BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Adapun penelitian terkait yang akan digunakan sebagai sumber acuan yang relevan dan terkini yaitu:

Berdasarkan penelitian yang dilakukan Linda Rahmayanti ((2019) dengan judul "Sistem Absensi Mahasiswa Berdasarkan Citra Wajah Menggunakan Metode *Principal Component Analysis* (PCA)" peneitian ini memiliki kontribusi nilai yang cukup besar dalam kontrak kuliah. Oleh karena itu keaslian data absensi sangat diperlukan. Metode *Principal Component Analysis* (PCA) adalah salah satu metode yang banyak digunakan dalam ektrasi ciri citra, dimana pada proses deteksi maupun pengenalan dapat mengenali bagian wajah terlepas dari *background* yang digunakan.

Berdasarkan penelitian yang dilakukan Fahmi Syuhada (2018) dengan judul "Pengenalan Wajah Untuk Sistem Kehadiran Menggunakan Metode *Eigenface* dan *Euclidean Distance*" Permasalahan yang diangkat yaitu bagaimana komputer yang sudah terkoneksi kamera *webcam* dapat mengenali wajah seseorang walaupun orang tersebut tidak secara langsung melakukan proses absensi. Hal ini dilakukan melalui kamera webcam yang terkoneksi akan melakukan proses perekam dan pengenalan terhadap objek wajah yang dideteksi secara *real-time*.

Berdasarkan penelitian yang dilakukan oleh Muhammad Yusuf (2016) dengan judul “Rancang Bangun Aplikasi Absensi Perkuliahan Mahasiswa Dengan Pengenalan Wajah” Proses absensi yang dilakukan secara manual dinilai kurang efektif karena terbukanya kesempatan melakukan kecurangan. Selain itu, proses rekapitulasi manual membutuhkan waktu yang lama. Sistem absensi dengan teknologi dapat diterapkan untuk membantu proses absensi dan rekapitulasi yang efektif. Pada tugas akhir ini, teknologi yang digunakan adalah sistem pengenalan wajah. Pembuatan aplikasi absensi dengan pengenalan wajah ini menggunakan metode *Eigenface* untuk melakukan proses pengenalan wajah.

Berdasarkan penelitian yang dilakukan oleh Muhammmad Rizki Muliawan (2015) dengan judul “Implementasi Pengenalan Wajah Dengan Metode Eigenface Pada Sistem Absensi**”** Secara umum proses absensi menggunakan pengenalan wajah ini dilakukan dengan memasukkan data wajah terlebih dahulu beserta password dari masing-masing orang, setelah itu dilakukan proses pemindaian untuk proses absensi. Metode eigenface dari opencv ini mencari data wajah yang mendekati dengan data wajah yang ada di database. Pada pengujian penelitian ini hasil yang didapat berbeda-beda antara wajah satu dengan wajah yang lainnya, pada saat database berisi 10 data wajah, hasil rata-rata persentase kecocokan mencapai 88%, sedangkan pada saat database berjumlah 20 data wajah, hasil rata-rata persentase kecocokan mencapai 52%.

Berdasarkan penelitian ini bertujuan untuk merancang sistem yang baru dalam absensi wajah karyawan pada APJ Gatsu Berbasis Android dengan menerapkan metode Dynamic Times Wrapping. Aplikasi yang di rancang oleh

penulis dapat mempermudah pekerjaan perusahaan dan karyawan khususnya dalam pelaksanaan absensi karyawan, sistem yang akan diusulkan penulis menggunakan aplikasi Android Studio dan Mysql.

II.2. Landasan Teori

II.2.1. Aplikasi

Istilah aplikasi berasal dari bahasa Inggeris *application* yang berarti penerapan, lamaran ataupun penggunaan. Sedangkan secara umum, pengertian aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanakan suatu fungsi bagi pengguna jasa aplikasi serta jasa pengguna aplikasi lain yang dapat digunakan oleh pengguna yang akan dituju. Menurut kamus komputer eksekutif, pengertian aplikasi merupakan pemecahan masalah yang biasanya berpacu pada sebuah komputasi yang diinginkan atau diharapkan maupun pemrosesan data yang diharapkan. Aplikasi biasanya berupa perangkat lunak yang berbentuk *Software* yang berisi kesatuan perintah atau program yang dibuat untuk melaksanakan sebuah pekerjaan yang diinginkan. (Inayah, 2015 : 3).

II.2.2. Pengenalan Pola

Pola adalah suatu bentuk dimana masing-masing pola memiliki ciri-cirinya. Ciri-ciri tersebut digunakan utuk mem-bedakan suatu pola dengan pola yang lainnya. Ciri yang baik adalah ciri yang memiliki daya pembeda yang tinggi, se-hingga pengelompokkan pola berdasarkan ciri yang dimiliki dapat dilakukan dengan keakuratan yang tinggi. (Muhammad Rizki : 2015)

II.2.3. Pengenalan Wajah

Pengenalan wajah adalah salah satu teknologi biometrik yang telah banyak diaplikasikan dalam sistem keamanan selain pengenalan retina mata, pengenalan sidik jari dan iris mata. Dalam aplikasinya sendiri pengenalan wajah menggunakan sebuah kamera untuk menangkap wajah seseorang kemudian dibandingkan dengan wajah yang sebelumnya telah disimpan di dalam *database* tertentu. Pengenalan wajah melibatkan banyak variabel, misalnya citra sumber, cira hasil pengolahan citra, citra hasil ekstraksi dan data profil seseorang. Dibutuhkan juga alat pengindra berupa sensor kamera dan metode untuk menentukan apakah citra yang ditangkap oleh webcam tergolong wajah manusia atau bukan, sekaligus untuk menentukan informasi profil yang sesuai dengan citra wajah yang dimaksud. (Muhammad Rizki : 2015)

Pengenalan wajah merupakan salah satu pendekatan pengenalan pola untuk keperluan identifikasi wajah seseorang dengan pendekatan biometrik. Suatu biometrik bersifat unik sehingga dapat digunakan untuk mengenali identitas seseorang. Proses pengenalan biometrik dapat dibagi menjadi dua karakteristik, yaitu secara fisik dan secara perilaku. Biometrik fisik berasal dari pengukuran dan data yang ada langsung dari bagian manusia misalnya pengenalan sidik jari, pengenalan wajah, iris, retina, dan tangan. Sedangkan biometrik perilaku berasal dari pengukuran dan data yang berasal dari tindakan seperti suara, tanda tangan, dan *keystrokes*. Sistem biometrik mengacu pada terintegrasinya antara perangkat keras dan perangkat lunak untuk melakukan proses identifikasi dan verifikasi. (Muhammad Yusuf : 2016)

II.2.4. *Dynamic Time Warping*

DTW (*Dynamic Time Warping*) adalah metode untuk menghitung jarak antara dua data time series. Keunggulan DTW dari metode jarak yang lainnya adalah mampu menghitung jarak dari dua vektor data dengan panjang berbeda. Jarak DTW diantara dua vektor dihitung dari jalur pembengkokkan optimal (*optimal warping path*) dari kedua vektor tersebut. (Romi Wiryadinata : 2018)

Pose yang digunakan untuk data pelatihan pada data pelatihan adalah sebagai berikut :

1. Wajah normal 1
2. Wajah normal + menutup wajah
3. Wajah tersenyum lebar
4. Wajah dengan mulut terbuka
5. Wajah dengan posisi menghadap ke kiri 20°
6. Wajah dengan posisi menghadap ke kanan 20°
7. Wajah dengan memakai kacamata
8. Wajah normal 2
9. Wajah dengan sedikit mendongak ke atas
10. Wajah dengan lidah menjulur keluar (Romi Wiryadinata : 2018)

Dynamic Time Warping (DTW) merupakan salah satu ukuran jarak yang paling relevan dalam analisis time series . DTW merupakan sebuah metode yang menggunakan pendekatan pemrograman dinamis untuk menyelaraskan dua time series dengan menghitung jarak paling minimal antar titik data. Penyelarasan dua time series menggunakan DTW membutuhkan sebuah matriks berukuran $N \times M$

yang berisikan elemen jarak $D(i,j)$ antara dua titik xi dan yj dengan $i \in [1, N]$,$j \in [1, M]$ [18]. Algoritma DTW dimulai dengan membangun matriks jarak $\boldsymbol{C} \in \mathbb{R}$ NxMyang mewakili semua jarak berpasangan antara time series X dan Y di mana $X = (x1 , x2 , \dots , xN)$, $N \in \mathbb{N}$ dan $Y = (y1 , y2 , \dots , yM)$, $M \in \mathbb{N}$. Matriks jarak ini disebut dengan local cost matrix untuk penyelarasan dua time series X dan Y (Inas Salsabila : 2019)

DTW (DynamicTime Warping) adalah metode untuk menghitung jarak antara dua data time series. Keunggulan DTW dari metode jarak yang lainnya adalah mampu menghitung jarak dari dua vektor data dengan panjang berbeda. Jarak DTW diantara dua vektor dihitung dari jalur pembengkokkan optimal (optimal warping path) dari kedua vektor tersebut. Sebuah teknik yang cukup popular di awal perkembangan teknologi pengolahan sinyal wicara adalah dengan memanfaatkan sebuah teknik dynamic-programming yang juga lebih dikenal sebagai Dynamic Time Warping (DTW). Teknik ini ditujukan untuk mengakomodasi perbedaan waktu antara proses perekaman saat pengujian dengan yang tersedia pada template sinyal referensi. Prinsip dasarnya adalah dengan memberikan sebuah rentang 'steps' dalam ruang (dalam hal ini sebuah frame-frame waktu dalam sample, frame-frame waktu dalam template) dan digunakan untuk mempertemukan lintasan yang menunjukkan local match terbesar (kemiripan) antara time frame yang lurus. Total `similarity cost' yang diperoleh dengan algorithm ini merupakan sebuah indikasi seberapa bagus sample dan template ini memiliki kesamaan, yang selanjutnya akan dipilih best-matching template. (Darma Putra : 2018)

Dynamic Time Warping Salah satu metode yang banyak dipakai untuk menghitung jarak / tingkat kesamaan dari dua datatime series, seperti data gestureadalah Dynamic Time Warping (DTW). DTW adalah metode untuk menghitung tingkat kesamaan antara dua time series data yang mungkin berbeda panjangnya dengan menghitung jarak dua data time series tersebut secara optimal. DTW menggunakan prinsip Dynamic Programming yaitu dengan mencari solusi optimal berdasarkan solusi paling optimal pada submasalahnya. Pada DTW jarak antara dua data time series / deret data yang optimal didapat dari jarak optimal antar poin data dalam data time series. Tujuan dari DTW adalah menemukan Optimal Warping Path atau pasangan indeks poin data berjarak paling optimal pada data time series yang dibandingan. Sebagai contoh dari dua data time series berikut: (Sam Chaerul : 2016)

Q = { q1, q2, q3,..., qi ,..., qn }

C = { c1, c2, c3, c4, c5,..., cj ,...,cm }

Perhitungan dengan DTW diawali dengan membuat matriks (cost matrix) berukuran n x m selanjutnya perhitungan jarak antara dua poin data pada indeks idan j dengan menggunakan nilai absolut |qi – cj | atau fungsi jarak seperti Euclidean Distance. (Sam Chaerul : 2016)

$$d(q_i, c_j) = \sqrt{(q_i - c_j)^2}$$

Setiap sel pada cost matrix akan diisi dengan nilai jarak pada indeks yang dievaluasi dan ditambahkan dengan nilai jarak pada indeks sebelumnya atau dinyatakan sebagai global distance D(i,j). Kemudian optimal warping path dibuat

dengan menelusuri nilai paling minimum diantara D(i-1, j-1), D(i-1, j) atau D(i, j-1) yang sudah ditambahkan dengan local distanced(qi, cj)yang dapat dinyatakan sebagai berikut: (Sam Chaerul : 2016)

$$D(i,j) = d(q_i,c_j) + min\{D(i-1,j-1), D(i-1,j), D(i,j-1)\}$$

Nilai jarak / optimal warping distance ditemukan pada indeks terakhir indeks ke-n dan ke-m pada cost matrix. Permasalahan utama yang muncul pada DTW adalah apabila dimensi data / ukuran data semakin besar maka dibutuhkan sumberdaya untuk komputasi yang lebih besar lagi atau hal ini biasa disebut sebagai curse of dimensionality pada masalahmasalah yang diselesaikan dengan metode yang menggunakan prinsip Dynamic Programming (Sam Chaerul : 2016)

Algoritma DTW merupakan salah satu metode yang digunakan untuk membandingkan dua buah sequence di waktu dan kecepatan yang berbeda. Keunggulan dari algoritma DTW sendiri jika dibandingkan dengan metode jarak yang lainnya adalah mampu menghitung jarak atau selisih antar dua vektor yang dibandingkan dengan menghitung dari optimal warping path atau jalur pembengkokan optimal. Dari tiga proses utama teknik DTW, perhitungan yang sering digunakan adalah dengan menggunakan metode pemrograman dinamis. (Fawwaz Muhammad S : 2018)

$$D = \sqrt{(x1 - y1)\,2 + (x2 - y2)\,2}$$

Keterangan :

(X1,X2) = Sinyal Uji

(Y1,Y2) = Sinyal Latih

Satu masalah yang cukup rumit dalam pengenalan suara adalah proses perekaman yang terjadi seringkali berbeda durasinya, meskipun kata atau kalimat yang diucapkan sama. Bahkan untuk satu suku kata/ vokal yang sama seringkali terjadi dalam durasi yang berbeda. Sebagai akibatnya proses pencocokan antara sinyal uji dengan sinyal record suara di database seringkali tidak menghasilkan nilai yang optimal. Sebuah teknik yang cukup popular di awal perkembangan teknologi pengolahan sinyal pengenalan suara (voice recognition) adalah teknik Dynamic Time Warping (DTW) yang juga lebih dikenal sebagai dynamic programming. Teknik ini ditujukan untuk mengakomodasi perbedaan waktu antara proses perekaman saat pengujian dengan suara yang tersedia pada record suara di database. Prinsip dasarnya adalah dengan memberikan sebuah rentang 'steps' dalam ruang (dalam hal ini sebuah frame-frame waktu dalam sampel, frame-frame waktu dalam database) dan digunakan untuk mempertemukan lintasan yang menunjukkan local match terbesar (kemiripan) antara time frame yang lurus. Total `similarity cost' yang diperoleh dengan algoritma ini merupakan sebuah indikasi seberapa bagus sampel suara dan record suara di database ini memiliki kesamaan, yang selanjutnya akan dipilih best-matching record suara di database. (Destian Tri Handoko : 2018)

Gerak isyarat merupakan data sekuensial. Tiap elemen data sekuensial diukur dan disimpan pada satu periode waktu yang tetap antara elemen satu dengan yang berikutnya. Salah satu metode untuk membandingkan dua data sekunesial dengan panjang yang berbeda adalah dengan algoritma Dynamic Time Warping (DTW). DTW merupakan algoritma yang digunakan untuk mengukur

kemiripan antara dua sekuensial dengan panjang atau jumlah data yang berbeda. DTW mencocokkan dua sekuensial dengan menghitung tranformasi temporal sehingga keduanya dapat diselaraskan (aligned). Penyelarasan (alignment) adalah optimal jika terukur jarak kumulatif terkecil antara dua sampel yang telah diselaraskan. Jika diasumsikan terdapat dua data sekuensial, Q dan C, dengan panjang masing-masing n dan m sebagaimana persamaan 1 dan persamaan 2, |(Mohammad Iqbal : 2018)

Q = q1, q2, ..., qi , ..., qn

C = c1, c2, ..., cj , ..., cm

Maka untuk menyelaraskan (align) kedua sekuensial tersebut mengunakan DTW, dibentuk matriks m x n dengan elemen matriks (i,j) berupa nilai jarak d(qi ,cj) antara dua titik qi dan, yaitu d(qi ,cj) = (qi– cj) 2 . Setiap elemen matriks (i,j) berhubungan dengan penyelarasan (alignment) antara titik qi dan cj . Warping path W merupakan sekelompok elemen matriks yang berdampingan yang mendefinisikan pemetaan antara Q dan C. Elemen ke-k dari W dirumuskan sebagai wk = (i,j)k , sehingga didapat persamaan 3 |(Mohammad Iqbal : 2018)

W = w1, w2, ..., wk , ..., wK dengan: $\max(m,n) \leq K < m+n-1$

Sedangkan path didefinisikan sebagai jarak kumulatif D(i,j) yaitu jarak d(qi ,cj) untuk elemen tersebut ditambah dengan minimum dari jarak kumulatif dari elemen bertetanggaan (adjacent), sebagaimana persamaan 4. |(Mohammad Iqbal : 2018)

D(i,j) = d(qi ,cj) +min{D(i-1,j-1),D(i-1,j),D(i,j-1)

Setelah didapatkan warping path yang optimal maka jarak atau warping cost dihitung berdasarkan persamaan 5 |(Mohammad Iqbal : 2018)

$$\mathrm{DTW}(Q,C) = \min \left\{ \sqrt{\sum_{k=1}^{K} w_k} \right\}$$

II.2.5. *Android*

Android adalah sistem operasi yang berbasis Linux untuk telepon seluler seperti telepon pintar dan komputer tablet. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, TMobile, dan Nvidia. Pada saat perilisan perdana Android, 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, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler (Harni Kusniyati Dan Dicky : 2018)

Android merupakan sistem operasi *mobile*. *Android* tidak membedakan antara aplikasi inti dengan aplikasi pihak ketiga. *Application Programming Interface* (API) yang disediakan menawarkan akses ke *hardware*, maupun data data ponsel sekalipun, atau data sistem sendiri. *Android* merupakan sebuah sistem

operasi perangkat *mobile* berbasis *linux* yang mencangkup sistem operasi, *middleware*, dan aplikasi. Beberapa pengertian lain dari *Android*, yaitu :

1. Merupakan *platform* terbuka (*Open* Source) bagi para pengembang (*Programer*) untuk membuat aplikasi.
2. Merupakan sistem operasi yang dibeli *Google Inc*. dari *Android Inc*.
3. Bukan bahasa pemrograman, tetapi hanya menyediakan lingkungan hidup atau *run time enviroment* yang disebut DVM (*Dalvik Virtual Machine*) yang telah dioptimasi untuk alat/*device* dengan sistem memori yang kecil. (Ni Kadek Ceryna, dkk: 2018 : 101)

Android merupakan sistem operasi berbasis Linux yang dirancang untuk perangkat bergerak layar sentuh (*mobile*) seperti *smartphone* dan *tablet*. *Android* adalah sistem operasi yang bersifat *open source* dan *google* merilis kodenya di bawah lisensi *apache*. *Android* awalnya dikembangkan oleh *Android*, Inc., dengan dukungan finansial dari *google* dan kemudian google membelinya pada tahun 2005. Android dibuat menggunakan bahasa pemrograman C(core), C++, Java(UI) dan dukungan berbagai bahasa pemrograman lain. Berdasarkan *dashboard developer android*, *Android* 8.0/1 *Oreo* adalah versi *Android* yang paling banyak digunakan, yakni sekitar 28.3% dari keseluruhan perangkat *Android* di seluruh dunia. Hingga saaat ini, *Android* sistem operasi berbasis *mobile* yang paling banyak digunakan di seluruh dunia dan android sudah di rilis versi 10 pada 3 september 2019. (Roberto Kaban, dkk: 2019)

II.2.6. Andorid Studio

Android Studio adalah *Integrated Development Enviroment* (IDE) untuk sistem operasi *Android* yang dibangun berdasarkan perangkat lunak IntelliJ IDEA. Selain editor kode dan alat pengembang yang didesain khusus untuk pengembangan *android*, *Android studio* merupakan pengganti dari *Eclipse Android Development Tools* (ADT) yang sebelumnya merupakan IDE utama untuk pengembangan aplikasi *android*. *Android studio* menyediakan banyak fitur untuk meningkatkan produktivitas dalam pengembangan aplikasi berbasis android seperti tersedianya sistem pembangunan berbasis *gradle* yang *fleksibel*, emulator yang memiliki banyak fitur, integrasi dengan GitHub, dukungan bawaan untuk *Google Cloud Platform* sehingga mempermudah pengembang aplikasi untuk mengintegrasikan *Google Cloud Messaging* dan *App Engine*. (Roberto Kaban, dkk: 2019)

II.2.7 Cara Kerja Android

Untuk membangun sebuah sistem operasi *Android* dapat menggunakan Mac, *Windows* PC, ataupun *Linux*. *Tools* yang dibutuhkan gratis dan dapat di *download* dari *web*. Berikut adalah beberapa *tools* yang digunakan untuk membangun aplikasi *android*.

1. JDK (*Java Development Kit*)
2. Android SDK
3. ADT (*Android Development Tools*) (Ni Kadek Ceryna, dkk : 2018 : 102)

II.2.8 Perkembangan Android

Adapun perkembangan Android adalah sebagai berikut :

a. Android versi 1.1

Android memang diluncurkan pertama kali pada tahun 2007, namun sistem operasi ini mulai dirilis dan diterapkan ke berbagai gadget pada tanggal 9 Maret 2009 silam. Android versi 1.1 merupakan Android awal yang dimana versi ini baru memberikan sentuhan dibeberapa aplikasinya seperti sistem antar muka bagi pengguna (user interface) yang lebih baik, serta beberapa aplikasi yang lain.

b. Android versi 1.5 (Cupcake)

Pada bulan Mei 2009 Android kembali mengalami perubahan versi. Android versi 1.1 kemudian disempurnakan dengan Android versi 1.5 atau yang dikenal sebagai Android Cupcake.

c. Android versi 1.6 (Donut)

Donut (versi 1.6) diluncurkan dalam tempo kurang dari 4 bulan semenjak peluncuran perdana Android Cupcake, yaitu pada bulan September 2009.

d. Android versi 2.0/2.1 (Eclair)

Masih ditahun yang sama, Android kembali merilis operating sistem versi terbarunya, yaitu Android versi 2.0/2.1 Eclair. Android Eclair diluncurkan oleh Google 3 bulan setelah peluncuran.

e. Android versi 2.2 (Froyo: Frozen Yoghurt)

Butuh 5 bulan bagi Google untuk melakukan regenerasi dari Android Eclair versi sebelumnya ke versi Froyo Frozen Yoghurt. Pada tanggal 20 Mei 2010, Android versi 2.2 alias Android Froyo ini dirilis.

f. Android versi 2.3 (Gingerbread)

7 bulan kemudian Android kembali melakukan gebrakan dengan merilis kembali Android versi 2.3 atau yang dikenal sebagai Android Gingerbread.

g. Android versi 3.0/3.1 (Honeycomb)

Pada bulan Mei 2011 Android versi 3.0/3.1 atau Android Honeycom dirilis. Android Honeycomb merupakan sebuah sistem operasi Android yang tujuannya memang dikhususkan bagi penggunaan tablet berbasis Android.

h. Android versi 4.0 (ICS: Ice Cream Sandwich)

Android ICS atau Ice Cream Sandwich juga dirilis pada tahun yang sama dengan Honeycomb, yaitu pada bulan Oktober 2011.

i. Android versi 4.1 (Jelly Bean)

Android Jelly Bean merupakan versi Android yang terbaru pada saat ini. Salah satu gadget yang menggunakan sistem operasi Jelly Bean adalah Google Nexus 7 yang diprakarsai oleh ASUS, vendor asal Taiwan yang juga menjadi teman satu kampung halaman dengan Acer.

j. Android versi 4.4 (Kit Kat)

Kehadiran android kitkat merupakan pelucuran produk OS anyar yang dilucurkan pada 4 september 2013, sebelumnya banyak kabar beredar jikalau android akan meluncurkan OS baru yang bernama Android Key Lime Pie

namun setelah di analisa tidak sesuai dengan ejaan orang umum, sehingga namanya diganti dengan OS Android KitKat yang sebagian besar orang sudah familiar dengan itu.

k. Android versi 5.0.2 (Lollipop)

Android Lollipop merupakan keberadaan OS Android yang memang saat ini sudah menjadi trend baru di industri smartphone, hal ini tak lepas dari keunikan dan kelebihan yang banyak di miliki dari OS tersebut. Kehadiran android versi ini amat di nanti oleh sekian banyak orang karna diharapkan sistem operasi Lollipop ini bias lebih baik dibandingkan versi-versi sebelumnya.

l. Android versi 6.0 (Marshmallow)

Android 6.0 Marshmallow adalah versi dari sistem operasi mobile Android. Pertama kali diperkenalkan Mei 2015 di Google I / O di bawah kode nama Android M, secara resmi dirilis pada Oktober 2015. Android Marshmallow memperkenalkan model izin aplikasi didesain ulang sekarang ada hanya delapan kategori izin, dan aplikasi yang tidak lagi secara otomatis diberikan semua hak akses mereka ditentukan pada waktu instalasi. (Harni Kusniyati Dan Dicky : 2018)

II.2.9. Basis Data (*Database*)

Pangkalan data atau basis data (*database*) adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis

data tersebut. Perangkat lunak yang digunakan untuk mengelola dan memanggil kueri (*query*) basis data disebut sistem manajemen basis data (*database management system*, DBMS). Sistem basis data dipelajari dalam ilmu informasi. Konsep dasar dari basis data adalah kumpulan dari catatan-catatan, atau potongan dari pengetahuan. Sebuah basis data memiliki penjelasan terstruktur dari jenis fakta yang tersimpan di dalamnya: penjelasan ini disebut skema. Model yang umum digunakan sekarang adalah model relasional, yang mewakili semua informasi dalam bentuk tabel-tabel yang saling berhubungan dimana setiap tabel terdiri dari baris dan kolom. Model yang lain seperti model hierarkis dan model jaringan menggunakan cara yang lebih eksplisit untuk mewakili hubungan antar tabel (Neni Purwati dan Hendra Kurniawan, 2015 : 50).

II.2.10. Normalisasi

Normalisasi adalah proses pembentukan struktur basis data sehingga sebagian besar ambiguity bisa dihilangkan. Normalisasi merupakan sebuah teknik dalam logical desain sebuah basis data relasional yang mengelompokkan atribut dari suatu tabel sehingga membentuk struktur tabel yang normal. Adapun kriteria tabel dikatakan normal adalah ketika tidak ada kerangkapan data (redudansi data). (Puspitasari et al., 2016)

Tujuan dari normalisasi adalah untuk :

1. Untuk menghilangkan kerangkapan data sehingga meminimumkan pemakaian *storage* yang dipakai oleh *base relations* (*file*).
2. Untuk mengurangi kompleksitas.

3. Untuk mempermudah pemodifikasian data.

II.2.10.1. Proses Normalisasi

1. Proses Normalisasi
   a. Data diuraikan dalam bentuk tabel, selanjutnya dianalisis berdasarkan persyaratan tertentu kebeberapa tingkat.
   b. Apabila tabel yang diuji belum memenuhi persyaratan tertentu maka tabel tersebut pelu dipecah menjadi beberapa tabel yang lebih sederhana sampai memenuhi bentuk yang optimal.
2. Tahapan Normalisasi :
   1) Bentuk tidak normal : Menghilangkan perulangan *grup*.

Tabel II.1. Contoh bentuk tidak normal (Unnormal)

No-Mhs	Nama Mhs	Jurusan	Kode-MK	Nama-MK	Kode Dosen	Nama Dosen	Nilai
2683	Welli	MI	M1350	Manajemen DB	B104	Ati	A
			M1465	Analisis Perc. Sistem	B317	Dita	B
5432	Bakti	AK	M1350	Manajemen DV	B104	Ati	C
			Akn201	Akuntansi	D310	Lia	B
			MKT300	Dasar Pemasaran	B212	Lola	A

Sumber : Mukhlisulfatih Latief : 2016

2) Bentuk Normal pertama (1NF) : Menghilangkan ketergantungan sebagian.

Yaitu : suatu relasi dikatakan sudah memenuhi bentuk normal kesatu bila setiap data bersifat atomik yaitu setiap irisan baris dan kolom hanya mempunyai satu nilai data.

Tabel II.2. Contoh Bentuk Normal Pertama (1NF)

No-Mhs	Nama Mhs	Jurusan	Kode-MK	Nama-MK	Kode Dosen	Nama Dosen	Nilai
2683	Welli	MI	M1350	Manajemen DB	B104	Ati	A
2683	Welli	MI	M1465	Analisis Perc. Sistem	B317	Dita	B
5432	Bakti	AK	M1350	Manajemen DV	B104	Ati	C
5432	Bakti	AK	Akn201	Akuntansi	D310	Lia	B
5432	Bakti	AK	MKT300	Dasar Pemasaran	B212	Lola	A

Sumber : Mukhlisulfatih Latief : 2016

3) Bentuk Normal kedua (2NF) : Menghilangkan ketergantungan transitif.

Yaitu : suatu relasi dikatakan sudah memenuhi bentuk normal kedua bila relasi tersebut sudah memenuhi bentuk normal kesatu dan atribut yang bukan *key* sudah tergantung penuh terhadap *key*-nya.

Tabel II.3. Contoh Bentuk Normal Kedua (2NF)

Kode-MK	Nama-MK	Kode Dosen	Nama Dosen
M1350	Manajemen DB	B104	Ati
M1465	Analisis Perc. Sistem	B317	Dita
M1350	Manajemen DV	B104	Ati
Akn201	Akuntansi	D310	Lia
MKT300	Dasar Pemasaran	B212	Lola

Sumber : Mukhlisulfatih Latief : 2016

4) Bentuk Normal ketiga (3NF) : Menghilangkan anomali-anomali hasil dari ketergantungan fungsional. Yaitu : suatu relasi dikatakan sudah memenuhi bentuk normal ketiga bila relasi tersebut sudah memenuhi bentuk normal kedua dan atribut yang bukan key tidak tergantung transitif terhadap *key*-nya.

Tabel II.4. Contoh Tabel Mahasiswa Dan Tabel Kuliah (3NF)

No_Mhs	Nama Mhs	Jurusan
2683	Welli	MI
5432	Bakti	AK

Sumber : Mukhlisulfatih Latief : 2016

Normalisasi adalah proses penyusunan *table-table* yang tidak redundan (*double*) yang dapat menyebabkan anomali pada saat terjadi manipulasi data seperti tambah, ubah dan hapus. Tujuan dari normalisasi adalah:

a. Untuk meghilangkan kerangkapan data.

b. Untuk mengurangi, kompleksitas.

c. Untuk mempermudah pemodifikasian data. (Trisnawati, 2016)

II.2.11. MySQL

MySQL (*My Structure Query Language*) merupakan sebuah program pembuat *database* yang bersifat *Open Source,* artinya semua orang dapat menggunakannya dan dapat dijalankan pada semua *platform* baik *Windows* maupun linux. *MySQL* juga merupakan sebuah perangkat lunak sistem manajemen basis data *SQL* yang bersifat jaringan sehingga dapat digunakan intuk aplikasi multi *user*. *MySQL* juga sering dikenal dengan nama sistem manajemen *database* relasional. Suatu *database* relasional menyimpan data dalam *table* yang terpisah. *Tabel -table* tersebut terhubung oleh suatu relasi terdefinisi yang memungkinkan memperoleh kombinasi data dari beberapa table dalam suatu permintaan. Untuk administrasi *database*, seperti pembuatan *database*, pembuatan tabel, dan sebagainya dapat digunakan aplikasi berbasis web seperti *PHP MyAdmin* dengan aplikasi *XAMPP*. (Saipul Anwar, 2016 : 96**)**

II.2.12. XAMPP

XAMPP adalah perangkat lunak bebas, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai server yang berdiri sendiri (*localhost*), yang terdiri atas program *Apache HTTP Server*, *MySQL database*, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman *PHP* dan *Perl*. Nama *XAMPP* merupakan singkatan dari X (empat sistem operasi apapun), *Apache*, *MySQL*, *PHP* dan *Perl*. Program ini tersedia dalam *GNU General Public License* dan bebas, merupakan *web server* yang mudah digunakan yang dapat melayani tampilan halaman *web* yang dinamis. Untuk mendapatkanya dapat men-*download* langsung dari *web* resminya. (Randi V Palit, 2015 : 2).

II.2.13. PHP

PHP atau kependekan dari *Hypertext Preprocessor* adalah salah satu bahasa pemrograman *open source* yang sangat cocok atau dikhususkan untuk pengembangan *web* dan dapat ditanamkan pada sebuah skripsi HTML. Bahasa PHP dapat dikatakan menggambarkan beberapa bahasa pemrograman seperti C, *Java*, dan *Perl* serta mudah untuk dipelajari. PHP merupakan bahasa *scripting server – side*, dimana pemrosesan datanya dilakukan pada sisi *server*. Sederhananya, *server* yang akan menterjemahkan skrip program, baru kemudian hasilnya akan dikirim kepada *client* yang melakukan permintaan. Adapun pengertian lain PHP adalah *akronim* dari *Hypertext Preprocessor*, yaitu suatu bahasa pemrograman berbasiskan kode – kode *(script)* yang digunakan untuk

mengolah suatu data dan mengirimkannya kembali ke *web browser* menjadi kode HTML". (Astria Firman, 2016 : 30).

II.2.14. *UML* (*Unified Modelling Language*)

Unified Modelling Language (UML) merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek. UML merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan sistem berorientasi objek karena UML menyediakan Bahasa pemodelan visual yang memungkinkan pengembang sistem membuat *blue print* atas visinya dalam bentuk yang baku. UML berfungsi sebagai jembatan dalam mengkomunikasikan beberapa aspek dalam sistem melalui jumlah elemen grafis yang bisa dikombinasikan menjadi.

Unified Modeling Language (UML) biasa digunakan untuk :

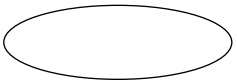
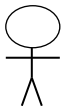
a. Menggambarkan batasan sitem dan fungsi - fungsi sistem secara umum, dibuat dengan *use case* dan *actor*.
b. Menggambarkan kegiatan atau proses bisnis yang dilaksanakan secara umum, dibuat dengan *interaction diagrams*.
c. Menggambarkan representasi struktur *static* sebuah sistem dalam betuk *class diagrams*.
d. Membuat model *behavior* "yang menggambarkan kebiasaan atau sifat sebuah sistem" dengan *state transition diagrams*.
e. Menyatakan arsitektur implementasi fisik menggunakan *component and development.*

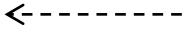
f. Menyampaikan atau memperluas *fungsionality* dengan *stereotypes*. (Ommi Alfina dan Fitriana Harahap : 2019 : 36)

II.2.14.1. *Use Case* Diagram

Use case diagram merupakan pemodelan untuk sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram dapat dilihat pada tabel II.6 dibawah ini :

Tabel II.6. Simbol *Use Case*

Gambar	Keterangan	Deskripsi
	*Use case*	Menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor	Sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem.

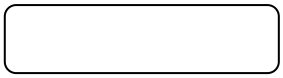
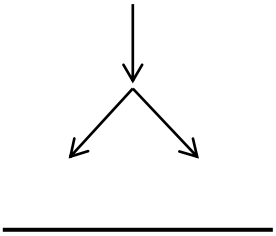
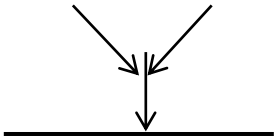
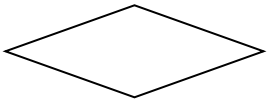
	Asosiasi	Penghubung antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi	Penghubung antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include	Merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	*Extend*	Merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

***(*Sumber : Ade Hendini, 2016)**

II.2.14.2. *Activity* Diagram

Activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada tabel II.7 dibawah ini:

Tabel II.7. Simbol *Activity Diagram*

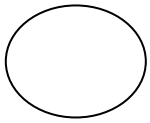
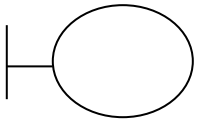
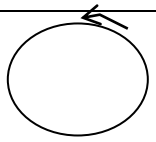
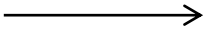
Gambar	Keterangan	Deskripsi
	*Start point*	Diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*	Akhir aktifitas.
	*Activites*	Menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan).	Digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan)	Digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*	Menggambarkan pilihan untuk pengambilan keputusan, *true, false.*
	*Swimlane*	Pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

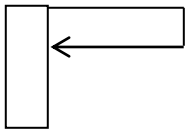
(Sumber : Ade Hendini, 2016 : 109)

II.2.14.3. *Sequence* Diagram

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram* dapat dilihat pada tabel II.8 dibawah ini :

Tabel II.8. Simbol *Sequence Diagram*

Gambar	Keterangan	Deskripsi
	*Entity Class*	Merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*	Berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class*	Suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*	Simbol mengirim pesan antar *class*.

	*Recursive*	Menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation*	Mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline	Garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Ade Hendini, 2016 : 110)

II.2.14.4. *Class* Diagram

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. *Class* diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class* diagram secara khas meliputi : Kelas (*Class*), Relasi *Assosiations*, *Generalitation* dan *Aggregation*, attribut (*Attributes*), operasi (*operation/method*) dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau attribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *Multiplicity* atau *Cardinality* (Ade Hendini, 2016 : 111).

Tabel II.9. *Multiplicity Class Diagram*

Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Ade Hendini, 2016 : 110)