, maharan; Tanduk (Mlku.)
23.	Mersawa	*Anisoptera* spp. (mis. *A. laevis* Ridl., *A. marginata* Korth., *A. thurifera* Bl.)	Cengal padi, damar kunyit; Masegar (Smt.), ketimpun (Klm.), mersawa daun besar; tabok, tahan
24.	Nyatoh	*Palaquium* spp., *Payena* spp., *Madhuca* spp.	Suntai, balam, jongkong, hangkang, katingan, mayang batu, bunut, kedang, bakalaung, ketiau, jengkot, kolan

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
25.	Palapi	*Heritiera* (*Tarrietia*) spp.; mis. *H. javanica* (Bl.) Kosterm., *H. simplicifolia* (Mast.) Kosterm., *H. littoralis* Ait., *H. sylvatica* S. Vidal	Mengkulang, teraling; Dungun, talutung, lesi-lesi.
26.	Penjalin	*Celtis* spp.	Rempelas, ki jeungkil, ki endog (Sd.), cengkek (Jw.), pusu (Sumbawa)
27.	Perupuk	*Lophopetalum* spp.; mis. *L. javanicum* (Zoll.) Turcz., *L. multinervium* Ridl., *L. subobovatum* King, *L. wightianum* Arn.	Kerupuk (Smt.), pasana (Klm.), mandalaksa (Jw.), aras
28.	Pinang	*Pentace* spp.	Melunak, ki sigeung, ki sinduk, kelembing
29.	Pulai	*Alstonia* spp. (di antaranya *A. pneumatophora* Back., *A. scholaris*R.Br., *A. spatulata* Bl., *A. macrophylla* Wall., *A. spectabilis* R.Br.)	Kayu gabus, rita, gitoh, bintau, basung, pule, pulai miang. Ingg.: *white cheesewood*, *milkwood*, *milky pine*.
30.	Rasamala	*Altingia excelsa* Noroña	Tulasan (Smt.), mandung (Min.), mala (Jw.)
31.	Resak	*Vatica* spp.; mis. *V. maingayi* Dyer, *V. oblongifolia* Hook.f., *V. rassak* Bl.	Damar along, resak putih

2. Kelompok Jenis Kayu Rimba Campuran / Kelompok Komersial Dua

Tabel 2.3 Kelompok Jenis Kayu Rimba Campuran

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
1.	Bakau	*Rhizophora* spp. dan Bruguiera spp	Tumu, Lenggadai, Jangkar, Tanjang, Putut,

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
			Busing, Mata buaya
2.	Bayur		Balang, Walang, Wadang, Wayu
3.	Benuang	*Octomeles sumatrana* Miq.	Benuang bini (Klm.), winuang (Slw.)
4.	Berumbung	*Adina minutiflora* Val.); *Pertusadina* spp.	Kayu lobang, Barumbung, Kayu gatal
5.	Bintangur	*Calophyllum* spp.; mis. *C. calaba* L., *C. inophyllum* L., *C. papuanum* Lauterb., *C. pulcherrimum* Wall.ex Choisy, *C. soulattri*Burm.f.	Bintangor, penaga; Nyamplung; Sul atri; Bunoh, bintangur bunut
6.	Bipa	*Pterygota* spp.	Kayu wipa
7.	Bowoi	*Serianthes minahassae* Merr. & Perry (Syn. *Albizia minahasae*Koord.)	Rayango, Merang, Terangkuse
8.	Bugis	*Koordersiodendron pinnatum* Merr.	Grepau
9.	Cenge	*Mastixia rostrata* BI.	Cenge, Cingo
10.	Duabanga	*Duabanga moluccana* BI.	Benuang laki, Takir, Aras, Raju mas
11.	Ekaliptus	*Eucalyptus* spp.; mis. *E. alba* Reinw.ex Bl., *E. deglupta* Bl., *E. urophylla* S.T. Blake	Kayu putih; Leda, aren (Mlku.), tampai; Ampupu (Timor) ,
12.	Gelam	*Melaleuca* spp.	Kayu putih
13.	Gempol	*Nauclea* spp.	Wosen, Klepu pasir, Anggrit

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
14.	Gopasa	*Vitex* spp.	Teraut, Laban
15.	Gerunggang/Derum	*Cratoxylum* spp.; mis. *C. arborescens* (Vahl) Bl., *C. cochinchinense* (Lour.) Bl.	Madang baro; Mampat, butun; kemutul, temau; edat
16.	Jabon	*Anthocephalus* spp. (*A. chinensis* (Lamk.) A.Rich ex Walp. dan *A. macrophyllus* (Roxb.) Havil.)	Kelampayan (Mly.), laran (Klm.), semama (Amb.). Ingg.: *cadamba*.
17.	Jambu-jambu	*Syzygium* spp. [3]	Kelat, Ki tembaga, Jambu
18.	Kapas-kapasan	*Exbucklandia populnea* R. Brown	Hapas-hapas, Tapa-tapa, Leman
19.	Kayu kereta	*Swintonia* spp.	Rengas sumpung, Merpauh, Bagel mirah
20.	Kecapi	*Sandoricum* spp.	Papung, Kelam, Sentul
21.	Kedondong Hutan	*Spondias* spp.	Coco, Kacemcem leuweung
22.	Kelumpang	*Sterculia* spp.	Kepuh, Kalupat, Lomes
23.	Kembang semangkok	*Scaphium macropodum* J. B.	Kepayang, merpayang (Smt.)
24.	Kempas	*Koompassia malaccensis* Maing.	Hampas, impas, tualang ayam
25.	Kenanga	*Cananga* sp.	Kananga
26.	Keranji	*Dialium* spp.; mis. *D. indum* L., *D. platysepalum* Baker, *D. procerum* (v.Steen.) Stey	Kayu lilin; Maranji
27.	Ketapang	*Terminalia* spp.	Kalumpit, Klumprit,

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
			Jelawai, Jaha
28.	Ketimunan	*Timonius* spp.	Seranai, Temirit, Kayu reen
29.	Lancat	*Mastixiodendron* spp.	Kundur, Modjiu, Raimagago
30.	Lara	*Metrosideros* spp. dan *Xanthostemon* spp.	Lompopaito, Nani, Langera
31.	Mahang	*Macaranga* spp.	Merkubung, Mara, Benua
32.	Medang	*Litsea firma* Hook f.; *Dehaasia* spp.	Manggah, Huru kacang, Keleban, Wuru, Kunyit
33.	Mempisang	*Mezzetia parviflora* Becc.; *Xylopia* spp.; *Alphonsea* spp.; *K andelia candel* Druce	Mahabai, Hakai rawang, Empunyit, Jangkang, Banitan, Pisang-pisang
34.	Mendarahan	*Myristica* spp., *Knema* spp.	Darah-darah, Tangkalak, Au-au, Ki mokla, Kumpang, Kayu luo, Huru
35.	Menjalin	*Xanthophyllum* spp.	Lilin, Ki endog, Segi landak
36.	Mentibu	*Dactylocladus stenostachys* Oliv.	Jongkong, merebung
37.	Merambung	*Vernonia arborea* Han.	Merambung, sembung
38.	Punak	*Tetramerista glabra* Miq.	Kayu malaka (Smt.), cerega (Klm.)
39.	Puspa	*Schima* spp.; terutama *S. wallichii* Korth.	Seru (Jw.), simartolu (Smt.), madang gatal

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
			(Klm.)
40.	Rengas	*Gluta aptera* (King) Ding Hou	Rengas tembaga, Rangas
41.	Saninten	*Castanopsis argentea* A. DC.	Sarangan (Jw.), ki hiur (Sd.), kalimorot
42.	Sengon	*Paraserianthes falcataria* (L) Nielsen	Jeungjing, Tawa kase, Sika (Maluku)
43.	Sepat	*Berrya cordofolia* Roxb.	Waru gunung, Kalong
44.	Sesendok	*Endospermum* spp.; mis. *E. diadenum* (Miq.) Airy Shaw, *E. moluccanum* (T & B) Kurz, *E. peltatum* Merr.	Sendok-sendok, kayu labuh (Smt.), kayu bulan (Mly.), garung (Klm.); Kayu raja (Mlku.)
45.	Simpur	*Dillenia* spp.; mis. *D. grandifolia* Wall., *D. obovata* Hoogl., *D. pentagyna* Roxb.	Sempur, segel, janti, dongi
46.	Surian	*Toona sureni* Merr.	Suren, kalantas
47.	Tembesu	*Fagraea* spp.; mis. *F. fragrans* Roxb., *F. sororia* J.J. Sm.	Tomasu (Smt.), kulaki (Slw.), malbira, ki tandu
48.	Tempinis	*Sloetia elongata* Kds.	Damuli, Kayu besi
49.	Tepis	*Polyalthia glauca* Boerl.	Banitan, Pemelesian, Kayu tinyang, Kayu bulan, Banet, Kayu kalet
50.	Tenggayun	*Parartocarpus* spp.	Buku ongko, Pejatai, Purut bulu
51.	Terap	*Artocarpus* spp.	Cempedak, Kulur, Tara, Teureup
52.	Terentang	*Campnosperma* spp.; mis. *C. auriculatum* (Bl.)	Tumbus (Smt.), pauh

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
		Hook.f., *C. brevipetiolatum* Volkens, dll.	lebi
53.	Terentang ayam	*Buchanania* spp.	Pauhan, Antumbus, Talantang
54.	T u s a m	*Pinus* spp.	Pinus, Damar batu, Uyam
55.	Utup	*Aromadendron* sp.	U t u p

3. Kelompok Jenis Kayu Eboni / Kelompok Indah Satu

Tabel 2.4 Kelompok Jenis Kayu Eboni

No.	Nama Perdagangan	Nama Ilmiah	Nama-nama Daerah
1.	Eboni bergaris	*Diospyros celebica* Bakh.	Maitong, Kayu lotong, Sora, Amara
2.	Eboni hitam	*Diospyros rumphii* Bakh.	Kayu hitam, Maitem, Kayu waled
3.	E b o n i	*Diospyros* spp.; di antaranya *D. areolata* King et G., *D. cauliflora*BI., *D. ebenum* Koen, *D. ferrea* Bakh., *D. lolin* Bakh., *D. macrophylla* BI.	Baniak, Toli-toli, Kayu arang, Kanara, Gito-gito, Bengkoal, Malam

4. Kelompok Jenis Kayu Indah / Kelompok Indah Dua

Tabel 2.5 Kelompok Jenis Kayu Indah

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
1.	Bongin	*Irvingia malayana* Oliv.	Pauh kijang, Sepah, Kayu batu
2.	Bungur	*Lagerstroemia speciosa* Pers.	Ketangi, wungu (Jw.), tekuyung, benger

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
3.	Cempaka	*Michelia* spp., *Elmerrillia* spp.	Minjaran, Wasian, Manglid, Sitekwok, Kantil (Jw.), Capuka
4.	Cendana	*Santalum album* L.	Kayu kuning, Lemo daru
5.	Dahu	*Dracontomelon* spp.; mis. *D. dao* Merr. & Rolfe, *D. mangiferum* Bl.	Dao, basuong (Smt.), sengkuang (Mly.), koili
6.	Johar	*Senna* spp.[4]	Juar, Trengguli, Sebusuk, Bobondelan
7.	Kuku	*Pericopsis mooniana* Thw.	Kayu laut, Papus, Nani laut
8.	Kupang	*Ormosia* spp.	Kayu ruan, Saga
9.	Lasi	*Adina fagifolia* Ridl.	Adina, Kilaki
10.	Mahoni	*Swietenia* spp.; mis. *S. macrophylla* King, *S. mahagoni* (L.) Jacq.	Mahoni
11.	Melur	*Dacrydium* spp.; *Podocarpus* spp. dan *Phyllocladus* spp. Mis.*Dacrydium junghuhnii* Miq.	Alau, cemantan (Klm.); Jamuju, kayu embun (Slw.), sampinur bunga (Smt.);Sampinur tali; Kayu cina; Ki merah, Sandu
12.	Membacang	*Mangifera* spp.	Ambacang, Asam, Limus piit, Mempelam, Wani, Mangga
13.	Mindi	*Melia* spp.; terutama *M. azedarach* L.	Bawang kungut
14.	Nyirih	*Xylocarpus granatum* J. Konig	Nyireh, Niri
15.	Pasang	*Quercus* spp.	Mempening, Baturua, Kasunu, Triti

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
16.	Perepat darat	*Combretocarpus rotundatus* Dans.	Marapat, Teruntum batu
17.	Raja bunga	*Adenanthera* spp	Saga, Segawe, Klenderi
18.	Rengas	*Gluta* spp.; *Melanorrhoea* spp.	Ingas, Suloh, Rangas, Rengas burung
19.	Ramin	*Gonystylus bancanus* Kurz	Gaharu buaya, Medang keladi, Keladi, Miang
20.	Sawo kecik	*Manilkara* spp.; mis. *M. fascicularis* H.J. Lam & Maas Geest., *M. kauki* (L.) Dub.	Subo, Ki sawo
21.	Salimuli	*Cordia* spp.	Kendal, Klimasada, Purnamasada
22.	Sindur	*Sindora* spp.; mis. *S. bruggemanii* de Wit, *S. coriacea* Maing., *S. wallichii* Graham	Sepetir (Mly.), sasundur (Klm.), mobingo (Slw.)
23.	Sonokembang	*Pterocarpus indicus* Willd.	Angsana, Linggua, Nala, Candana
24.	Sonokeling	*Dalbergia latifolia* Roxb.	Linggota, sono sungu, sonobrits
25.	Sungkai	*Peronema canescens* Jack	Jati seberang, Jati londo
26.	Tanjung	*Mimusops elengi* L.	Sawo manuk (Jw.), karikis (Slw.)
27.	Tapos	*Elateriospermum tapos* BI.	Kelampai, Setan, Kedui, Wayang
28.	Tinjau belukar	*Pteleocarpus lampongus* Bakh.	Lontar kuning
29.	Torem	*Manilkara kanosiensis* H.j. L. et B. M.	Sawai, Torem
30.	Trembesi	*Samanea saman* Merr.	Ki hujan
31.	Ulin	*Eusideroxylon zwageri* T.et B.	Kayu besi, bulian, kokon

No.	Nama Perdagangan	Nama Ilmiah	Nama-Nama Daerah
32.	Weru	*Albizia procera* Benth.	Beru, Ki hiyang, Bengkal

2.6.2 Anatomi Kayu Jati



Gambar 2.13 Kayu Jati

Jati adalah sejenis pohon penghasil kayu bermutu tinggi. Pohon besar, berbatang lurus, dapat tumbuh mencapai tinggi 30-40 m. Berdaun besar, yang luruh di musim kemarau**.** Nama ilmiah jati adalah Tectona grandis L.f. Jati dapat tumbuh di daerah dengan curah hujan 1 500 – 2 000 mm/tahun dan suhu 27 – 36 °C baik di dataran rendah maupun dataran tinggi. Tempat yang paling baik untuk pertumbuhan jati adalah tanah dengan pH 4.5 – 7 dan tidak dibanjiri dengan air. Jati memiliki daun berbentuk elips yang lebar dan dapat mencapai 30 – 60 cm saat dewasa. Kayu jati mengandung semacam minyak dan endapan di dalam sel-sel kayunya, sehingga dapat awet digunakan di tempat terbuka meski tanpa divernis; apalagi bila dipakai di bawah naungan atap.

Jati sejak lama digunakan sebagai bahan baku pembuatan kapal laut, termasuk kapal-kapal VOC yang melayari samudera pada abad ke-17. Juga dalam konstruksi berat seperti jembatan dan bantalan rel. Di dalam rumah, selain dimanfaatkan sebagai bahan baku furniture kayu jati digunakan pula dalam struktur bangunan. Rumah-rumah tradisional Jawa, seperti rumah joglo Jawa Tengah, menggunakan kayu jati di hampir semua bagiannya: tiang-tiang, rangka atap, hingga ke dinding-dinding berukir.

Dalam industri kayu sekarang, jati diolah menjadi venir (*veneer*) untuk melapisi wajah kayu lapis mahal; serta dijadikan keping-keping parket (*parquet*)

penutup lantai. Selain itu juga diekspor ke mancanegara dalam bentuk furniture luar-rumah. Ranting-ranting jati yang tak lagi dapat dimanfaatkan untuk mebel, dimanfaatkan sebagai kayu bakar kelas satu. Kayu jati menghasilkan panas yang tinggi, sehingga dulu digunakan sebagai bahan bakar lokomotif uap.

2.6.3 Keunggulan Kayu Jati

Pada industri meubel, mayoritas penggunaan material kayu adalah dengan menggunakan kayu jati. Penggunaan kayu jati untuk meubel dibandingkan dengan kayu lainnya dimungkinkan karena kayu jati memiliki kandungan minyak alami yang berfungsi sebagai pelindung dari cuaca maupun kondisi yang ekstrim. Dengan demikian, ada beberapa perbedaan kayu jati dengan kayu lainnya khususnya dalam industri meubel, diantaranya [14] :

1. Pori-pori kayu jati umumnya lebih padat dan teksturnya lebih berminyak dibandingkan dengan kayu lainnya.
2. Kayu jati memiliki batang pohon yang pada umumnya lurus jadi memungkinkan mudah dijadikan sebagai bahan meubel apapun.
3. Tingkat kekerasan pada kayu jati lebih rendah sehingga kayu jati tidak mudah pecah dan melengkung walaupun dalam ukuran (ketebalan) yang tipis.
4. Warna dari kayu jati lebih terang dibandingkan dengan warna kayu lainnya.
5. Kayu jati pada umumnya memiliki berat rata-rata yang cukup ringan dibandingkan dengan kayu lainnya.

2.6.3 Kegunaan Kayu Jati Pada Meubel

Dalam penelitian ini, kayu yang digunakan adalah kayu jati. Kayu Berikut tabel mengenai kegunaan kayu jati sebagai bahan pembuatan meubel.

Tabel 2.6 Kegunaan Kayu Jati [14]

Usia Kayu	Kegunaan
1-5 tahun	Bangku
	Kayu Lapis

	Almari Pakaian
	Rangka Pintu dan Jendela
	Meja
	Bantalan
>5 tahun	Almari Hias
	Lemari Rak
	Lantai
	Papan Dinding
	Tempat Tidur
	Kursi
	Finir mewah

Berdasarkan tabel 2.6, kegunaan setiap kayu jati akan berbeda sesuai dengan usianya. Perbedaan kayu jati akan terlihat dari tekstur dan warnanya. Untuk warna kayu jati yang masih muda akan memiliki warna yang cerah atau mencolok, sedangkan warna kayu jati yang tua memiliki warna yang kusam. Tekstur kayu jati yang muda akan lebih kasar dibandingkan dengan tekstur kayu jati yang lebih tua.



(a) Kayu Jati Usia 1-5 tahun



(b) Kayu Jati Usia >5 tahun

Gambar 2.14 (a) dan (b) Perbedaan Usia Kayu Jati

2.7 Analisis Berorientasi Objek

Analisis dan desain berorientasi objek adalah cara baru dalam memikirkan suatu masalah dengan menggunakan model yang dibuat menurut konsep sekitar dunia nyata. Dasar pembuatan adalah objek, yang merupakan kombinasi antara 19 struktur data dan perilaku dalam suatu entitas. Model berorientasi objek bermanfaat untuk memahami masalah, komunikasi dengan ahli aplikasi, pemodelan suatu organisasi, menyiapkan dokumentasi serta perancangan program

dan basis data. Suatu model analisis dibuat untuk menggambarkan aspek dasar dari domain aplikasi, dimana model tersebut berisi objek yang terdapat dalam domain aplikasi termasuk deskripsi dari keterangan objek dan perilakunya.

2.7.1 Pemrograman Berorientasi Objek

Pemrograman berorientasi objek (*object oriented programming*) merupakan paradigma yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Bandingkan dengan logika pemrograman terstrukur, setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya. Model data berorpientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Konsep dasar dari pemrograman berorientasi objek adalah sebagai berikut :

1. *Class*

   Merupakan kumpulan atas definsi data dan fungsi-fungsi dalam suatu unit untuk suatu tujuan tertentu. Sebuah class adalah dasar dari modularitas dan struktur dalam pemrograman berorientasi objek. Sebuah class secara tipikal sebaiknya dapat dikenali oleh seorang non-programer sekalipun terkait dengan domain permasalah yang ada, dan kode yang terdapat dalam sebuah class sebaiknya (relatif) bersifat mandiri dan independen (sebagaimana kode tersebut digunakan jika tidak menggunakan OOP). Dengan modularitas, struktur dari sebuah program akan terkait dengan aspek-aspek dalam masalah yang akan diselesaikan melalui program tersebut. Cara seperti ini akan menyederhanakan pemetaan dari masalah ke sebuah program ataupun sebaliknya.

2. *Object*

   Setiap objek dibangun dari sekumpulan data (atribut) yang disebut variable untuk menjabarkan karakteristik khusus dari objek, dan juga terdiri dari sekumpulan method yang menjabarkan tingkah laku dari objek. Bisa dikatakan bahwa objek adalah sebuah perangkat lunak yang berisi

sekumpulan variable dan method yang berhubungan dan sebuah objek adalah sebuah instance (keturunan) dari class.

3. *Abstrak*

   Kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti. Setiap objek dalam sistem melayani sebagai model dari "pelaku" abstrak yang dapat melakukan kerja, laporan dan perubahan keadaannya, dan berkomunikasi dengan objek lainnya dalam sistem tanpa mengungkapkan bagaimana kelebihan ini diterapkan. Proses, fungsi, atau method dapat juga dibuat abstrak, dan beberapa teknik digunakan untuk mengembangkan sebuah pengabstrakan.

4. *Encapsulation*

   Merupakan dasar untuk pembatasan ruang lingkup program terhadap data yang diproses. Enkapsulasi juga untuk memastikan pengguna sebuah objek tidak dapat mengganti keadaan dalam dari sebuah objek denga cara yang tidak layak, hanya metode dalam objek tersebut yang diberi izin untuk mengakses keadaannya. Setiap objek mengakses *interface* yang menyebutkan bagaimana objek lainnya dapat berinteraksi dengannya. Objek lainnya tidak akan mengetahui dan tergantung kepada representasi dalam objek tersebut.

5. *Polymorphism*

   Polimorfisme merupakan konsep yang menyatakan bahwa sesuatu yang sama dapat mempunyai bentuk dan perilaku berbeda. Dengan kata lain bahwa operasi yang sama mungkin mempunyai perbedaan dalam class yang berbeda. Konsep ini tidak bergantung pada pemanggilan subrutin, bahasa orientasi objek dapat mengirim pesan. Metode tertentu yang berhubungan dengan sebuah pengiriman pesan tergantung kepada objek tertentu di mana pesan tersebut dikirim.

6. Pewarisan (*Inheritance*)

   Inheritance adalah teknik yang menyatakan bahwa anak dari objek akan mewarisi data/atribut dan metode dari induknya langsung. Atribut dan

metode dari objek induk diturunkan kepada anak objek, demikian seterusnya. Inheritance mempunyai arti bahwa atribut dan operasi yang dimiliki bersama diantara class yang mempunyai hubungan secara hierarki.

2.7.2 *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah himpunan struktur dan teknik untuk pemodelan desain program berorientasi objek (OOP) serta aplikasinya. UML adalah metodologi untuk mengembangkan sistem OOP dan sekelompok perangkat tool untuk mendukung pengembangan sistem tersebut. UML mulai diperkenalkan oleh Objek Management Group, sebuah organisasi yang telah mengembangkan model, teknologi, dan standar OOP sejak tahun 1980-an. Sekarang UML sudah mulai banyak digunakan oleh para praktisi OOP. UML merupakan dasar bagi perangkat (*tool*) desain berorientasi objek dari IBM. UML merupakan suatu bahasa yang digunakan untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan suatu sistem informasi. UML dikembangkan sebagai suatu alat untuk analisis dan desain berorientasi objek oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson. Penggunaan UML dalam industri terus meningkat, ini merupakan standar terbuka yang menjadikannya sebagai bahasa pemodelan yang umum dalam industri perangkat lunak dan pengembangan sistem.

UML adalah sebuah bahasa standar untuk pengembangan sebuah software yang dapat menyampaikan bagaimana membuat dan membentuk model-model, tetapi tidak menyampaikan apa dan kapan model yang seharusnya dibuat yang merupakan salah satu proses implementasi pengembangan software. UML tidak hanya merupakan sebuah bahasa pemrograman visual saja, namun juga dapat secara langsung dihubungkan ke berbagai bahasa pemrograman, salah satunya bahas pemrograman Java, atau bahkan dihubungkan secara langsung ke dalam sebuah object-oriented database. Begitu juga mengenai pendokumentasian dapat dilakukan, seperti : requirements, arsitektur, design, source code, project plan, test, dan prototypes.

Untuk dapat memahami UML, membutuhkan bentuk konsep dari sebuah bahasa model, dan mempelajari 3 (tiga) elemen utama dari UML seperti building block, aturan-aturan yang meyatakan bagaimana building block diletakkan secara bersamaan, dan beberapa mekanisme umum (*common*). Building block dalam UML adalah sebagai berikut :

1. *Things* (Benda)

   Things merupakan elemen yang paling penting dalam pemodelan UML, yaitu menjelaskan dari sebuah konsep. Komponen dari *things* dalah *classes, interface, collaboration, usecase, active classes, component, nodes, interactions, state machines, packages* dan *notes*.

2. *Relationship* (Hubungan)

   Relationship merupakan elemen yang menghubungkan *thing* dengan *thing* lainnya. Terdapat 4 macam *relationship* dalam UML, yaitu :

4. *Dependency*

   Dependency merupakan hubungan semantik antara dua benda yang apabila satu benda berubah maka benda yang berhubungan mengalami perubahan. Dependency digambarkan dengan sebuah panah dengan garis putus – putus.

5. *Association*

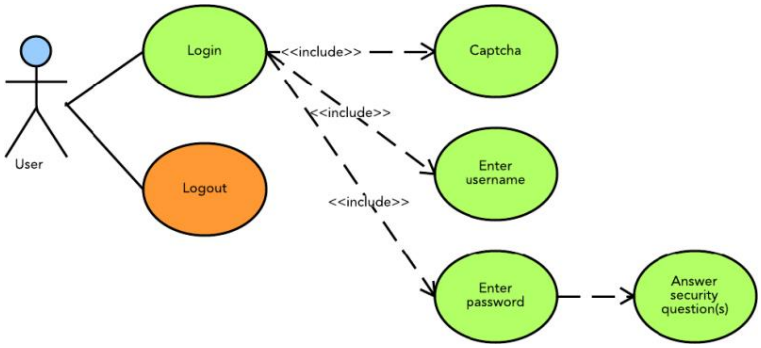
   Association merupakan hubungan antar benda yang berhubungan. hubungan ini menggambarkan hubungan structural antar seluruh atau sebagian objek. *Association* digambarkan dengan garis yang diberi label, nama dan status hubungan.

6. *Generalization*

   Generalization merupakan hubungan antara objek *child* yang menggantikan objek *parent.* Objek *child* atau anak memiliki pengaruh dalam hal struktur dan tingkah laku objek *parent. Generalization* digambarkan dengan garis panah.

2.7.2.1 *Usecase Diagram*

Usecase diagram digunakan untuk memodelkan bisnis proses berdasarkan perspektif pengguna sistem. *Usecase diagram* terdiri atas diagram untuk use case dan actor. Actor merepresentasikan orang yang akan mengoperasikan atau orang yang berinteraksi dengan sistem aplikasi. *Usecase* merepresentasikan operasi-operasi yang dilakukan oleh actor. *Usecase* digambarkan berbentuk elips dengan nama operasi dituliskan didalamnya. Actor yang melakukan operasi dihubungkan dengan garis lurus ke *Usecase*. Berikut ini adalah contoh gambar *usecase* dari proses login.



Gambar 2.15 *Usecase Diagram L*ogin

Dari contoh gambar *usecase diagram* diatas terdapat beberapa simbol, berikut ini adalah table yang menjelaskan simbol – simbol pada *usecase diagram* :

Tabel 2.7 Simbol U*secase Diagram*

Nama Komponen	Keterangan	Simbol
Use Case	*Use case* digambarkan sebagai lingkaran elips dengan nama *use case* dituliskan didalam elips tersebut	Use case
Actor	*Actor* adalah pengguna sistem. Actor tidak terbatas hanya manusia saja, jika sebuah sistem berkomunikasi dengan aplikasi lain dan membutuhkan input atau memberikan output, maka aplikasi tersebut juga bias dianggap sebagai actor.	
Association	*Association* digunakan untuk menghubungkan actor dengan use case.	

Nama Komponen	Keterangan	Simbol
	Asosiasi digambarkan dengan sebuah garis yang menghubungkan antara actor dengan use case.	

Ada tiga hal kegunaan dari usecase diagram, yaitu :

1. Menjelaskan fasilitas yang ada (requirments)

   Case baru selalu menghasilkan fasilitas baru ketika sistem di analisa, dan desing menjadi lebih jelas.
2. Komunikasi dengan klien

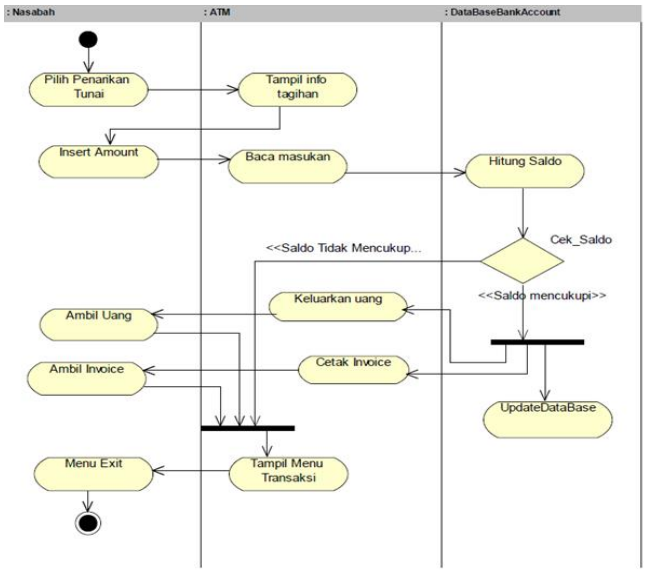
   Penggunaan notasi dan symbol dalam diagram usecase membuat pengembang lebih mudah berkomunikasi dengan klien-kliennya.
3. Membuat test dari kasus-kasus secara umum

   Kumpulan dari kejadian-kejadian untuk usecase bias dilakukan test kasus layak untuk kejadian-kejadian tersebut.

2.7.2.2 Activity Diagram

Activity diagram merupakan sebuah bagan bagaimana sistem atau aplikasi akan mencapai tujuannya. *Activity diagram* menunjukkan tindakan untuk mewakili suatu proses yang terjadi dalam sistem atau aplikasi. *Activity diagram* adalah sebuah proses bisnis yang terjadi dalam sistem atau aplikasi. Pada dasarnya diagram activity sering digunakan oleh *flowchart*. Diagram ini berfokus pada aktifitas-aktifitas yang terjadi terkait dengan suatu proses tunggal.

Diagram activity dapat dibagi menjadi beberapa jalur kelompok yang menunjukkan objek yang mana bertanggung jawab untuk suatu aktifitas. Peralihan tunggal (*single transition*) timbul dari setiap adanya *activity* (aktifitas), yang saling menghubungi pada aktifitas berikutnya. Berikut ini adalah gambar contoh dari diagram activity dari sebuah mesin ATM.



Gambar 2.16 *Activity diagram* mesin ATM

Dari contoh gambar *activity diagram* diatas terdapat beberapa symbol yang membentuknya, berikut ini adalah table yang menjelaskan simbol – simbol pada *activity diagram* :

Tabel 2.8 Simbol A*ctivity Diagram*

Simbol	Keterangan
	Titik awal
	Titik akhir
	*Activity*
	Pilihan untuk mengambil keputusan
	Fork; Digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan paralel menjadi satu.
	*Rake*; Menunjukkan adanya dekomposisi
	Tanda waktu
	Tanda pengiriman
	Tanda penerimaan
	Aliran akhir (*Flow Final*)

2.7.2.3 Class Diagram

Class dalam sebuah sistem atau aplikasi yang menggunakan *object programming* memiliki hubungan satu dengan yang lainnya. *Calss diagram* menggambarkan apa yang terjadi pada calss-class dalam sebuah sistem atau aplikasi. *Class diagram* tidak menjelaskan apa yang terjadi jika class pada sistem

atau aplikasi berhubungan dengan class yang lainnya. Terdapat tiga jenis relationship dalam *class diagram*, yaitu :

1. *Association*

   Association merupakan hubungan antara bagian dari dua class.

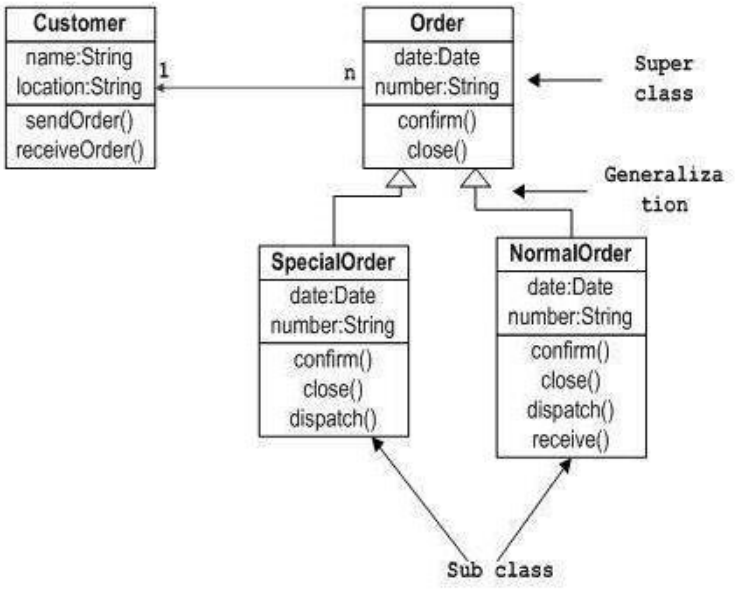
2. *Aggregation*

   Aggregation merupakan suatu association dimana salah satu kelasnya merupakan bagian dari suatu kumpulan. Aggregation memiliki titik pusat yang mencakup keseluruhan bagian.

3. *Generalization*

   Generalization merupakan suatu hubungan turunan dengan mengasumsikan satu class adalah suatu super class dari class yang lain.

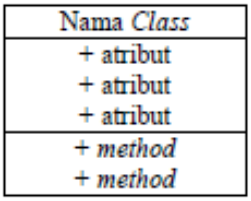
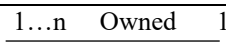
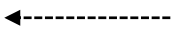
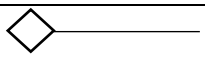
Berikut ini adalah contoh class diagram sebuah proses pemesanan .



Gambar 2.17 *Class diagram* pemesanan

Dari contoh gambar *calss diagram* diatas, terdapat beberapa simbol yang membentuknya, berikut ini adalah table yang menjelaskan simbol – simbol pada *class diagram*.

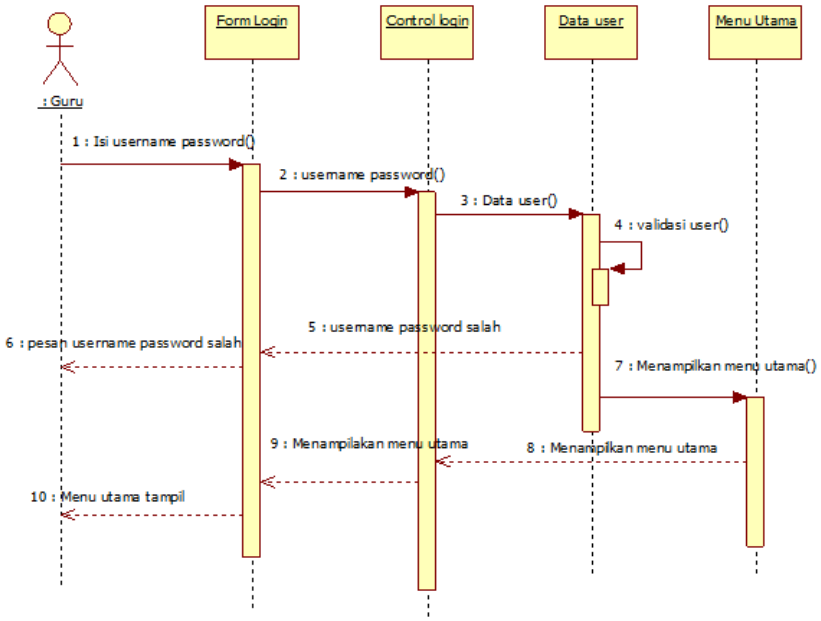
Tabel 2.9 Simbol C*lass Diagram*

Nama Komponen	Keterangan	Simbol
Class	Class adalah blok - blok pembangun pada pemrograman berorientasi obyek. Sebuah class digambarkan sebagai sebuah kotak yang terbagi atas 3 bagian. Bagian atas adalah bagian nama dari class. Bagian tengah mendefinisikan property/atribut class. Bagian akhir mendefinisikan method-method dari sebuah class.	
Association	Sebuah asosiasi merupakan sebuah relationship paling umum antara 2 class dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 class. Garis ini bisa melambangkan tipe-tipe relationship dan juga dapat menampilkan hukum-hukum multiplisitas pada sebuah relationship. (Contoh: One-to-one, one-to-many, many-to-many).	
Dependency	Kadangkala sebuah class menggunakan class yang lain. Hal ini disebut dependency. Umumnya penggunaan dependency digunakan untuk menunjukkan operasi pada suatu class yang menggunakan class yang lain. Sebuah dependency dilambangkan sebagai sebuah panah bertitik-titik.	
Aggregation	Aggregation mengindikasikan keseluruhan bagian relationship dan biasanya disebut sebagai relasi	

2.7.2.4 Sequence Diagram

Sequence diagram adalah bagian penting dalam pemodelan sistem atau aplikasi menggunakan UML. Sequence diagram atau diagram interaksi merupakan memodelkan urutan interaksi dari bagian-bagian yang membentuk sebuah sistem atau aplikasi. *Sequence diagram* menggambarkan interaksi mana yang akan bereaksi ketika penggunaan fungsi-fungsi dilakukan dan untuk tujuan apa sebuah unteraksi bereaksi. Secara sederhana *sequence diagram* merupakan cara sederhana dan efektif bagaiman cara berkomunikasi urutan peristiwa dalam

suatu interaksi [12]. Dibawah ini adalah contoh gambar dari *sequence diagram* login :



Gambar 2.18 *Sequence diagram* login

Dari contoh gambar *sequence diagram* diatas, terdapat beberapa simbol yang membentuknya, berikut ini adalah table yang menjelaskan simbol – simbol pada *sequence diagram* :

Tabel 2.11 Simbol *Sequence Diagram*

Nama Komponen	Keterangan	Simbol
Object	*Object* merupakan *instance* dari sebuah *class* dan dituliskan tersusun secara *horizontal*. Digambarkan sebagai sebuah *class* (kotak) dengan nama *object* didalamnya yang diawali dengan sebuah titik koma.	: Object1
Actor	*Actor* juga dapat berkomunikasi dengan *object* , maka *actor* juga dapat diurutkan sebagai kolom. Simbol *Actor* sama dengan simbol pada *Actor* Use Case Diagram.	
Lifeline	*Lifeline* mengindikasikan keberadaan sebuah *object* dalam basis waktu. Notasi untuk *Lifeline* adalah garis putus-putus *vertikal* yang ditarik dari sebuah *object* .	
Activation	*Activation* dinotasikan sebagai sebuah kotak segi empat yang digambar pada sebuah *lifeline*. mengindikasikan sebuah obyek yang akan melakukan sebuah aksi.	
Message	*Message*, digambarkan dengan anak panah horizontal antara *Activation* *Message* mengindikasikan komunikasi antara *object -object* .	message

2.8 Bahasa Pemrograman Java

Bahasa pemrograman yang digunakan untuk membuat aplikasi ini adalah bahasa pemrograman Java. Java adalah salah satu bahasa pemrograman OOP. Bahasa ini awalnya dibuat oleh James Gosling. 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 berbagai Mesin Virtual Java (JVM). Java merupakan bahasa pemrograman yang bersifat umum / non-spesifik (general purpose), dan secara khusus didesain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda. Saat ini java merupakan bahasa pemrograman yang populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web.

Versi awal Java ditahun 1996 sudah merupakan versi release sehingga dinamakan Java Versi 1.0. Java versi ini menyertakan banyak paket standar awal yang terus dikembangkan pada versi selanjutnya. Dibawah ini adalah beberapa paket yang disediakan java, diantaranya :

- java.lang, untuk kelas elemen-elemen dasar.
- java.io, untuk kelas input dan output, termasuk penggunaan berkas.
- java.util, untuk kelas pelengkap seperti kelas struktur data dan kelas penanggalan.
- java.net, untuk kelas TCP/IP yang memungkinkan berkomunikasi dengan komputer lain menggunakan jaringan TCP/IP.
- java.awt, kelas dasar untuk aplikasi antarmuka dengan pengguna (GUI).
- java.applet, kelas dasar aplikasi antarmuka untuk diterapkan pada penjelajah web.

2.8.1 Kelebihan bahasa pemrograman Java antara lain :

1. Multiplatform

   Kelebihan utama dari Java ialah dapat dijalankan di beberapa platform/ sistem operasi komputer, sesuai dengan prinsip tulis sekali, jalankan di mana saja. Dengan kelebihan ini pemrogram cukup menulis sebuah program Java dan dikompilasi (diubah, dari bahasa yang dimengerti manusia menjadi bahasa mesin/ bytecode) sekali lalu hasilnya dapat dijalankan di atas beberapa platform tanpa perubahan. Kelebihan ini memungkinkan sebuah program berbasis java dikerjakan diatas operating system Linux tetapi dijalankan dengan baik di atas Microsoft Windows. Platform yang didukung sampai saat ini adalah Microsoft Windows, Linux, Mac OS dan Sun Solaris. Penyebanya adalah setiap sistem operasi menggunakan programnya sendiri-sendiri (yang dapat diunduh dari situs Java) untuk meninterpretasikan *bytecode* tersebut.
2. OOP (Object Oriented Programming)
3. Class library yang lengkap

   Java terkenal dengan kelengkapan library/perpustakaan (kumpulan program program yang disertakan dalam pemrograman java) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas Java yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.
4. Bergaya C++

   Memiliki sintaks seperti bahasa pemrograman C++ sehingga menarik banyak pemrogram C++ untuk pindah ke Java. Saat ini pengguna Java sangat banyak, sebagian besar adalah pemrogram C++ yang pindah ke Java. Universitas-universitas di Amerika Serikat juga mulai berpindah dengan mengajarkan Java kepada murid-murid yang baru karena lebih mudah dipahami oleh murid dan dapat berguna juga bagi mereka yang bukan mengambil jurusan komputer.

5. Pengumpulan sampah otomatis

   Memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa C++ yang dipakai secara luas).

2.8.2 Kekurangan bahasa pemrograman Java :

1. Tulis sekali, jalankan di mana saja

   Masih ada beberapa hal yang tidak kompatibel antara platform satu dengan platform lain. Untuk J2SE, misalnya SWT-AWT bridge yang sampai sekarang tidak berfungsi pada Mac OS X.

2. Mudah didekompilasi

   Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini dimungkinkan karena kode jadi Java merupakan bytecode yang menyimpan banyak atribut bahasa tingkat tinggi, seperti nama-nama kelas, metode, dan tipe data. Hal yang sama juga terjadi pada Micosoft, NET Platform. Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/ *reverse-engineer*.

3. Penggunaan memori yang banyak

   Penggunaan memori untuk program berbasis Java jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan Pascal (lebih spesifik lagi, Delphi dan Object Pascal). Biasanya ini bukan merupakan masalah bagi pihak yang menggunakan teknologi terbaru (karena trend memori terpasang makin murah), tetapi menjadi masalah bagi mereka yang masih harus berkutat dengan mesin komputer berumur lebih dari 4 tahun.

2.9 Android Studio

2.9.1 Sejarah Android Studio

Android studio merupakan sebuah *Integrated Deveopment Environment* (IDE) untuk platform Android. Android studio ini diumumkan pada tanggal 16 Mei 2013 pada Konferensi Google I/O oleh Produk Manajer Google, Ellie

Powers. Android studio bersifat free dibawah *Apache License 2.0.* Android studio awalnya dimulai dengan versi 0.1 pada bulan Mei 2013, kemudian dibuat versi beta 0.8 yang dirilis pada bulan Juni 2014. Yang paling stabil dirilis pada bulan Desember 2014, dimulai dari versi 1.0. Berbasiskan JetBrainns' Intellij IDEA, studio di desain khusus untuk *Android Development.* Ini sudah bisa di download untuk Windows, Mac OS X, dan Linux.

2.9.2 Fitur Android Studio

Sebagai pengembangan dari Eclipse, Android Studio mempunyai banyak fitur baru dibandingkan dengan Eclipse IDE. Berbeda dengan Eclipse yang menggunakan Ant, Android Studio menggunakan Gradle sebagai build environment. Fitur – fitur lain yang terdapat di Android Studio adalah sebagai berikut :

1. Live Layout : WYSIWYG Editor – Live Coding – Real time App Rendering.
2. Developer Console : optimization tips, assistance for translation, referral tracking, campaigning and promotions – Usage Metrics.
3. Provision for beta releases and staged rollout.
4. Gradle-based build support.
5. Android – specific refactoring and quick fixes.
6. Lint tools to catch performance, usability, version compatibility and other problems.
7. ProGuard and app-signing capabilities.
8. Template-based wizards to create common Android designs and components.
9. A rich layout editor that allows user to drag-and-drop UI components, option to preview layouts on multiple screen configurations.
10. Support for building Android Wear apps.
11. Built-in support for Google Cloud Platform, enabling integration with Google Cloud Messaging and App Engine.