

BAB I

PENDAHULUAN

1.1 Latar Belakang

Sistem monitoring presensi pada tingkat sekolah atau kampus saat ini sudah mulai beralih menggunakan alat elektronik seperti *fingerprint*, *RFID* dan pengenalan wajah. Proses dari sistem presensi menggunakan alat pengenalan wajah yang sudah ada dilakukan dengan menggunakan sebuah mesin pengenalan wajah yang dipasang pada sebuah tempat atau ruangan. Sistem ini masih mempunyai suatu kelemahan, yaitu identifikasi wajah orang hadir harus dilakukan secara bergiliran, tidak bisa dilakukan secara bersamaan sekaligus, sehingga dapat menimbulkan antrian yang panjang.

Presensi kelas di Teknik Informatika Universitas Darma Persada masih menggunakan dua cara yaitu, *namely roll-call* dan *sign in sheet*. *Namely roll-call*, presensi kelas yang dilakukan dengan cara seorang dosen membuka menu Presensi Kelas di sistem akademik UNSADA lalu menyebutkan nama mahasiswa satu persatu agar terdata kehadirannya. *Sign in sheet*, setiap mahasiswa mencari nama mereka lalu menandatangani kertas tersebut agar terdata kehadirannya lalu dilakukan proses input pada sistem akademik UNSADA.

Berdasarkan dua cara presensi kelas yang masih digunakan di Teknik Informatika Universitas Darma Persada tersebut serta adanya mesin absensi pengenalan wajah yang tidak dapat dilakukan secara bersamaan sekaligus, maka menjadi alasan untuk melakukan perancangan “*APLIKASI MULTIPLE FACE RECOGNITION* UNTUK PRESENSI KELAS DI TEKNIK INFORMATIKA

UNIVERSITAS DARMA PERSADA MENGGUNAKAN METODE LBPH (*LOCAL BINARY PATTERNS HISTOGRAMS*)”.

Aplikasi ini berbasis web dan menggunakan Metode LBPH (*Local Binary Patterns Histograms*). Metode LBPH memiliki prinsip kerja mengidentifikasi wajah manusia dengan mengubah citra menjadi matriks 3x3 lalu dibuat diagram histogram setelah itu disimpan di database, diagram histogram citra training akan dibandingkan dengan citra uji untuk mengidentifikasi wajah.

Aplikasi ini nantinya dapat diimplementasikan di jurusan Teknik Informatika Universitas Darma Persada untuk Presensi Kelas agar dapat melakukan presensi kelas menggunakan wajah manusia.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan diatas, maka rumusan masalah dalam pembuatan aplikasi ini adalah sebagai berikut:

1. Bagaimana menerapkan metode LBPH (*Local Binary Patterns Histograms*) untuk melakukan pengenalan banyak wajah yang telah terdeteksi sehingga presensi kelas dapat dilakukan.
2. Bagaimana membuat aplikasi *multiple face recognition* untuk presensi kelas di Teknik Informatika Universitas Darma Persada.

1.3 Batasan Masalah

Dalam perancangan aplikasi ini terdapat beberapa batasan masalah. Hal ini dilakukan agar aplikasi dapat terfokus / sesuai kebutuhan. Batasan masalah tersebut sebagai berikut :

1. Aplikasi ini dirancang hanya untuk presensi kelas terhadap mahasiswa Jurusan Teknik Informatika Universitas Darma Persada yang sudah memiliki foto sebagai data training dan terdaftar pada database.
2. Metode yang digunakan pada aplikasi ini adalah LBPH (*Local Binary Patterns Histograms*).
3. Citra wajah yang diambil untuk data latih (*training*) 50 foto untuk setiap mahasiswa dengan posisi 5° menghadap ke kanan, 5° menghadap ke kiri, 5° menghadap ke atas, 5° menghadap ke bawah, serta menggunakan satu ekspresi yaitu senyum.
4. Citra wajah yang diambil untuk data latih (*training*) dilakukan di dalam ruangan dengan menggunakan cahaya lampu ruang pada jarak 50cm - 100cm.
5. Citra wajah yang diambil sebagai data uji harus sesuai dengan citra wajah yang digunakan sebagai data *training* dan tidak menggunakan pas foto..

1.4 Tujuan dan Manfaat

1.4.1 Tujuan

Tujuan dari perancangan aplikasi ini adalah untuk merancang aplikasi *multiple face recognition* yang dapat digunakan untuk proses presensi kelas mahasiswa Teknik Informatika Universitas Darma Persada dengan menggunakan metode LBPH (*Local Binary Patterns Histograms*).

1.4.2 Manfaat

Manfaat yang didapat dari penelitian ini adalah :

1. Hasil penulisan Laporan Tugas Akhir ini dapat menjadi tambahan referensi untuk penulisan dan penelitian selanjutnya tentang presensi kelas.

2. Hasil penelitian ini diharapkan dapat menjadi alternatif pengganti sistem sebelumnya untuk presensi kelas di Teknik Informatika Universitas Darma Persada.

1.5 Metodologi Penulisan

Metodologi yang digunakan dalam penulisan laporan kerja praktek ini sebagai berikut:

1. Observasi

Observasi adalah suatu cara pengumpulan data dengan melakukan pengamatan secara langsung dan melakukan pencatatan secara sistematis terhadap objek yang akan diteliti. Observasi dilakukan oleh peneliti dengan cara melakukan pengamatan dan pencatatan terhadap aktivitas pelaksanaan kegiatan yang terkait dengan presensi kelas di Teknik Informatika Universitas Darma Persada.

2. Wawancara (*Interview*)

Wawancara dilakukan dengan mahasiswa serta dosen yang terlibat langsung dengan kegiatan presensi kelas di Teknik Informatika Universitas Darma Persada. Wawancara dalam penelitian ini dilakukan untuk mendapatkan data dan informasi yang lebih lengkap.

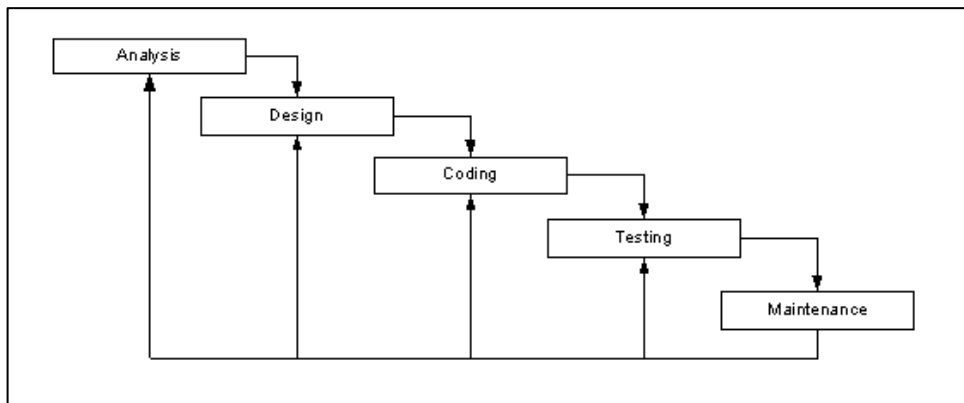
3. Studi Kepustakaan

Studi pustaka merupakan metode pengumpulan data yang diarahkan kepada pencarian data dan informasi melalui dokumen-dokumen, baik dokumen tertulis, foto-foto, gambar, maupun dokumen elektronik yang dapat mendukung dalam proses penelitian. Studi pustaka dilakukan oleh penulis dengan cara mempelajari teori-teori dari sumber bacaan dan

literatur yang berhubungan dengan objek penelitian sebagai bahan referensi dalam penulisan ini.

1.6 Metodologi Pengembangan Sistem

Dalam pengembangan aplikasi *multiple face recognition* untuk presensi kelas di Teknik Informatika Universitas Darma Persada menggunakan metodologi *waterfall*. Metodologi *waterfall* adalah suatu proses pengembangan perangkat lunak berurutan, di mana kemajuan dipandang sebagai terus mengalir ke bawah (seperti air terjun) melewati fase-fase perencanaan, pemodelan, implementasi (konstruksi), dan pengujian.



Gambar 1.1. Metodologi *Waterfall* Menurut Pressman, Sommerville (2010)

Adapun penjelasan urutan dari tahapan-tahapan yang dimiliki metodologi *waterfall* adalah sebagai berikut:

1. *Analysis*

Dalam fase ini penulis melakukan analisa kebutuhan, seperti mengumpulkan data-data yang dibutuhkan sebagai bahan untuk melakukan pembuatan aplikasi.

2. *Design*

Dalam fase ini penulis membuat tampilan-tampilan layout sistem yang akan dibangun dalam aplikasi.

3. *Coding*

Dalam fase ini penulis melakukan pembuatan aplikasi dengan menggunakan kode-kode program yang sesuai dengan tujuan awal yaitu dengan pemrograman *web*.

4. *Testing*

Dalam fase ini penulis melakukan pengujian apakah setelah dikerjakan ada kesalahan atau tidak.

5. *Maintenance*

Dalam fase ini adalah dengan melakukan pengembangan dan pemeliharaan terhadap aplikasi, apakah nantinya ada kesalahan atau tidak.

1.7 Sistematika Penulisan

Pada penulisan skripsi ini, akan dipergunakan sistematika penulisan sebagai berikut :

BAB I : PENDAHULUAN

Bagian ini berisikan informasi mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan dan manfaat, metode yang digunakan dan sistematika penulisan.

BAB II : LANDASAN TEORI

Bab ini menjelaskan tentang teori-teori yang berhubungan dengan penulisan laporan tugas akhir, yaitu Pengolahan Citra, LBPH

(Local Binary Patterns Histograms), UML (*Unified Modeling Language*) dan lain sebagainya.

BAB III : DESAIN DAN PERANCANGAN SISTEM

Bagian ini berisikan tentang data-data yang dibutuhkan dalam perancangan suatu sistem yang terdiri dari UML, desain-desain struktur database, serta desain tampilan untuk aplikasi web.

BAB IV : IMPLEMENTASI SISTEM DAN ANALISIS SISTEM

Pada bab ini berisi tentang implementasi program yang telah dihasilkan, gambaran umum sistem dan evaluasi mengenai sistem yang telah dirancang dan dibuat.

BAB V : PENUTUP

Bagian ini berisi mengenai kesimpulan yang dapat diambil dari penyusunan tugas akhir, serta saran-saran penulis yang diharapkan dapat bermanfaat bagi pihak-pihak lain yang berkepentingan.

BAB II

LANDASAN TEORI

2.1 Penelitian Sebelumnya

Absensi Kehadiran Mahasiswa di Kelas Secara Real - Time Berbasis Multi Wajah Menggunakan Metode Eigenface [2017]. Oleh Yudha Pratama (Program Studi Sistem Komputer Institut Bisnis dan Informatika Stikom Surabaya). Pada Penelitian ini dibuat suatu sistem absensi kehadiran mahasiswa di kelas berbasis multi wajah menggunakan webcam HD (resolusi 1280x720) sebagai media input utama, pembuatan program menggunakan Microsoft Visual Studio. Sistem absensi multi wajah dapat mendeteksi dan mengenali objek wajah apabila objek wajah masuk dalam jangkauan kamera webcam dan dalam rentang jarak objek wajah terhadap kamera webcam 15 cm sampai dengan 250 cm. Akurasi ada sistem absensi berbasis multi wajah mendapatkan 91% terhadap tingkat pengenalan wajah.

Perbandingan Akurasi Pengenalan Wajah Menggunakan Metode LBPH dan Eigenface dalam Mengenali Tiga Wajah Sekaligus secara Real-Time [2016]. Oleh Harris Simaremare dan Agung Kurniawan (Program Studi Teknik Elektro Fakultas Sains dan Teknologi UIN Sultan Syarif Kasim Riau). Aplikasi pengenalan wajah semestinya mampu mengenali banyak wajah sekaligus ketika tertangkap oleh kamera. Oleh karena itu peneliti akan menguji tingkat akurasi dari kedua metode ini untuk mengenali tiga wajah sekaligus. Pengujian dilakukan pada 300 sampel citra wajah dengan empat kondisi pencahayaan yaitu siang hari dalam ruangan dan siang hari diluar ruangan. Hasil pengujian menunjukkan bahwa

tingkat akurasi LBPH lebih baik dibandingkan eigenface dengan rata-rata akurasi LBPH adalah 93.54 % dan eigenface adalah 63.54%.

LBPH Based Improved Face Recognition At Low Resolution [2018]. Oleh Aftab Ahmed, Jiandong Guo, dll (School of Information & Software Engineering, University of Electronic Science & Technology of China). Pada penelitian ini ditujukan untuk membandingkan. Pada penelitian ini dilakukan pengenalan wajah menggunakan algoritma LBPH dengan menggunakan citra latih sebanyak 2500 sehingga didapatkan akurasi sebesar 94% untuk citra wajah yang berukuran 45px dan 90% untuk citra wajah yang berukuran 35px.

2.2 Presensi Kelas

Menurut KBBI (Kamus Besar Bahasa Indonesia) definisi dari presensi adalah kehadiran. Di Teknik Informatika Universitas Darma Persada kehadiran mahasiswa di kelas disebut dengan istilah presensi kelas. Pengertian presensi kelas mengandung dua arti, yaitu masalah kehadiran di kelas (*school attendance*) dan ketidakhadiran di kelas (*non school attendance*). Kehadiran mahasiswa di kelas (*school attendance*) adalah kehadiran dan keikutsertaan mahasiswa secara fisik dan mental terhadap aktivitas belajar pada jam-jam efektif di kampus. Sedangkan ketidakhadiran adalah ketiadaan partisipasi secara fisik mahasiswa terhadap kegiatan belajar di kampus.

2.3 Pengolahan Citra Digital

Menurut Rifkie Primartha, 2018 dalam buku “Belajar *Machine Learning* Teori dan Praktik”. Kecerdasan buatan adalah studi tentang teori dan pengembangan sistem komputer agar mampu melakukan tugas-tugas yang dahulu

hanya dapat dilakukan oleh manusia. Seperti membedakan berbagai gambar, menjawab pertanyaan, mengenali dan menerjemahkan bahasa, dan sebagainya.

Menurut Zulfian Azmi dan Verdi Yasin, 2017 dalam buku “Pengantar Sistem Pakar dan Metode (*Introduction of Expert System and Methods*)”. *Computer Vision* adalah salah satu cabang ilmu pengetahuan yang mempelajari bagaimana komputer dapat mengenali objek yang diamati atau diobservasi. *Computer vision* merupakan salah satu sub disiplin ilmu dalam kecerdasan buatan.

Menurut Abdul Kadir dan Adhi Susanto, 2013 dalam buku “Teori dan Aplikasi Pengolahan Citra”. Secara umum istilah pengolahan citra digital menyatakan “pemrosesan gambar berdimensi-dua melalui komputer digital”, pengolahan citra adalah istilah umum untuk berbagai teknik yang keberadaannya untuk memanipulasi dan memodifikasi citra dengan berbagai cara. Foto adalah contoh gambar berdimensi dua yang dapat diolah dengan mudah. setiap foto dalam bentuk citra digital (misalnya berasal dari kamera digital) dapat diolah melalui perangkat lunak tertentu.

Pada *machine vision* (sistem yang dapat “melihat” dan “memahami” yang dilihatnya), pengolahan citra berperan untuk mengenali bentuk-bentuk khusus yang dilihat oleh mesin. Penggunaan kamera pemantau ruangan merupakan contoh bagian aplikasi pemrosesan citra. Citra adalah suatu gambaran atau kemiripan dari suatu objek. Citra digital adalah citra yang dapat diolah oleh komputer.

2.4 Pemrograman Aplikasi

2.4.1 HTML

Menurut Heru Sulistiono, 2018 dalam buku “Coding mudah dengan Codeigniter, JQuery, Bootstrap, dan Datatable”. HTML (*Hypertext Markup Language*) adalah sebuah bahasa markah yang digunakan untuk membuat sebuah halaman web, menampilkan berbagai informasi seperti gambar, teks, video dan suara pada penjelajah web internet, yang ditulis dalam berkas format ASCII agar dapat menghasilkan tampilan wujud yang terintegrasi.

2.4.2 CSS

Menurut Heru Sulistiono, 2018 dalam buku “Coding mudah dengan Codeigniter, JQuery, Bootstrap, dan Datatable”. CSS (*Cascading Style Sheet*) merupakan aturan untuk mengendalikan beberapa komponen dalam sebuah web sehingga akan lebih terstruktur dan seragam. CSS bukan merupakan bahasa pemrograman. Pada umumnya CSS dipakai untuk memformat tampilan halaman web yang dibuat dengan bahasa HTML dan XHTML.

CSS dapat mengendalikan ukuran gambar, warna bagian tubuh pada teks, warna tabel, ukuran border, warna border, warna *hyperlink*, warna mouse over, spasi antar paragraf, spasi antar teks, margin, dan parameter lainnya. CSS adalah bahasa style sheet yang digunakan untuk mengatur tampilan dokumen. Dengan adanya CSS memungkinkan kita untuk menampilkan halaman yang samab dengan format yang berbeda.

2.4.3 JavaScript

Menurut Jubilee Enterprise, 2017 dalam buku “Otodidak Pemrograman JavaScript”. Pemrograman dasar untuk pembuatan website adalah HTML. Namun

HTML hanya bisa menghasilkan dokumen statis dan non-interaktif. Javascript diciptakan untuk mengatasi kelemahan ini. Apabila ingin membuat situs yang dapat berinteraksi antara user dengan website secara cepat tanpa harus melibatkan web server, maka digunakanlah Javascript. Tanpa koneksi internet sekalipun, data yang diinput oleh user dapat langsung diproses. Hal ini terjadi karena Javascript bersifat *client-side*. Javascript dapat digunakan untuk berbagai keperluan. Bahkan HTML5 baru bisa menghasilkan output optimal apabila Anda melibatkan Javascript.

2.4.4 PHP

Menurut Heru Sulistiono, 2018 dalam buku “Coding mudah dengan Codeigniter, JQuery, Bootstrap, dan Datatable”. PHP (*Hypertext Preprocessor*) adalah bahasa pemrograman yang digunakan untuk membuat website atau situs dinamis dan menangani rangkaian bahasa pemrograman antara *client side scripting* dan *server side scripting*. Berbeda dengan HTML yang source kodenya di tampilkan di website, source code PHP tidak ditampilkan di halaman muka suatu website karena PHP diolah dan diproses di server, PHP bersifat server-side scripting yang mampu berjalan di berbagai system operasi seperti windows, Linux, Mac OS, dll.

2.4.5 Bootstrap

Menurut Jubilee Enterprise, 2016 dalam buku “Pemrograman Bootstrap untuk Pemula”. Bootstrap adalah framework front-end yang intuitif dan powerful untuk pengembangan aplikasi web yang lebih cepat dan mudah. Bootstrap menggunakan HTML, CSS dan Javascript. Bootstrap dikembangkan oleh Mark

Otto dan Jacob Thornton dari twitter Framework ini diluncurkan sebagai produk open source pada Agustus 2011 di GitHub.

Bootstrap memiliki fitur-fitur komponen interface yang bagus seperti Typography, Form, Buttons, Tables, Navigations, Dropdowns, Alerts, Modals, Tabs, Accordion, Carousel dan lain sebagainya. Dengan menggunakan Bootstrap, Anda dapat membuat layout situs yang responsif dengan mudah.

2.4.6 AJAX

Menurut Kadir , 2011 dalam buku “Membangun Web Responsive dengan AJAX”. AJAX (Asynchronous JavaScript and XML) adalah suatu teknik yang memungkinkan untuk membuat aplikasi web interaktif (mirip dengan fitur pada aplikasi desktop). AJAX atau Asynchronous JavaScript and XML merupakan kumpulan teknologi di mana klien berinteraksi dengan server dengan menggunakan JavaScript dengan proses asinkron (background) secara periodik, di mana pengiriman data menggunakan XML. AJAX biasanya digunakan untuk terus menampilkan update terbaru dari halaman web. Menggunakan AJAX berarti proses interaksi dengan server dilakukan terus-menerus selama halaman web dibuka secara periodik, hal ini menyebabkan memakan bandwidth yang cukup besar.

Mekanisme kerja AJAX adalah sebagai berikut :

1. Klien mengirimkan request ke server.
2. Server melakukan proses pengambilan data yang diperlukan guna memenuhi permintaan klien.
3. Server mengirimkan data ke klien dengan menggunakan XML.

4. Klien menerima data dan memperbarui tampilan halaman web sesuai dengan data yang dikirimkan oleh server.

2.4.7 JSON

Menurut Egi Aditya, 2013 dalam jurnal Web Service, UNIKOM . JSON (JavaScript Object Notation) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (generate) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemrograman JavaScript, Standar ECMA-262 Edisi ke-3 – Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran data.

2.4.8 Python

Menurut Abdul Kadir, 2018 dalam buku “ Dasar Pemrograman Python 3. Panduan untuk Mempelajari Python dengan Cepat dan Mudah bagi Pemula”. Python merupakan bahasa pemrograman beraras tinggi yang diciptakan oleh Guido van Rossum pada tahun 1989 di Amsterdam, Belanda. Sebagai bahasa beraras tinggi, Python menawarkan berbagai kemudahan dalam menulis suatu program. Sebagai bahasa yang multiplatform, yang dapat berjalan dalam lingkungan seperti Windows, UNIX, Linux, dan MAC, Python memberikan portabilitas yang tinggi. Bahkan, saat ini Python juga digunakan untuk memprogram *hardware* di Raspberry Pi, sebuah komputer seukuran kartu kredit.

Versi pertama Python sebenarnya baru dipublikasi pada tahun 1991. Sebelum tahun 2008, versi Python yang beredar adalah Python 2. Pada Desember

2008, versi Python 3.0 mulai dirilis. Versi ini tidak bersifat kompetibel dengan Python 2.

2.5 Basis Data dan PostgreSQL

2.5.1 Basis Data

Menurut Dewi Kusumawati, 2015 dalam buku “Basis Data dengan PostgreSQL”. Basis data atau juga sering disebut sebagai database adalah kumpulan dari item data yang saling berhubungan satu dengan yang lainnya yang diorganisasikan berdasarkan sebuah skema atau struktur tertentu, tersimpan di hardware komputer dan dengan software untuk melakukan manipulasi untuk kegunaan tertentu.

Database juga dapat disebut dengan jantung sebuah sistem informasi atau komponen yang sangat penting dalam sistem informasi. *Database* bisa dipakai untuk menentukan kualitas informasi yang dinilai dari keakuratan, relevan dan tepat pada waktunya. *Database* juga diperlukan untuk mengurangi adanya duplikasi data (*data redundancy*), meningkatkan hubungan data dan mengurangi pemborosan tempat simpanan di luar.

Dalam *database* terdapat tingkatan-tingkatan, tingkatan tersebut antara lain:

1. *Database*. Kumpulan dari file/tabel membentuk suatu database.
2. File. File terdiri dari record-record yang menggambarkan satu kesatuan data yang sejenis. Misalnya file buku berisi data tentang semua data-data buku yang ada.
3. Record. Kumpulan dari field membentuk suatu record. Record menggambarkan suatu unit data individu yang tertentu.

4. Field. Merepresentasikan suatu atribut dari record yang menunjukkan suatu item dari data, seperti misalnya nama, alamat dan lain sebagainya. Kumpulan dari field membentuk suatu record.
5. Characters. Merupakan bagian data yang terkecil, dapat berupa karakter numerik, huruf ataupun karakter-karakter khusus (*special characters*) yang membentuk suatu item data/field.

2.5.2 PostgreSQL

Menurut Dewi Kusumawati, 2015 dalam buku “Basis Data dengan PostgreSQL”. PostgreSQL atau sering disebut postgres merupakan salah satu dari sejumlah *database* besar yang menawarkan skalabilitas, keluwesan, dan kinerja yang tinggi. Penggunaannya begitu meluas di berbagai *platform* dan didukung oleh banyak bahasa pemrograman.

PostgreSQL pertama kali ada pada tahun 1996. PostgreSQL merupakan *database* server yang bersifat *open source*, memiliki lisensi GPL (*General Public License*) dan merupakan salah satu dari sejumlah *database server*.

2.6 Algoritma Sistem

Menurut Michael Yoseph, 2009 dalam buku “Pengenal Computer Vision Menggunakan OpenCV dan FLTK”. OpenCV adalah sebuah *library* bebas yang awalnya dibangun oleh intel. *Library* ini terdiri dari fungsi-fungsi *computer vision* dan API (*Application Programming Interface*) untuk *image processing* *high level* maupun *low level* sebagai optimasi aplikasi *realtime*. OpenCV mampu menciptakan aplikasi yang handal dan mempunyai kemampuan yang mirip dengan kemampuan manusia. *Library* ini dapat digunakan di *platform* mana saja,

termasuk Windows, Linux, Mac OS, dan lain-lain. OpenCV difokuskan untuk memproses gambar yang berjalan secara langsung (*real-time*).

Pengaplikasian OpenCV mencakup :

1. *2D and 3D feature toolkits*
2. *Ego-motion*
3. *Face Recognition*
4. *Gesture Recognition*
5. *Human-Computer Interface (HCI)*
6. *Mobile robotics*
7. *Motion Understanding*
8. *Object Identification*
9. *Segmentation and Recognition*

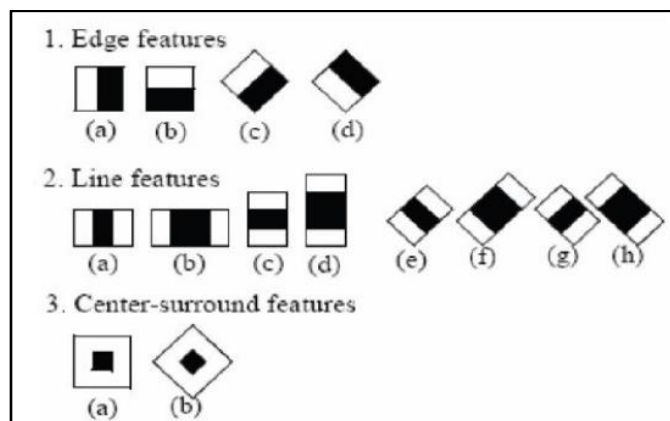
2.6.1 Haar Cascade Classifier

Menurut RD. Kusumanto, 2012 dalam jurnal “Aplikasi Sensor Vision untuk Deteksi MultiFace dan Menghitung Jumlah Orang”. Metode Haar menggunakan metode statistical dalam melakukan deteksi wajah. Metode ini menggunakan *sample Haarlike Fetures*. *Classifier* ini menggunakan gambar berukuran tetap (umumnya berukuran 24x24). Cara kerja dari *Haar* dalam mendeteksi wajah adalah dengan menggunakan teknik *sliding window* berukuran 24x24 pada keseluruhan gambar dan mencari apakah terdapat bagian dari gambar yang berbentuk seperti wajah atau tidak. Haar juga memiliki kemampuan untuk melakukan *scaling* sehingga dapat mendeteksi adanya wajah yang berukuran lebih besar ataupun lebih kecil dari gambar pada *Classifier*.

2.6.1.1 Haar Image

Menurut M. Dwisnanto, Putro, dkk, 2012 dalam jurnal “Sistem Deteksi Wajah dengan menggunakan Metode Viola-Jones”. Gambar (*image, picture*) adalah kombinasi antara titik, garis, bidang, dan warna untuk menciptakan suatu imitasi atau tiruan dari suatu objek fisik atau barang (manusia, binatang, tumbuhan, dan lain sebagainya)

Menurut RD. Kusumanto, 2012 dalam jurnal “Aplikasi Sensor Vision untuk Deteksi MultiFace dan Menghitung Jumlah Orang”. Proses yang dilakukan dalam metode ini antara lain ada 3, yaitu *haar feature*, *integral image* dan *cascade classifier*. *Haar Feature* adalah fitur yang didasarkan pada *Wavelet Haar*. *Wavelet Haar* adalah gelombang tunggal bujur sangkar (satu interval tinggi dan satu interval rendah). Untuk dua dimensi, satu terang dan satu gelap. Setiap *Haar-like feature* terdiri dari gabungan kotak-kotak hitam dan putih.



Gambar 2.1. Macam-macam Variasi *Feature* pada *Haar* menurut RD. Kusumanto, dkk (2012)

Adanya fitur *Haar* ditentukan dengan cara mengurangi rata-rata piksel pada daerah gelap dari rata-rata piksel pada daerah terang. Jika nilai perbedaannya itu di atas nilai ambang atau *threshold*, maka dapat dikatakan bahwa fitur tersebut

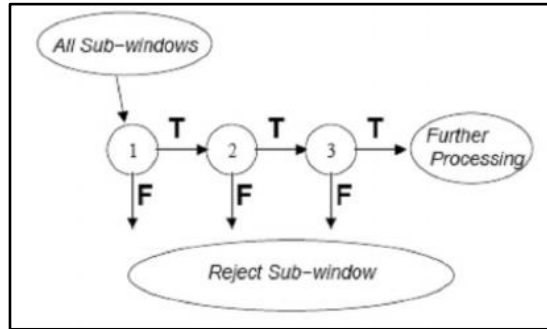
ada. Selanjutnya untuk menentukan ada atau tidaknya dari ratusan fitur *Haar* pada sebuah gambar dan pada skala yang berbeda secara efisien digunakan citra Integral.

2.6.1.2 *Integral Image*

Integral image digunakan untuk menentukan ada atau tidaknya dari ratusan fitur *Haar* pada sebuah gambar dan pada skala yang berbeda secara efisien. Pada umumnya, pengintegrasian tersebut berarti menambahkan unit-unit kecil secara bersamaan. Dalam hal ini unit-unit kecil tersebut adalah nilai-nilai piksel. Nilai integral untuk masing-masing piksel adalah jumlah dari semua piksel-piksel dari atas sampai bawah. Dimulai dari kiri atas sampai kanan bawah, keseluruhan gambar itu dapat dijumlahkan dengan beberapa operasi bilangan bulat per piksel.

2.6.1.3 *Cascade Classifier*

Cascade Classifier adalah sebuah rantai *stage classifier* dimana setiap *stage classifier* digunakan untuk mendeteksi apakah dalam gambar sub window terdapat objek yang diinginkan. *Stage classifier* dibangun dengan menggunakan algoritma *adaptive-boost* (*AdaBoost*). Algoritma tersebut mengkombinasikan *performance* banyak *weak classifier* untuk menghasilkan *strong classifier*. *Weak classifier* dalam hal ini adalah nilai dari *haar-like feature*. Jenis *AdaBoost* yang digunakan adalah *Gentle AdaBoost*.



Gambar 2.2. Model Classifier secara *cascade* menurut RD. Kusumanto, dkk
(2012)

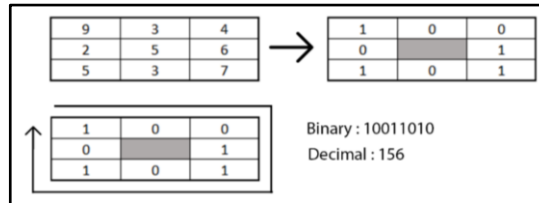
2.6.2 *Local Binary Patterns Histograms (LBPH)*

Menurut Septian Adi Wijaya, 2014 dalam jurnal “Perbandingan Metode Pengenalan Wajah Secara Real Time pada Perangkat Bergerak Berbasis Android”. *Local Binary Patterns Histograms (LBPH)* adalah fitur untuk mengklasifikasi yang dikombinasikan dengan histogram dan merupakan teknik baru dari metode LBP untuk mengubah performa hasil pengenalan wajah. LBP pada umumnya didesain untuk pengenalan tekstur. Metode LBPH ini langkahnya sama dengan original-LBP.

Langkah pertama dari LBPH adalah mencari objek wajah pada suatu gambar lalu dilakukan proses grayscale. Selanjutnya terdapat proses ekstraksi fitur, dimana metode LBPH dilakukan. LBP digunakan untuk mendapatkan tekstur dan bentuk dari suatu citra digital dengan cara menghitung perbandingan lebih besar atau kecil setiap piksel dengan piksel ketetanggaannya. Fitur yang didapat di setiap piksel berupa pola biner yang mewakili piksel tersebut. Setiap pola biner yang didapat kemudian dikonversi menjadi angka desimal kemudian disatukan untuk membentuk fitur histogram yang merupakan representasi dari citra digital tersebut. LBP dapat diformulasikan sebagai berikut :

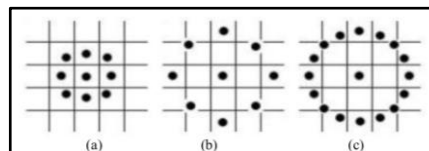
$$LBP_{P,R}(xc, yc) = \sum_{P=0}^{P=x-1} s(gp - gc)2^P$$

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$



Gambar 2.3. Original LBP operator menurut Septian Adi Wijaya (2014)

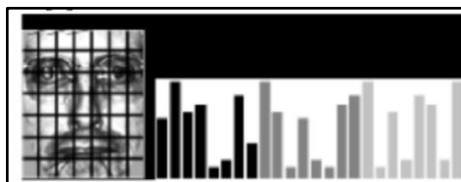
Pada metode ini melakukan pendekatan dengan metode pengembangan *extended-LBP* atau *multi-scale-LBP* yaitu dengan melakukan pembesaran nilai terhadap nilai radius dan tetangga *pixel*. Tidak seperti *original-LBP*, pada *extended-LBP* ini menggunakan nilai *sampling point* dan nilai radius yang berbeda. Secara umum operator dinotasikan nilai P menunjukkan besar *sampling point* dan R menunjukkan besar radius. Seperti yang terlihat pada gambar 2 $LBP_{8,1}$ artinya dengan nilai radius 1 tetangga (*sampling point*) dengan nilai 8. $LBP_{16,2}$ dengan nilai radius 2 dan nilai tetangga 16, pada daerah lingkaran *pixel*.



Gambar 2.4. Multiscale LBP: (a) P=8 R=1 (b) P=8 R=2 (c) P=16 R=2 menurut Septian Adi Wijaya (2014)

Dengan P adalah nilai *sampling point* yang digunakan dan R adalah besar radius. *gp* adalah besar nilai *pixel* tetangga, *gc* besar nilai tengah *pixel* pada *sampling point*. Pada penghitungan ini dilakukan *replacing pixel* atau penggantian *pixel* dengan nilai nol dan satu seperti pada gambar 2.3. Untuk pengklasifikasian citra dengan menghitung perbedaan jarak terkecil antar histogram dari gambar

yang dihasilkan. Citra dilakukan pembagian daerah pada hasil *cropping* menjadi beberapa bagian, pada penelitian ini digunakan konfigurasi pembagian citra menjadi 8x8 seperti pada gambar 2.4.



Gambar 2.5. Sub-region citra dan histogram menurut Septian Adi Wijaya (2014)

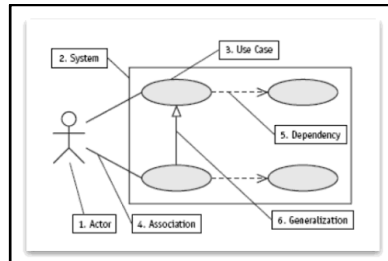
2.7 UML (*Unified Modeling Language*)

Menurut Sri Mulyani, 2016 dalam buku “Analisis dan Perancangan Sistem Informasi Manajemen Keuangan Daerah: Notasi Pemodelan Unified Modeling Language (UML)”. UML (*Unified Modelling Language*) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek. UML adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO).

2.7.1 *Use Case Diagram*

Menurut Sri Mulyani, 2016 dalam buku “Analisis dan Perancangan Sistem Informasi Manajemen Keuangan Daerah: Notasi Pemodelan Unified Modeling Language (UML)”. Use Case Diagram, yaitu diagram yang digunakan untuk menggambarkan hubungan antara sistem dengan aktor. Diagram ini hanya menggambarkan secara global. Diagram use case dapat digunakan selama proses analisis untuk menangkap requirements sistem dan untuk memahami bagaimana sistem seharusnya bekerja.

Karena *use case* diagram hanya menggambarkan sistem secara global, maka elemen-elemen yang digunakan pun sangat sedikit, berikut ini elemen-elemen yang digunakan pada *use case* diagram.



Gambar 2.6. Elemen dari *Use Case* Diagram menurut Sri Mulyani (2016)

Sistem, merupakan batasan-batasan proses yang sudah kita deskripsikan dalam sebuah sistem.

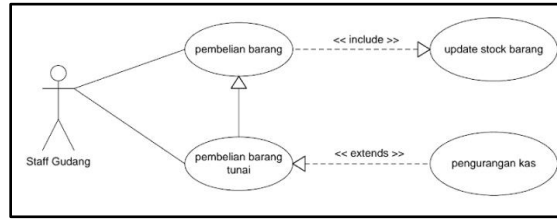
Aktor, elemen yang menjadi pemicu sistem. Aktor bisa berupa orang, mesin ataupun sistem lain yang berinteraksi dengan *use case*.

Use Case, potongan proses yang merupakan bagian dari sistem.

Association, menggambarkan interaksi antara *use case* dan aktor.

Dependency, menggambarkan relasi (*relationship*) antara dua *use case*. Ada 2 (dua) tipe dari *dependency* yaitu, *include* dan *extends*. *Include* merupakan tipe dari *dependency* yang menghubungkan dua *use case* dimana, satu *use case* membutuhkan *use case* yang satunya sedangkan *extends* adalah tipe dari *dependency* yang menghubungkan dua *use case* dimana satu *use case* terkadang akan memanggil *use case* yang satunya, tergantung pada kondisi.

Generalization, menggambarkan pewarisan antara dua aktor atau *use case* dimana salah satu aktor atau *use case* mewarisi properties ke aktor atau *use case* yang satunya.



Gambar 2.7. Contoh dari *Use Case Diagram* menurut Sri Mulyani (2016)

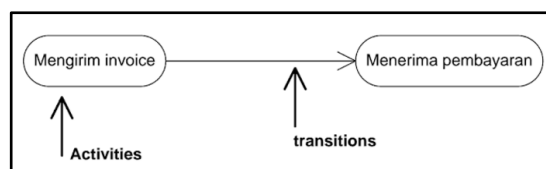
2.7.2 Activity Diagram

Menurut Sri Mulyani, 2016 dalam buku “Analisis dan Perancangan Sistem Informasi Manajemen Keuangan Daerah: Notasi Pemodelan *Unified Modeling Language* (UML)”. *Activity* diagram, yaitu diagram yang digunakan untuk menggambarkan alur kerja (aktivitas) pada *use case* (proses), logika, proses bisnis dan hubungan antara aktor dengan alur-alur kerja *use case*. Diagram ini sangat mirip dengan sebuah *flowchart* karena dapat dimodelkan sebuah alur kerja dari satu aktivitas ke aktivitas lainnya atau dari satu aktivitas ke dalam keadaan sesaat (*state*). Diagram aktivitas paling cocok digunakan untuk memodelkan urutan aktivitas dalam suatu proses. Berikut ini dijalankan elemen-elemen dari *activity* diagram:

Activities, yaitu elemen yang digunakan untuk menggambarkan aktivitas.

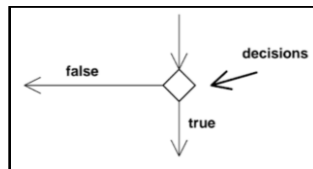
Transitions, yaitu elemen yang digunakan untuk menggambarkan transisi dari elemen yang satu ke elemen yang lainnya.

Berikut ini adalah contoh gambar dari *activities* dan *transitions*



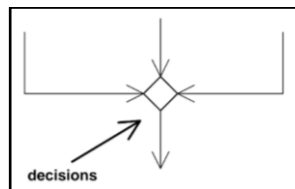
Gambar 2.8. Contoh *Activities* dan *Transition* menurut Sri Mulyani (2016)

Decisions, yaitu elemen yang digunakan untuk percabangan logika. Elemen ini sering kita jumpai pada *flowchart* terutama *flowchart* yang digunakan untuk menggambarkan sebuah algoritma. Berikut ini contoh gambar dari elemen *decisions*.



Gambar 2.9. Contoh Elemen *Decision* menurut Sri Mulyani (2016)

Merge Point, yaitu elemen yang digunakan untuk menggabungkan percabangan proses. Elemen ini merupakan kebalikan dari elemen *decisions*, dimana jika *decisions* digunakan untuk percabangan, sedangkan *merge point* digunakan sebagai penggabungan dari percabangan. Berikut ini contoh gambar dari elemen *merge point*.

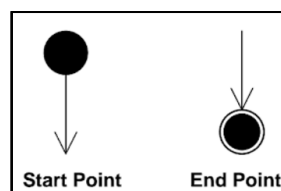


Gambar 2.10. Contoh Elemen *Merge Point* menurut Sri Mulyani (2016)

Start Point, yaitu elemen yang digunakan untuk memulai *activity* diagram

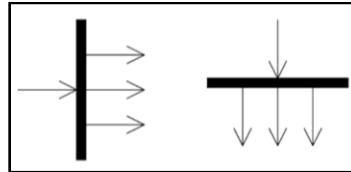
End Point, yaitu elemen yang digunakan untuk mengakhiri *activity* diagram.

Berikut ini contoh gambar dari *start* dan *end point*



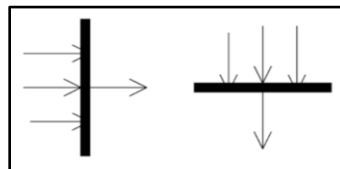
Gambar 2.11. Contoh *Start* dan *End Point* menurut Sri Mulyani (2016)

Concurrency, yaitu elemen yang digunakan sebagai percabangan proses (bukan percabangan logika). Proses yang ada di dalam elemen ini, bisa dilakukan secara *random* (tidak berurutan). Berikut ini contoh gambar dari *concurrency*.



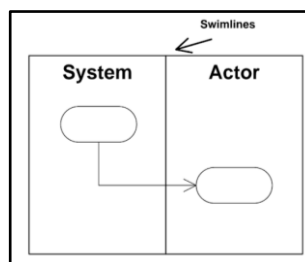
Gambar 2.12. Contoh *Concurrency* menurut Sri Mulyani (2016)

Synchronization, yaitu elemen yang digunakan untuk menggabungkan proses yang dipisahkan oleh *concurrency*. Berikut ini contoh gambar dari *synchronization*.



Gambar 2.13. Contoh *Synchronization* menurut Sri Mulyani (2016)

Swimlanes, yaitu elemen yang digunakan untuk memisahkan antara aktor dan sistem ataupun antara aktor yang satu dengan aktor yang lain atau antara sistem yang satu dengan sistem yang lain.



Gambar 2.14. Contoh *Swimlanes* menurut Sri Mulyani (2016)

Sinyal, yaitu acuan waktu yang bisa dijadikan trigger (pemicu) untuk aktivitas tertentu, misalnya setiap akhir jam kerja seluruh *staff* wajib memberikan laporan kepada manajer. Setiap akhir jam kerja bisa disimbolkan dengan sinyal waktu.








Gambar 2.15. Contoh Sinyal menurut Sri Mulyani (2016)

2.7.3 Sequence Diagram

Menurut (A.S Rosa dan Shalahuddin M, 2011) dalam Buku “Modul Pembelajaran Rekayasa Perangkat Lunak”. *Sequence Diagram* adalah diagram yang menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu, untuk menggambar diagram *sequence* harus diketahui objek-objek yang terlibat dalam sebuah *use case*.

Tabel 2.1. Simbol-simbol *Sequence Diagram* menurut A.S Rosa dan Shalahuddin M (2011)

| No | SIMBOL | KETERANGAN |
|----|---|--|
| 1 |  | Menyatakan objek yang berinteraksi pesan dan menyatakan kehidupan suatu objek. |
| 2. |  | Menyatakan objek dalam keadaan aktif dan berinteraksi pesan. |

| | | |
|----|---|--|
| 3. |  | Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri. |
| 4. |  | Menyatakan bahwa suatu objek mengirimkan data ke objek lainnya. |
| 5. |  | Menyatakan bahwa suatu objek yang telah menjalankan menghasilkan suatu hasil perintah ke objek tertentu. |

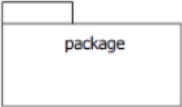
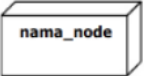


2.7.4 Deployment Diagram

Deployment diagram menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi. *Deployment* diagram juga dapat digunakan untuk memodelkan hal-hal berikut :

1. Sistem tambahan (*embedded system*) yang menggambarkan rancangan *device*, *node* dan *hardware*.
2. Sistem *client/server*
3. Sistem terdistribusi murni
4. Rekayasa ulang aplikasi

Berikut adalah simbol-simbol yang ada pada *deployment diagram* :

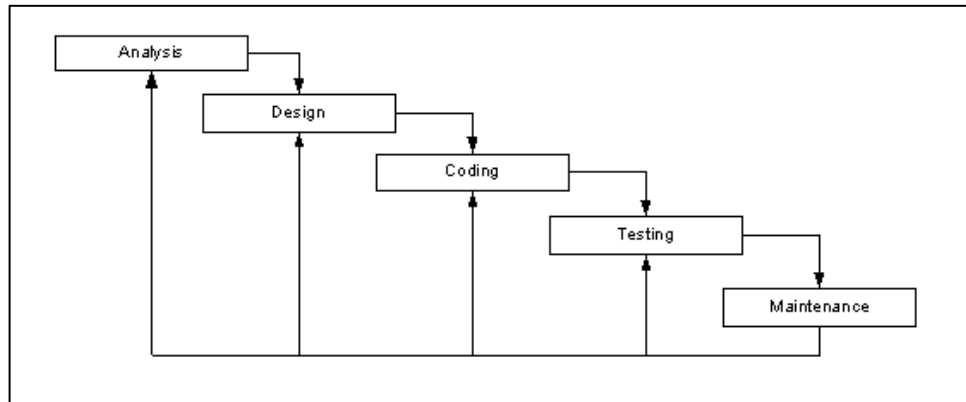
Tabel 2.2. Simbol-simbol *Deployment* Diagram (Rosa A.S dan M. Shalahuddin, 2016)

| Simbol | Deskripsi |
|---|---|
| Package  | package merupakan sebuah bungkusan dari satu atau lebih <i>node</i> |
| Node  | biasanya mengacu pada perangkat keras (<i>hardware</i>), perangkat lunak yang tidak dibuat sendiri (<i>software</i>), jika di dalam <i>node</i> disertakan komponen untuk mengkonsistenkan rancangan maka komponen yang diikutsertakan harus sesuai dengan komponen yang telah didefinisikan sebelumnya pada diagram komponen |
| Kebergantungan / dependency  | Kebergantungan antar <i>node</i> , arah panah mengarah pada <i>node</i> yang dipakai |
| Link  | relasi antar <i>node</i> |

2.8 Metodologi Pengembangan Sistem (*Waterfall*)

Menurut Pressman, Sommerville (2010), Metodologi waterfall adalah suatu proses pengembangan perangkat lunak berurutan, di mana kemajuan dipandang sebagai terus mengalir ke bawah (seperti air terjun) melewati fase-fase perencanaan, pemodelan, implementasi (konstruksi), dan pengujian.

Pengembangan dengan model ini adalah hasil adaptasi dari pengembangan perangkat keras, karena pada waktu itu belum terdapat metodologi pengembangan perangkat lunak yang lain.



Gambar 2.16.. Metodologi *Waterfall* Menurut Pressman, Sommerville (2010)

Adapun penjelasan urutan dari tahapan-tahapan yang dimiliki metodologi *waterfall* adalah sebagai berikut:

2.8.1 Analisis (*Analysis*)

Langkah ini merupakan analisa terhadap kebutuhan sistem. Pengumpulan data dalam tahap ini bisa melakukan sebuah penelitian, wawancara atau study literature. Seseorang sistem analisis akan menggali informasi sebanyak banyaknya dari *user* sehingga akan tercipta sebuah sistem komputer yang bisa melakukan tugas-tugas yang diinginkan oleh *user* tersebut.

2.8.2 Desain (*Design*)

Proses design akan menterjemahkan syarat kebutuhan sebuah perancangan perangkat lunak yang dapat diperkirakan sebelum dibuat koding. Proses ini berfokus pada struktur data, arsitektur perangkat lunak, representasi interface, dan detail (algoritma) prosedural. Tahapan ini akan menghasilkan dokumen yang disebut software requirement. Dokumen inilah yang akan digunakan programmer untuk melakukan aktivitas pembuatan sistemnya.

2.8.3 Pengkodena (*Coding*)

Tahapan ini merupakan proses penerjemahan bentuk desain menjadi kode atau bentuk/bahasa yang dapat dibaca oleh mesin. Setelah pengkodean selesai, dilakukan pengujian terhadap sistem dan juga kode yang sudah dibuat. Tujuannya untuk menemukan kesalahan yang mungkin terjadi untuk nantinya diperbaiki.

2.8.4 Pengujian Program (*Testing*)

Tahapan ini bisa dikatakan final dalam pembuatan sebuah sistem. Setelah melakukan analisa, design dan pengkodean maka sistem yang sudah jadi akan digunakan oleh user.

2.8.5 Pemeliharaan / Perawatan (*Maintenance*)

Perangkat lunak yang sudah disampaikan kepada pelanggan pasti akan mengalami perubahan. Perubahan tersebut bisa karena mengalami kesalahan karena perangkat lunak harus menyesuaikan dengan lingkungan (peripheral atau sistem operasi baru) baru, atau karena pelanggan membutuhkan perkembangan fungsional.

BAB III

ANALISIS DAN RANCANGAN SISTEM

3.1 Analisis (*Analysis*)

3.1.1 Analisis Perancangan Sistem

Perancangan aplikasi dimulai dengan analisis kebutuhan dengan cara melakukan pengamatan langsung terhadap sistem presensi yang digunakan. Analisis kebutuhan *website* seperti kebutuhan siapa saja yang boleh menggunakan *website*, apa saja data yang ingin di isi pada *website*, apa saja fungsi dari *website* tersebut dan lain-lain. Setelah dilakukan analisis kebutuhan diketahui bahwa aplikasi *multiple face recognition* untuk presensi kelas di teknik informatika Universitas Darma Persada tersebut belum ada. Berdasarkan analisis yang telah dilakukan maka kebutuhan fitur yang dibutuhkan adalah informasi presensi kelas yang dapat dilakukan oleh dosen dengan cara menggunakan teknik *face recognition*.

Teknik *Face Recognition* dipilih karena sistem monitoring presensi pada tingkat sekolah atau kampus saat ini sudah mulai beralih menggunakan alat elektronik seperti fingerprint, RFID dan pengenalan wajah. Presensi kelas di Teknik Informatika Universitas Darma Persada masih menggunakan dua cara tanpa menggunakan alat elektronik yaitu, *namely roll-call* dan *sign in sheet*. Kelebihan dari teknik *face recognition* dalam proses presensi kelas yang akan dikembangkan oleh peneliti yaitu dapat dilakukannya presensi menggunakan banyak wajah dalam satu waktu. Teknik *face recognition* dalam proses presensi kelas yang akan dikembangkan oleh peneliti dapat menjadi bahan penelitian selanjutnya agar proses presensi dapat dilakukan lebih mudah dari proses presensi

yang sudah ada. Contoh penelitian yang dapat dilakukan selanjutnya yaitu proses *face recognition* untuk presensi dapat terus berjalan setiap kali ada mahasiswa yang masuk kedalam kelas ataupun keluar kelas.

3.1.2 Analisis Kebutuhan

3.1.2.1 Observasi

Observasi adalah suatu cara pengumpulan data dengan melakukan pengamatan secara langsung dan melakukan pencatatan secara sistematis terhadap objek yang akan diteliti. Observasi dilakukan oleh peneliti dengan cara melakukan pengamatan dan pencatatan terhadap aktivitas pelaksanaan kegiatan yang terkait dengan presensi kelas di Teknik Informatika Universitas Darma Persada.

3.1.2.2 Wawancara

Wawancara dilakukan dengan mahasiswa serta dosen yang terlibat langsung dengan kegiatan presensi kelas di Teknik Informatika Universitas Darma Persada. Wawancara dalam penelitian ini dilakukan untuk mendapatkan data dan informasi yang lebih lengkap.

3.1.2.3 Studi Kepustakaan

Studi pustaka merupakan metode pengumpulan data yang diarahkan kepada pencarian data dan informasi melalui dokumen-dokumen, baik dokumen tertulis, foto-foto, gambar, maupun dokumen elektronik yang dapat mendukung dalam proses penelitian. Studi pustaka dilakukan oleh penulis dengan cara mempelajari teori-teori dari sumber bacaan dan

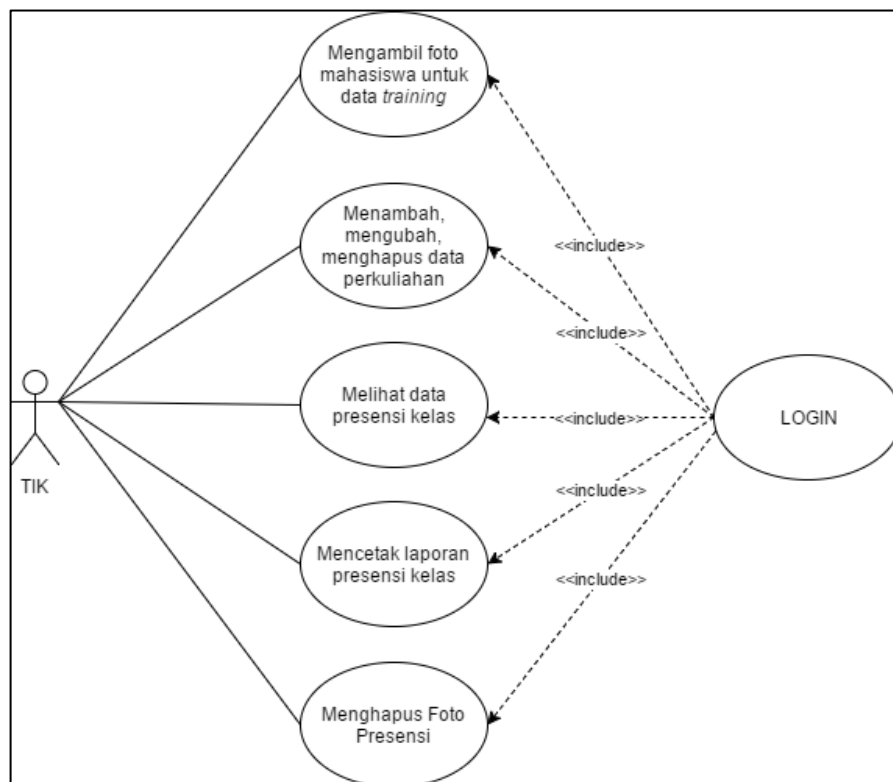
3.2 Desain (*Design*)

3.2.1 Perancangan Sistem

Perancangan sistem yang dibuat menggunakan *Unified Modelling Language* (UML) diagram yang meliputi *Use case diagram*, *Activity diagram*, *Sequence diagram* dan *Deployment Diagram*

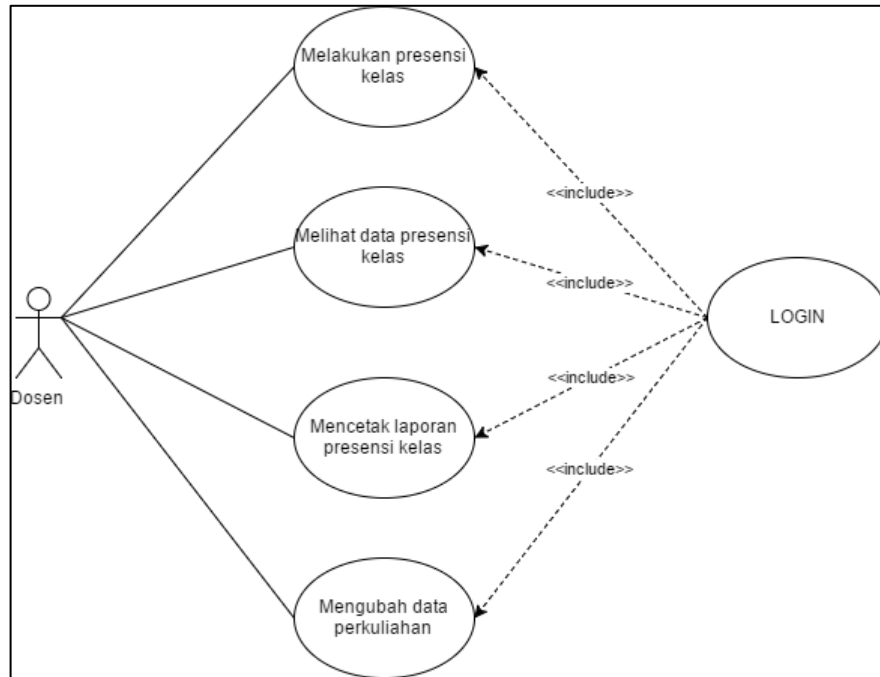
3.2.1.1 Use Case Diagram

Use case diagram memperlihatkan hubungan – hubungan yang terjadi antara aktor dengan use case – use case dalam sistem.



Gambar 3.1. *Use case Diagram TIK*

Gambar 3.1 adalah gambaran *use case diagram* dari TIK. Aktor TIK terhubung dengan 5 *use case*, yaitu *login*, mengambil gambar mahasiswa untuk data *training*, mengolah data perkuliahan seperti menambah, mengubah dan menghapus data, melihat data presensi kelas, dan mencetak laporan presensi kelas.

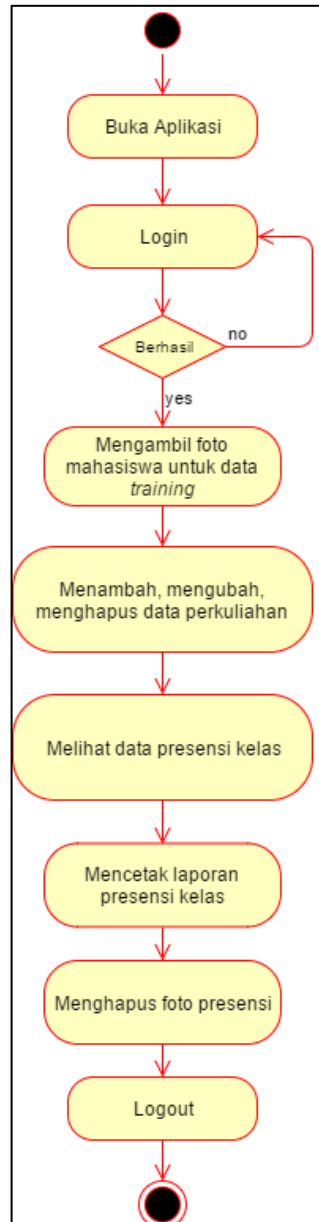


Gambar 3.2. *Use case* Diagram Dosen

Gambar 3.2 adalah gambaran *use case* diagram dari dosen. Aktor dosen terhubung dengan 5 *use case*, yaitu *login*, melakukan presensi kelas, melihat data presensi kelas, mencetak laporan presensi kelas dan mengubah data perkuliahan.

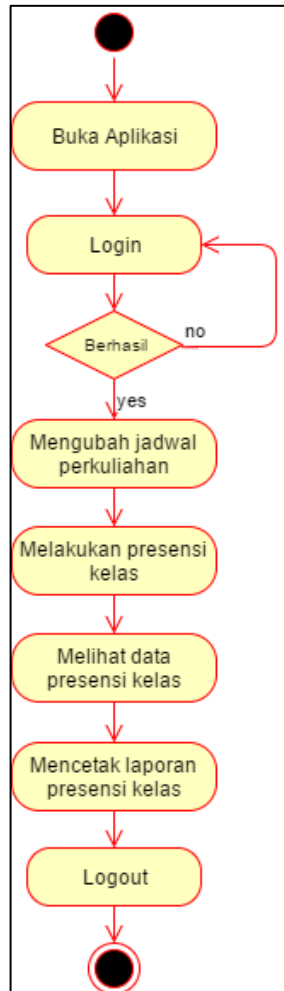
3.2.1.2 *Activity* Diagram

Activity diagram adalah salah satu cara untuk memodelkan *event – event* yang terjadi dalam suatu *use case*. Berikut adalah *activity* diagram dari Aplikasi *Multiple Face Recognition* untuk presensi kelas di Teknik Informatika Universitas Darma. *Activity* diagram memungkinkan siapapun yang melakukan proses untuk memilih urutan dalam melakukannya. Dengan kata lain, diagram hanya menyebutkan aturan-aturan rangkaian dasar yang harus kita ikuti.



Gambar 3.3. *Activity Diagram TIK*

Dari *activity diagram* gambar 3.3. *activity diagram* diatas dapat diketahui *activity TIK* yang ada di aplikasi *multiple face recognition* untuk presensi kelas di Teknik Informatika Universitas Darma. Pertama seseorang yang ditugaskan sebagai TIK masuk mengunjungi aplikasi dengan akses TIK lalu TIK dapat melakukan aktivitas sesuai fitur akses TIK tersebut.



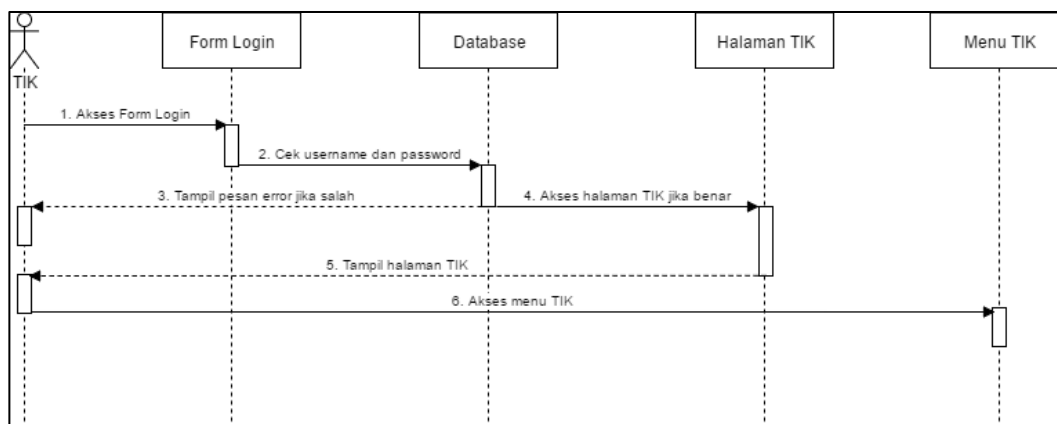
Gambar 3.4. Activity Diagram Dosen

Dari *activity* diagram gambar 3.4. *activity* diagram diatas dapat diketahui *activity* dosen yang ada di aplikasi *multiple face recognition* untuk presensi kelas di Teknik Informatika Universitas Darma. Pertama seseorang yang ditugaskan sebagai dosen masuk mengunjungi aplikasi dengan akses dosen lalu dosen dapat melakukan aktivitas sesuai fitur akses dosen tersebut.

3.2.1.3 Sequence Diagram

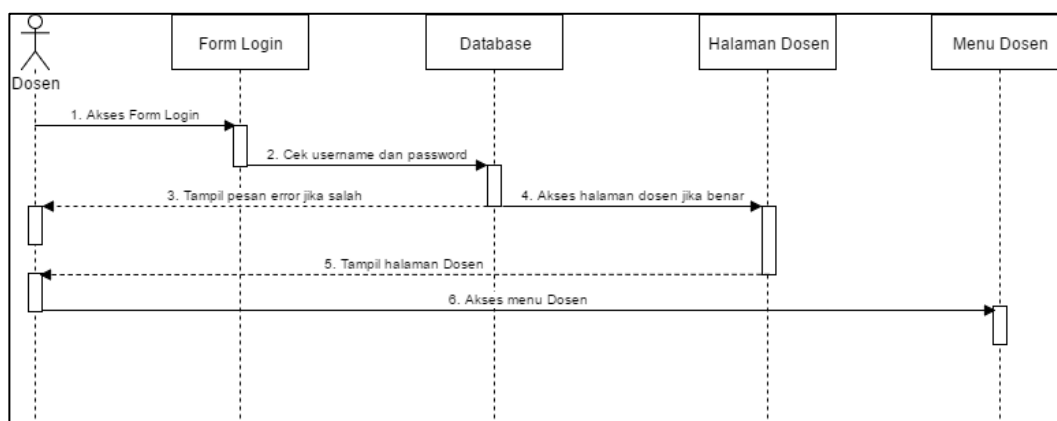
Sequence diagram adalah salah satu cara untuk menunjukkan bagaimana kelompok-kelompok objek saling berkolaborasi dalam beberapa behavior. Berikut

adalah sequence diagram dari Aplikasi *Multiple Face Recognition* untuk presensi kelas di Teknik Informatika Universitas Darma Persada.



Gambar 3.5. *Sequence Diagram* TIK

Dari *sequence* diagram gambar 3.5. dapat diketahui bahwa TIK dapat masuk halaman aplikasi dengan cara membuka form *login* lalu masukan *username* dan *password* yang *valid* dengan akses TIK dan kemudian dapat melakukan aktivitas sesuai dengan fitur yang tersedia.

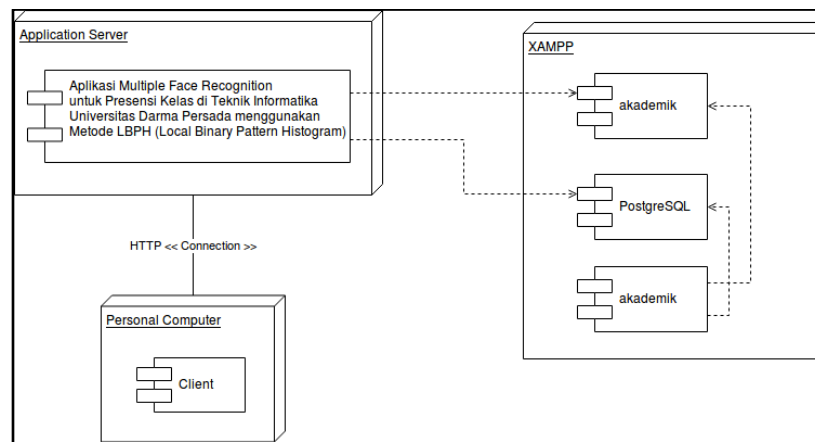


Gambar 3.6. *Sequence Diagram* Dosen

Dari *sequence* diagram gambar 3.6 dapat diketahui bahwa dosen dapat masuk halaman aplikasi dengan cara membuka form *login* lalu masukan *username* dan *password* yang *valid* dengan akses TIK dan kemudian dapat melakukan aktivitas sesuai dengan fitur yang tersedia.

3.2.1.4 Deployment Diagram

Deployment Diagram pada dasarnya menggambarkan detail bagaimana komponen infrastruktur sistem, dimana komponen akan terletak pada *user - user*, aplikasi server dan kemampuan jaringan pada lokasi spesifikasi *database server*.



Gambar 3.7. *Deployment* Diagram Aplikasi

3.2.2 Rancangan Database

Database sangat perlu untuk suatu pembangunan aplikasi didalam suatu sistem, terutama dalam proses penyimpanan data. Berikut ini merupakan rancangan *database* yang dibutuhkan di dalam aplikasi *Multiple Face Recognition* untuk Presensi Kelas.

3.2.2.1 Tabel User

Tabel 3.1. Struktur Tabel *User*

| Field | Tipe | Index |
|----------|-----------------------|-------------|
| nama | character varying(50) | |
| nik | numeric | Primary Key |
| password | character varying(30) | |
| level | character varying(10) | |

Tabel 3.1 merupakan rancangan tabel *user* untuk menyimpan data - data *user*. Tabel diatas terdiri dari *field* nama, nik, password, dan level. *Field* nik sebagai *Primary Key*.

3.2.2.2 Tabel Dosen

Tabel 3.2. Struktur Tabel Dosen

| Field | Tipe | Index |
|---------|---------|-------------|
| nik | numeric | Primary Key |
| idkelas | integer | |

Tabel 3.2 merupakan rancangan tabel dosen untuk menyimpan data - data dosen. Tabel diatas terdiri dari *field* nik dan idkelas. *Field* nik sebagai *Primary Key*.

3.2.2.3 Tabel Mata Kuliah

Tabel 3.3. Struktur Tabel ak_matakuliah

| Field | Tipe | Index |
|--------|------------------------|-------------|
| idmk | character varying(15) | Primary Key |
| namamk | character varying(100) | |
| sksmk | numeric | |

Tabel 3.3 merupakan rancangan tabel ak_matakuliah untuk menyimpan data-data mata kuliah. Tabel diatas terdiri dari idmk, namamk dan sksmk. *Field* idmk sebagai *Primary Key*.

3.2.2.4 Tabel Kelas

Tabel 3.4. Struktur Tabel ak_kelas

| Field | Tipe | Index |
|-----------|------------------------|-------------|
| idkelas | serial | Primary Key |
| idmk | character varying(15) | |
| namakelas | character varying(100) | |

Tabel 3.4 merupakan rancangan tabel ak_kelas untuk menyimpan data kelas dari setiap mata kuliah yang tersedia. Tabel ak_kelas terdiri dari idkelas, idmk dan namakelas. *Field* idkelas sebagai *Primary Key*.

3.2.2.5 Tabel Perkuliahan

Tabel 3.5. Struktur Tabel ak_perkuliahan

| Field | Tipe | Index |
|-------------------|-----------------------|-------------|
| idjadwal | serial | Primary Key |
| idruang | character varying(10) | |
| idkelas | integer | |
| tgljadwal | date | |
| waktumulai | character varying(10) | |
| waktuselesai | character varying(10) | |
| statusperkuliahan | character(1) | |

Tabel 3.5 merupakan rancangan tabel ak_perkuliahan untuk menyimpan data jadwal dari setiap kelas yang tersedia. Tabel ak_perkuliahan terdiri dari idjadwal, idruang, idkelas, tgljadwal, waktumulai, waktuselesai dan statusperkuliahan. *Field* idjadwal sebagai *Primary Key*.

3.2.2.6 Tabel KRS (Kartu Rencana Studi)

Tabel 3.6. Struktur Tabel ak_krs

| Field | Tipe | Index |
|---------|-----------------------|-------|
| idkelas | integer | |
| nim | character varying(20) | |

Tabel 3.6 merupakan rancangan tabel ak_krs untuk menyimpan data kelas yang diambil oleh setiap mahasiswa. Tabel ak_krs terdiri dari idkelas dan nim.

3.2.2.7 Tabel facerec

Tabel 3.7. Struktur Tabel facerec

| Field | Tipe | Index |
|---------|-----------------------|-------|
| idkelas | integer | |
| nim | character varying(20) | |

Tabel 3.7 merupakan rancangan tabel facerec untuk menyimpan data dari wajah yang berhasil dikenal. Tabel facerec terdiri dari nim, tgl, waktu dan idkelas.

3.2.2.8 Tabel Presensi Mahasiswa

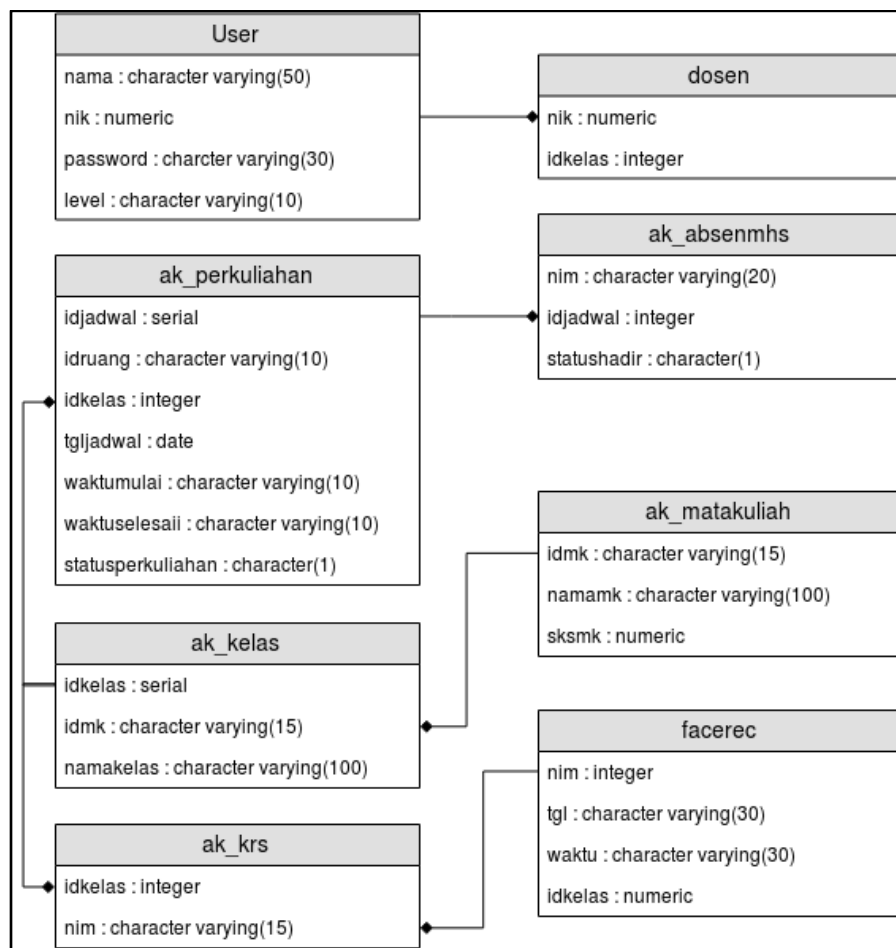
Tabel 3.8. Struktur Tabel ak_absensimhs

| Field | Tipe | Index |
|-------------|-----------------------|-------|
| nim | character varying(20) | |
| idjadwal | integer | |
| statushadir | character(1) | |

Tabel 3.8 merupakan rancangan tabel ak_absensimhs untuk menyimpan data kehadiran mahasiswa dari setiap jadwal yang tersedia. Tabel ak_absensimhs terdiri dari nim, idjadwal dan statushadir.

3.2.3 Diagram Relasi

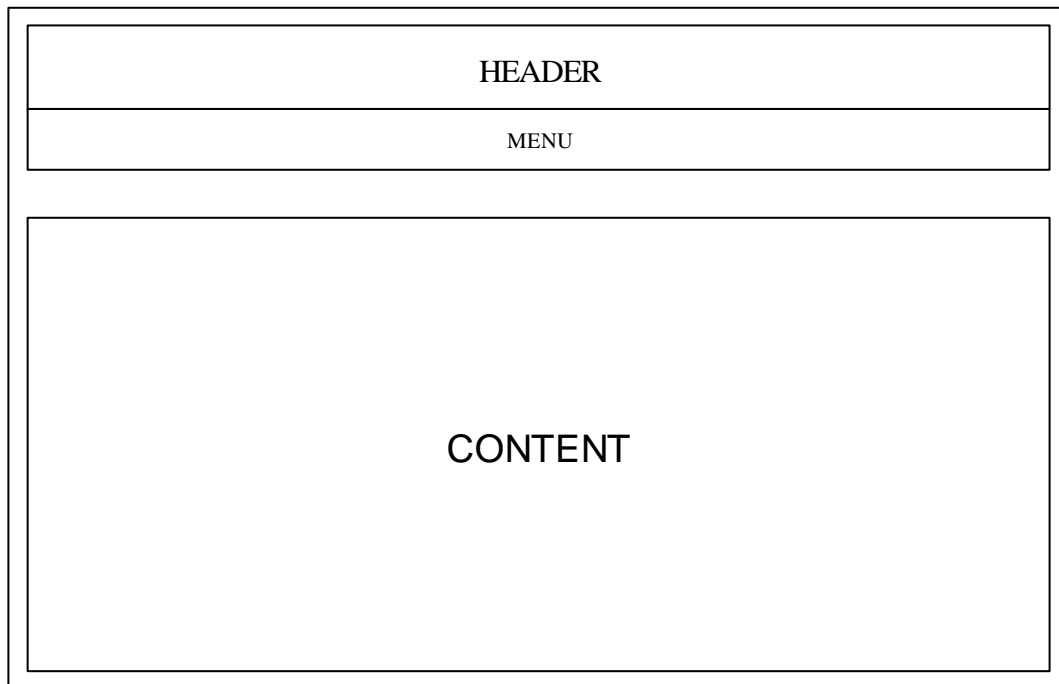
Berikut diagram relasi dari aplikasi *multiple face recognition* untuk presensi kelas di Teknik Informatika Universitas Darma Persada. Terdapat 8 tabel ak_perkuliahan, ak_absensimhs, ak_kelas, ak_krs, ak_matakuliah, User, facerec, dosen.



Gambar 3.8. Diagram Relasi

3.2.4 Rancangan Tampilan

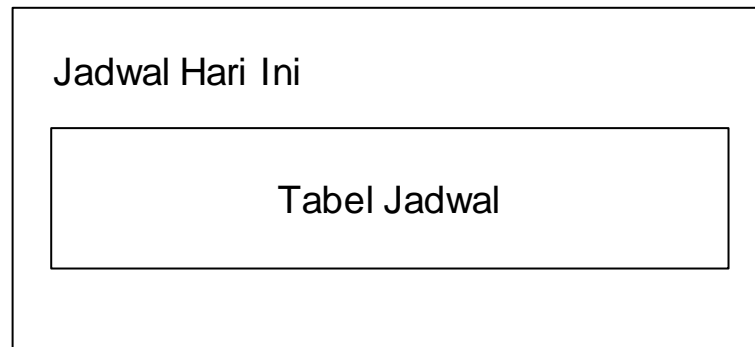
Langkah awal dalam pembuatan halaman suatu aplikasi adalah terlebih dahulu merancang tampilan halaman aplikasi tersebut, dan kemudian setelah rancangan selesai dimulailah langkah pembuatan fungsi - fungsi dan konten dari aplikasi tersebut.



Gambar 3.9. Rancangan Halaman Utama

3.2.4.1 Rancangan Tampilan *Dashboard*

Tampilan *dashboard* merupakan tampilan awal setelah proses *login* berhasil. Tampilan *dashboard* untuk menampilkan jadwal kelas yang berlangsung pada hari ini.



Gambar 3.10. Rancangan Halaman *Dashboard*

3.2.4.2 Rancangan Tampilan Peserta Kelas

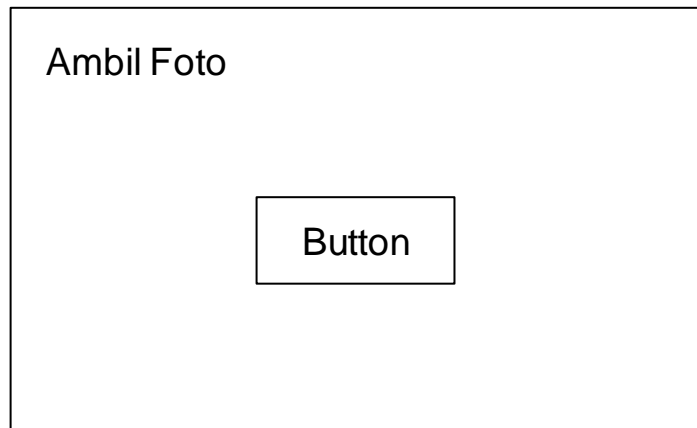
Tampilan peserta kelas untuk menampilkan data peserta kelas pada kelas yang dipilih.



Gambar 3.11. Rancangan Halaman Peserta Kelas

3.2.4.3 Rancangan Tampilan Tambah Foto

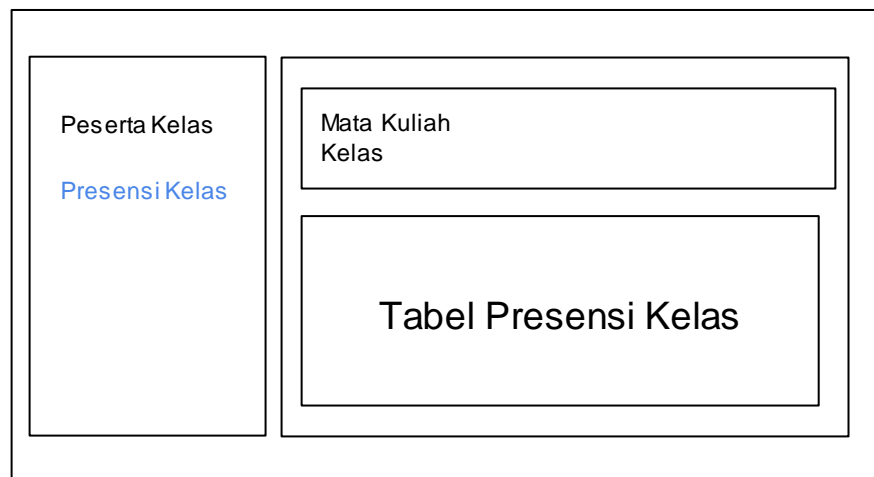
Tampilan tambah foto dapat diakses oleh TIK untuk mengambil foto mahasiswa yang akan digunakan sebagai data *training*.



Gambar 3.12. Rancangan Halaman Tambah Foto

3.2.4.4 Rancangan Tampilan Presensi Kelas

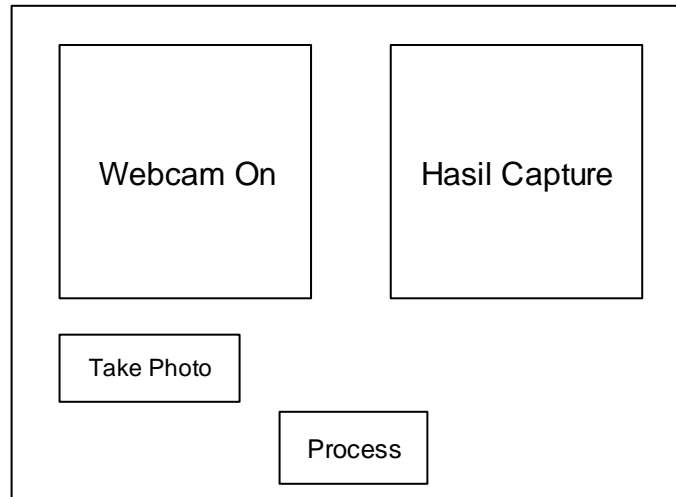
Tampilan presensi kelas untuk menampilkan data presensi kelas pada kelas yang dipilih.



Gambar 3.13. Rancangan Halaman Presensi Kelas

3.3.1.1 Rancangan Tampilan Presensi *Face Recognition*

Tampilan presensi *face recognition* untuk melakukan presensi kelas menggunakan *face recognition*.



Gambar 3.14. Rancangan Tampilan Presensi *Face Recognition*

BAB IV

IMPLEMENTASI HASIL

4.1 Implementasi Sistem

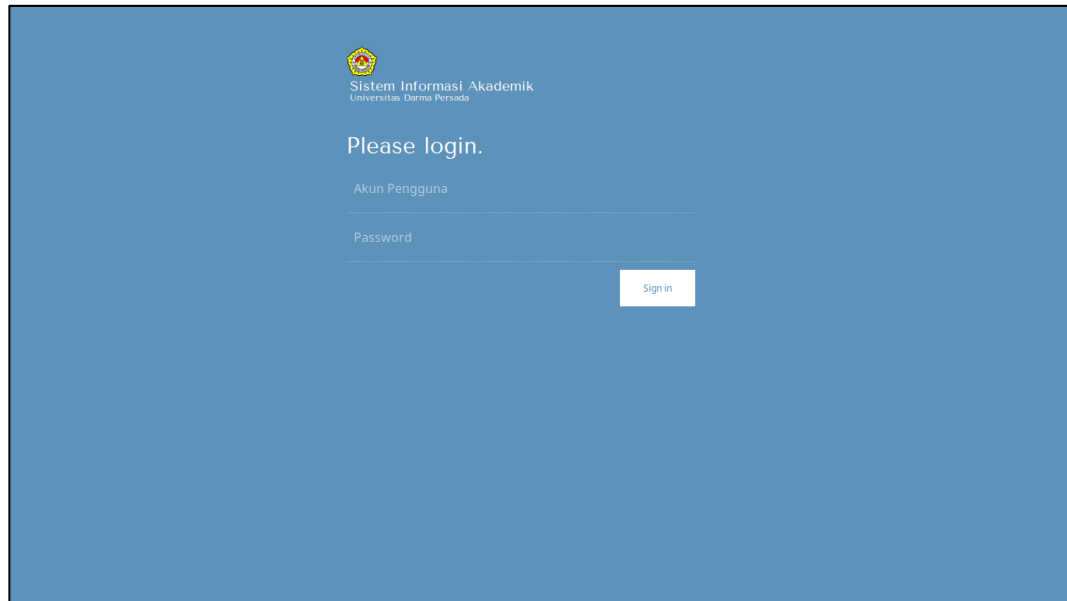
Pada implementasi sistem ini dilakukan secara offline dan menggunakan spesifikasi komputer sebagai berikut:

| | |
|------------------|--|
| Perangkat | : Laptop Lenovo G-40 |
| Operating system | : Linux Mint 64 bit |
| Processor | : Intel(R) Core(TM) i7-4510U CPU @ 2.00GHz |
| Memory | : 4GB |
| Hard disk | : 500 GB |
| Display | : AMD Radeon A5 Graphics |
| Web Server | : Apache 2.4.34 |
| Database Server | : PostgreSQL |
| Script Server | : PHP 5.6.3 |
| Browser | : Firefox 67.0.2 (64 bit) |

Implementasi sistem ini juga akan membahas setiap halaman sistem presensi kelas berbasis *multiple face recognition*. Halaman yang ada pada sistem ini adalah :

4.1.1 Halaman *Form Login*

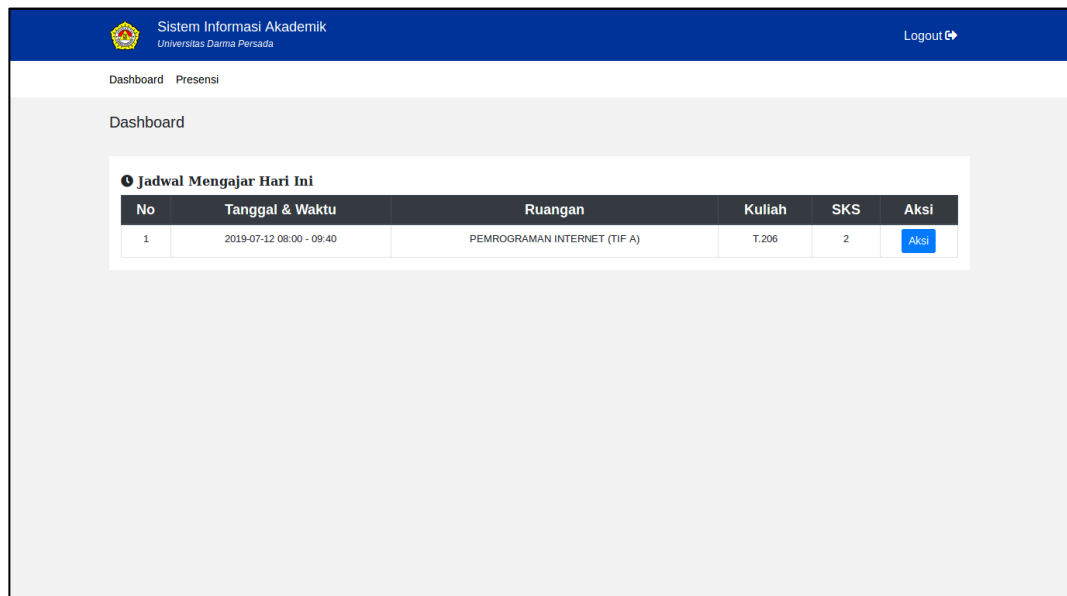
Halaman *form login* merupakan tampilan awal dari aplikasi. Pada halaman ini berfungsi untuk mengatur akses masuk ke halaman utama aplikasi. Halaman ini dapat diakses oleh dosen dan mahasiswa.



Gambar 4.1. Halaman *Login*

4.1.2 Halaman Utama (*Dashboard*)

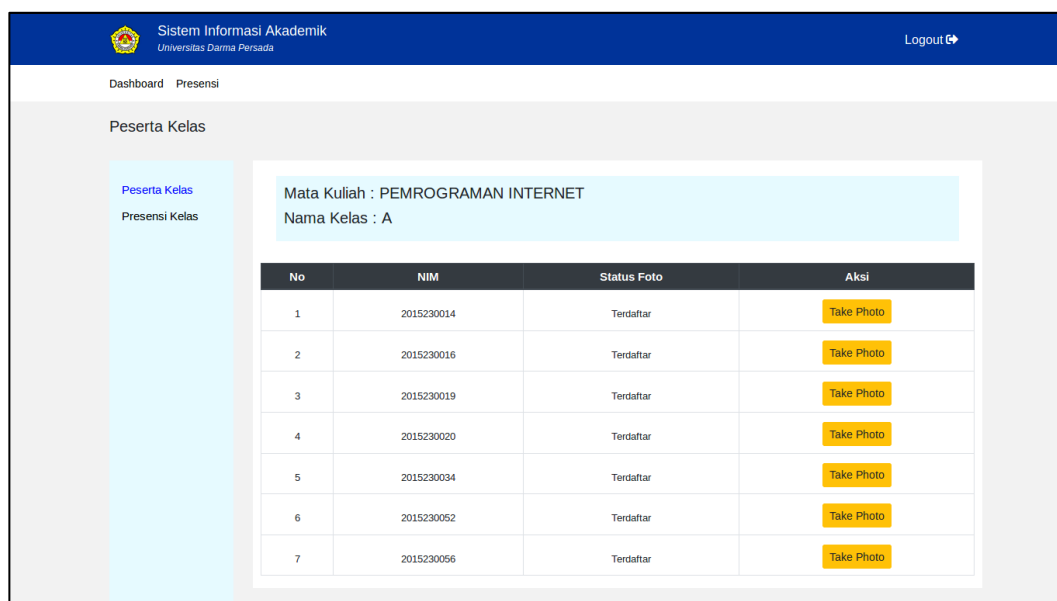
Halaman utama adalah halaman awal setelah login berhasil. Halaman ini berisikan menu-menu dalam aplikasi serta data kelas yang berlangsung pada hari tersebut.



Gambar 4.2. Halaman *Dashboard*

4.1.3 Halaman Peserta Kelas

Halaman Peserta Kelas merupakan halaman untuk melihat peserta kelas yang berada pada kelas tersebut. Pada halaman ini dosen dapat mengetahui peserta kelas yang tidak memiliki foto yang akan digunakan sebagai data *training* untuk proses presensi kelas. Halaman ini hanya dapat diakses oleh dosen.



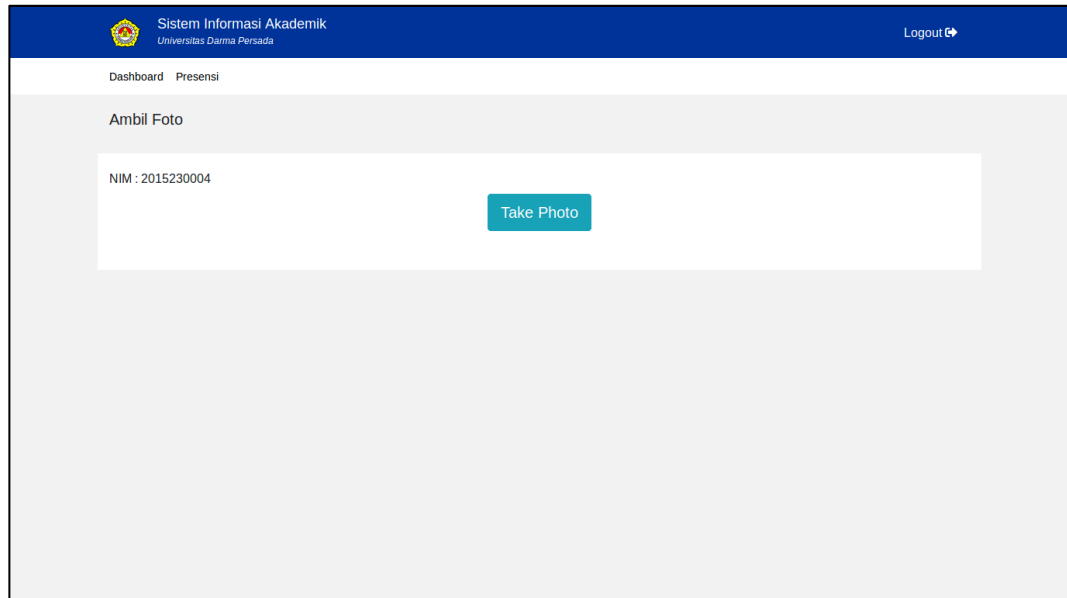
The screenshot shows the 'Peserta Kelas' page. The header includes the university logo and name, and a 'Logout' button. The main content area displays the course name 'PEMROGRAMAN INTERNET' and the class name 'A'. Below this is a table with columns for 'No', 'NIM', 'Status Foto', and 'Aksi'. The table lists 7 students, all with 'Terdftar' status and a 'Take Photo' button.

| No | NIM | Status Foto | Aksi |
|----|------------|-------------|------------|
| 1 | 2015230014 | Terdftar | Take Photo |
| 2 | 2015230016 | Terdftar | Take Photo |
| 3 | 2015230019 | Terdftar | Take Photo |
| 4 | 2015230020 | Terdftar | Take Photo |
| 5 | 2015230034 | Terdftar | Take Photo |
| 6 | 2015230052 | Terdftar | Take Photo |
| 7 | 2015230056 | Terdftar | Take Photo |

Gambar 4.3. Halaman Peserta Kelas

4.1.4 Halaman Tambah Foto Mahasiswa

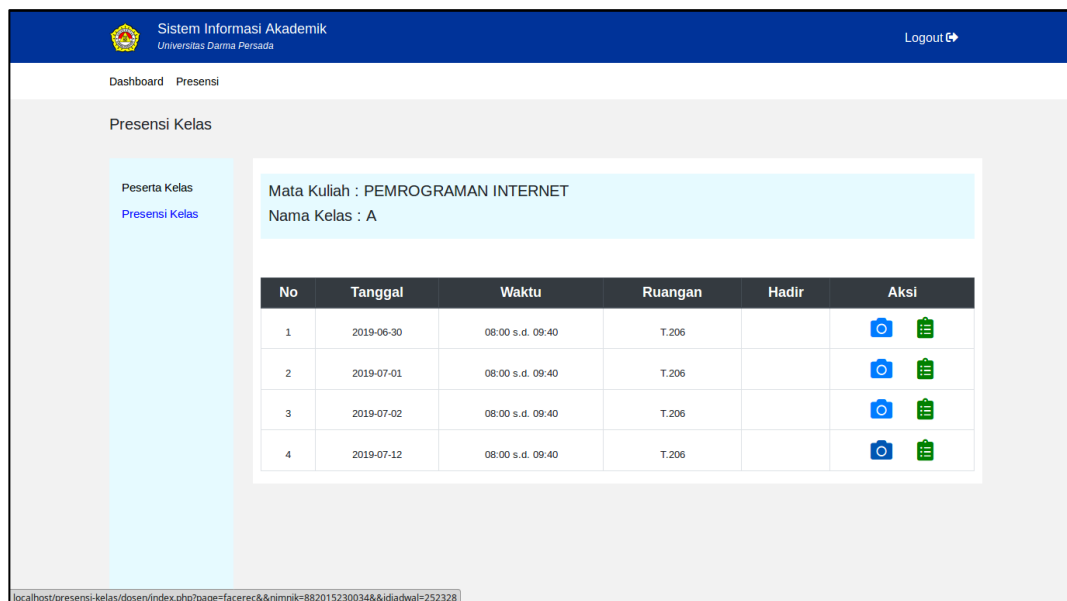
Halaman Tambah Foto berfungsi untuk mengambil foto mahasiswa dan akan disimpan sebagai dataset. Halaman ini hanya dapat diakses oleh dosen.



Gambar 4.4. Halaman Tambah Foto Mahasiswa

4.1.5 Halaman Presensi Kelas

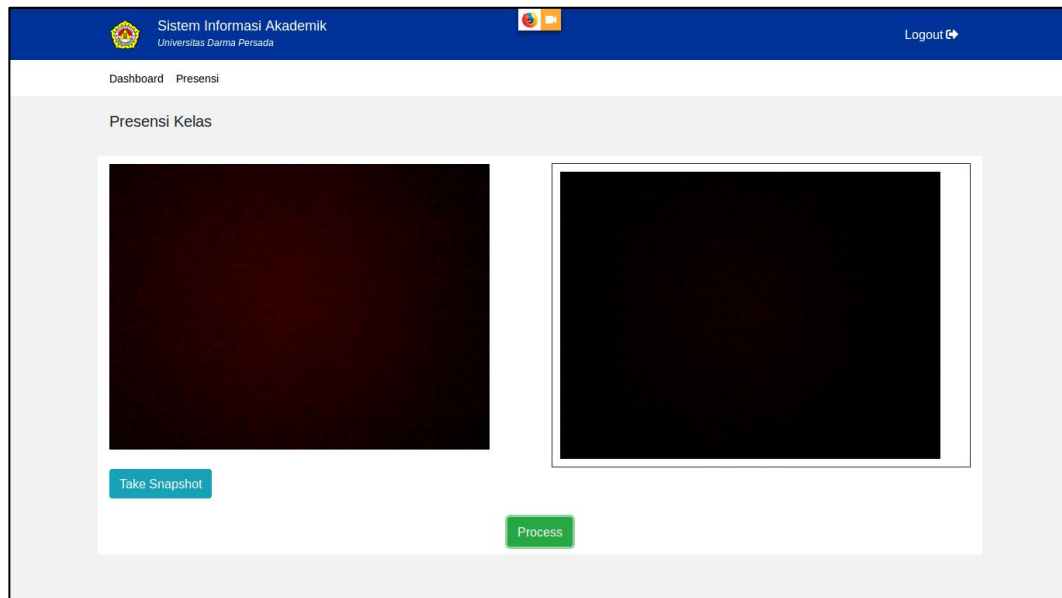
Halaman Presensi Kelas merupakan halaman untuk melihat jadwal kelas yang berada pada kelas tersebut. Pada halaman ini dosen dapat mengetahui jumlah peserta kelas yang hadir pada kelas tersebut serta daftar peserta kelas yang hadir. Halaman ini hanya dapat diakses oleh dosen.



Gambar 4.5. Halaman Presensi Kelas

4.1.6 Halaman Presensi *Face Recognition*

Halaman Presensi *Face Recognition* Kelas merupakan halaman untuk melihat jadwal kelas yang berada pada kelas tersebut. Pada halaman ini dosen dapat mengetahui jumlah peserta kelas yang hadir pada kelas tersebut serta daftar peserta kelas yang hadir. Halaman ini hanya dapat diakses oleh dosen.



Gambar 4.6. Halaman Presensi *Face Recognition*

4.2 Pengkodean (*Coding*)

Tahapan ini merupakan proses penerjemahan bentuk desain menjadi kode atau bentuk/bahasa yang dapat dibaca oleh mesin. Setelah pengkodean selesai, dilakukan pengujian terhadap sistem dan juga kode yang sudah dibuat. Tujuannya untuk menemukan kesalahan yang mungkin terjadi untuk nantinya diperbaiki.

| | |
|------------------|--|
| Perangkat | : Laptop Lenovo G-40 |
| Operating system | : Linux Mint 64 bit |
| Processor | : Intel(R) Core(TM) i7-4510U CPU @ 2.00GHz |
| Memory | : 4GB |
| Hard disk | : 500 GB |

| | |
|-----------------|---------------------------|
| Display | : AMD Radeon A5 Graphics |
| Web Server | : Apache 2.4.34 |
| Database Server | : PostgreSQL |
| Script Server | : PHP 5.6.3 |
| Browser | : Firefox 67.0.2 (64 bit) |

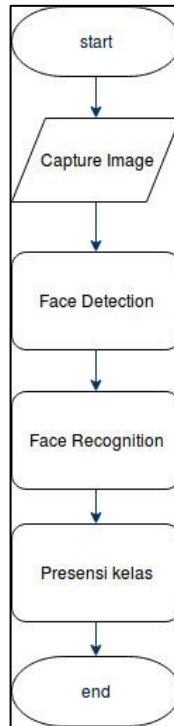
Dalam aplikasi ini tampilannya menggunakan HTML, CSS, Javascript, Bootstrap, untuk bahasa pemrogramannya menggunakan Python & PHP dan menggunakan database PostgreSQL dengan text editor Sublime Text.

4.3 Pengujian Program (*Testing*)

Tahapan ini bisa dikatakan final dalam pembuatan sebuah sistem. Setelah melakukan analisa, design dan pengkodean maka sistem yang sudah jadi akan digunakan oleh user.

4.3.1 Cara Kerja Aplikasi

Aplikasi *multiple face recognition* untuk presensi kelas memiliki dua proses utama yaitu deteksi wajah dan pengenalan wajah. Berikut diagram alur sistem aplikasi *multiple face recognition* untuk presensi kelas:



Gambar 4.7. Alur sistem






Berdasarkan gambar 4.7 alur sistem dimulai dengan proses *capture image* agar terdapat gambar yang dapat diproses. Proses selanjutnya yaitu *face detection*, dimana pada tahap ini dilakukan pencarian wajah yang terdapat pada gambar. Proses *face recognition* merupakan proses pengenalan wajah berdasarkan gambar wajah yang didapat dari proses *face detection*. Hasil dari proses *face recognition* adalah nim yang akan disimpan sebagai tanda kehadiran mahasiswa.






Aplikasi *multiple face recognition* dapat dijalankan dengan cara menjalankan *Flask* API terlebih dahulu di *command prompt*. Cara menjalankan *Flask* API adalah dengan mengetik perintah *python3 main.py* di *command prompt*. Saat *Flask* API berjalan, proses *face recognition* dapat dilakukan dengan baik.

4.3.1.1 Data *Training*

Data *training* berjumlah 500 data yang terdiri dari 10 mahasiswa dengan masing-masing foto wajah berjumlah 50. Pengambilan data dilakukan secara manual dengan mengambil foto dari setiap mahasiswa.

Tabel 4.1. Contoh Data *Training*

| No. | Citra | Nama Citra | Label |
|-----|---|-----------------------|------------|
| 1. |  | User.2015230014.1.jpg | 2015230014 |
| 2. |  | User.2015230016.1.jpg | 2015230016 |
| 3. |  | User.2015230019.1.jpg | 2015230019 |
| 4. |  | User.2015230034.1.jpg | 2015230034 |
| 5. |  | User.2015230052.1.jpg | 2015230052 |

| | | | |
|-----|---|-----------------------|------------|
| 6. |  | User.2015230056.1.jpg | 2015230056 |
| 7. |  | User.2015230058.1.jpg | 2015230058 |
| 8. |  | User.2015230082.1.jpg | 2015230082 |
| 9. |  | User.2015230103.1.jpg | 2015230103 |
| 10. |  | User.2015230117.1.jpg | 2015230117 |

4.3.1.2 Deteksi Wajah

Setelah pengambilan gambar pada aplikasi, gambar akan masuk proses pendeteksi wajah dengan menggunakan metode *Haar Cascade Classifier*. *Haar Cascade Classifier* merupakan metode *object detection* yang dapat digunakan untuk mendeteksi wajah pada suatu citra digital. Citra akan dibaca oleh fitur Haar dengan bantuan library OpenCV 2.2 yaitu `haarcascade_frontalface_alt.xml`. Dalam OpenCV 2.2 terdapat suatu library untuk membantu proses deteksi wajah

yaitu `Haarcascade_frontalface_alt` yang berfungsi sebagai proses untuk memanggil beberapa fitur Haar dalam suatu gambar.

Fitur Haar adalah fitur yang digunakan dalam metode Viola-Jones yang dapat juga disebut fitur gelombang tunggal bujur sangkar (satu interval tinggi dan satu interval rendah), sedangkan untuk dua dimensi disebut sebagai satu terang dan satu gelap. Adanya fitur Haar ditentukan dengan cara mengurangi rata-rata piksel pada daerah gelap dan rata-rata piksel pada daerah terang. Jika nilai perbedaannya itu di atas nilai ambang atau threshold, maka dapat dikatakan bahwa fitur tersebut ada.

Selanjutnya untuk menentukan ada atau tidaknya dari ratusan fitur Haar pada sebuah gambar dan pada skala yang berbeda secara efisien digunakan Integral Image. Pada umumnya, pengintegrasian tersebut menambahkan unit-unit kecil secara bersamaan. Dalam hal ini unit-unit kecil tersebut adalah nilai-nilai piksel. Nilai integral untuk masing-masing piksel adalah jumlah dari semua piksel-piksel dari atas sampai bawah. Dimulai dari kiri atas sampai kanan bawah, keseluruhan gambar itu dapat dijumlahkan dengan beberapa operasi bilangan bulat per-piksel.

Tahapan yang terakhir adalah menampilkan objek sampel gambar yang telah terdeteksi wajah ataupun bukan wajah, dengan memberi tanda bujur sangkar jika objek tersebut dianggap sebagai daerah wajah. Jika gambar tersebut dikenal sebagai wajah maka akan tampil gambar kotak berwarna hijau.



Gambar 4.8. Hasil Deteksi Wajah

4.3.1.3 Pengenalan Wajah

Setelah melalui tahap deteksi wajah dengan menggunakan metode Haar Cascade Classifier selanjutnya foto akan diproses dengan menggunakan metode Local Binary Patterns Histograms (LBPH). Pada proses ini gambar hasil capture video akan diproses saat button Process di klik. Metode LBPH merupakan metode yang sudah disiapkan oleh Library Python OpenCV. Metode LBPH akan dipanggil dengan inisialisasi `cv2.face.LBPHFaceRecognizer_create()`.

Citra wajah hasil deteksi dari metode Haar Cascade Classifier akan di proses untuk mendapatkan nilai histogramnya. Sehingga sistem dapat membandingkan nilai histogram dari gambar input yang berasal dari kamera dengan nilai histogram yang dimiliki citra wajah pada data training. Menerapkan metode LBPH pada pengenalan wajah secara real time dilakukan dengan proses training terlebih dahulu untuk mendapatkan nilai histogram yang kemudian akan dibandingkan nilainya dengan nilai citra yang akan dideteksi secara langsung (*real time*). Untuk mendapatkan kecocokan gambar dengan nilai yang sudah disimpan di database, maka perlu dibandingkan dua histogram antara gambar yang dideteksi dengan gambar pada database dan mencari jarak nilai histogram

terdekatnya. Jadi, output dari algoritma tersebut adalah nomor identifikasi dari tiap gambar yang diubah dengan nama pemilik wajah tersebut. Kemudian pada akhirnya ditampilkan pada monitor.



Gambar 4.9. Contoh Hasil *Face Recognition*

4.3.2 Evaluasi Hasil Pengujian Aplikasi

4.3.2.1 Evaluasi Aplikasi

Analisis hasil dilakukan dengan melihat proses input data sampai dengan melihat hasil presensi kelas berdasarkan *multiple face recognition*. Pada pengujian sistem ini diberikan beberapa penilaian sebagai berikut : Adanya jeda saat menjalankan fitur proses *face recognition* dikarenakan aplikasi akan membaca terlebih dahulu data training yang ada di dalam file xml untuk mengetahui hasil dari pengenalan wajah.

1. Adanya jeda saat menjalankan fitur proses *face recognition* dikarenakan aplikasi akan membaca terlebih dahulu data training yang ada di dalam file xml untuk mengetahui hasil dari pengenalan wajah.
2. Ada beberapa faktor yang sangat mempengaruhi hasil pengenalan wajah diantaranya adalah pencahayaan, jarak objek dari kamera, ekspresi wajah dan posisi wajah.

3. Data latih yang digunakan harus banyak agar hasil yang didapat lebih akurat karena analisa sangat bergantung dengan dataset yang ada.

4.3.2.2 Pengujian

Pengujian aplikasi *multiple face recognition* dilakukan dengan menggunakan foto berisi 1 wajah sebanyak 3 foto, 2 wajah sebanyak 3 foto, 3 wajah sebanyak 3 foto, 4 wajah sebanyak 3 foto, 5 wajah sebanyak 3 foto, dan 6 wajah sebanyak 3 foto. Pengujian dilakukan di Ruang Laboratorium dengan jarak 100cm - 150cm. Nilai akurasi dilakukan berdasarkan ketepatan sistem mengenali setiap wajah yang ada. Perhitungan akurasi dapat dilakukan dengan rumus :

$$\text{Akurasi} = \frac{\text{Jumlah wajah dikenali}}{\text{Jumlah wajah}} \times 100\%$$

Tabel 4.2. Hasil Analisa

| Jumlah Wajah Mahasiswa | Akurasi (%) | | | |
|------------------------|-------------------------|-------------------------|-------------------------|-------------------------------|
| | Pengambilan gambar ke-1 | Pengambilan gambar ke-2 | Pengambilan gambar ke-3 | Rata-rata akurasi (per-wajah) |
| 1 | 100 | 100 | 100 | 100 |
| 2 | 100 | 100 | 100 | 100 |
| 3 | 66.66 | 100 | 66.66 | 77.77 |
| 4 | 25 | 25 | 75 | 41.67 |
| 5 | 40 | 40 | 40 | 40 |
| 6 | 16.66 | 16.66 | 0 | 11.11 |
| Jumlah rata - rata | | | | 61.76 |

Dari hasil analisa di atas dapat diketahui jumlah rata-rata akurasi sistem mengenali setiap wajah pada foto adalah 61.76% atau 62%. Pada pengujian

dengan jumlah wajah 1 (satu), aplikasi dapat mengenali wajah dengan sangat baik dengan total rata-rata akurasi 100%. Pada pengujian dengan jumlah wajah 2 (dua), aplikasi dapat mengenali wajah dengan sangat baik dengan total rata-rata akurasi 100%. Pada pengujian dengan jumlah wajah 3 (tiga), aplikasi dapat mengenali wajah dengan baik dengan total rata-rata akurasi 77.77% sehingga presensi menggunakan tiga wajah sekaligus dapat dilakukan.

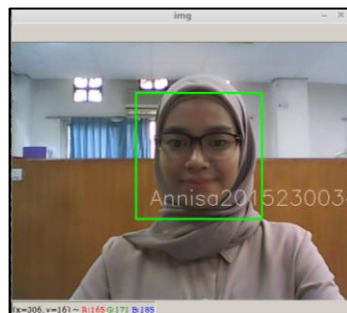
Pada pengujian dengan jumlah wajah 4 (empat), aplikasi dapat mengenali wajah dengan cukup baik dengan total rata-rata akurasi 41.67% sehingga presensi menggunakan empat wajah sekaligus dapat dilakukan dengan tingkan keberhasilan yang kecil. Pada pengujian dengan jumlah wajah 5 (lima), aplikasi dapat mengenali wajah dengan cukup baik dengan total rata-rata akurasi 40% sehingga presensi menggunakan lima wajah sekaligus dapat dilakukan dengan tingkan keberhasilan yang kecil. Pada pengujian dengan jumlah wajah 6 (enam), aplikasi dapat mengenali wajah dengan tidak cukup baik dengan total rata-rata akurasi 11.11% sehingga presensi menggunakan enam wajah sekaligus tidak dapat dilakukan.

Aplikasi *multiple face recognition* untuk presensi kelas di teknik informatika Universitas Darma Persada ini dapat dijalankan dengan baik menggunakan jumlah minimum wajah adalah 1 (satu) dan jumlah maksimum wajah adalah 5 (lima).

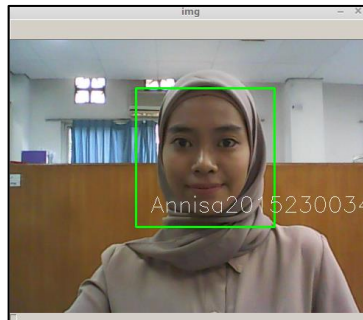
Tabel 4.3. Hasil Pengujian *Case*

| Percobaan | Sesuai Data Training | Berkacamata | | Perubahan Jilbab | Kesimpulan |
|-------------------------------|----------------------|-------------|-------|------------------|------------|
| | | Ya | Tidak | | |
| 1 | B | B | - | B | B |
| B=Berhasil, TB=Tidak Berhasil | | | | | |

Pada table 4.3 dilakukan percobaan pada satu wajah untuk melihat keberhasilan aplikasi dalam mengenali wajah dengan kondisi berbeda dari foto wajah yang digunakan sebagai data training. Foto yang digunakan untuk data *training* dan data pengujian *case* diambil dalam jarak 50-100cm dan cahaya ruang yang tidak jauh berbeda antara kedua data foto. Kesimpulan dari pengujian *case* adalah kondisi berbeda dari data *training* tidak mempengaruhi aplikasi dalam mengenal wajah. Berikut adalah contoh pengujian *case* berkacamata dan tidak berkacamata.



Gambar 4.10. Pengujian *case* menggunakan kacamata



Gambar 4.11. Pengujian *case* tidak menggunakan kacamata

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan dari bab-bab sebelumnya maka dapat disimpulkan sebagai berikut :

1. Perancangan aplikasi diawali dengan wawancara kebutuhan user serta observasi data mahasiswa, kemudian dilanjutkan dengan tahapan pengumpulan data foto mahasiswa untuk keperluan training data algoritma, lalu perancangan tampilan keseluruhan aplikasi dan selanjutnya proses training data dengan implementasi algoritma LBPH (*Local Binary Patterns Histograms*) ke dalam pengkodean yang dibuat dengan bahasa pemrograman PHP, Python dan database PostgreSQL.
2. Penerapan metode *haar cascade classifier* untuk melakukan deteksi wajah dapat dilihat dari tampilan kotak hijau yang akan diproses dalam pengenalan wajah.
3. Penerapan metode LBPH (*Local Binary Patterns Histograms*) untuk melakukan pengenalan wajah yang telah terdeteksi dengan metode *haar cascade classifier* sehingga presensi dapat dilakukan. Hal ini dilihat pada halaman presensi *face recognition* yang mana ketika hasil capture dilakukan proses pengenalan wajah, akan langsung tampil kotak hijau dan nim dari wajah yang dikenali.
4. Aplikasi *face recognition* dapat dijalankan jika *Flask* API dijalankan terlebih dahulu melalui perintah yang diketik di *command prompt*.

5. Dari pengujian aplikasi *multiple face recognition* untuk presensi kelas di teknik informatika Universitas Darma Persada diperoleh hasil sebagai berikut :
- a. Aplikasi *multiple face recognition* untuk presensi kelas di teknik informatika Universitas Darma Persada dapat mendeteksi wajah dengan jarak 30 cm - 150 cm menggunakan kamera *Logitech c170*.
 - b. Aplikasi *multiple face recognition* untuk presensi kelas di teknik informatika Universitas Darma Persada dapat dijalankan dengan *webcam* terpisah. Pemakaian kamera dengan kualitas yang lebih baik akan menghasilkan tangkapan gambar yang lebih baik sehingga gambar mudah dikenali.
 - c. Pencahayaan saat deteksi wajah dan pengenalan wajah diharapkan dalam kondisi cukup, tidak kurang dan juga tidak berlebih, karena dapat menyebabkan program berjalan tidak semestinya.
 - d. Penambahan benda seperti kacamata dan perubahan jilbab yang digunakan dapat dilakukan serta aplikasi dapat mengenali wajah.
5. Kelemahan - kelemahan aplikasi *multiple face recognition* untuk presensi kelas di teknik informatika Universitas Darma Persada antara lain :
- a. Aplikasi presensi kelas berbasis *multiple face recognition* tidak dapat mendeteksi dan mengenali wajah jika bagian wajah terlalu miring.
 - b. Aplikasi hanya dapat mendeteksi wajah dengan jarak 30 cm - 150 cm pada cahaya yang cukup.

- c. Aplikasi hanya dapat mengenali wajah dengan cahaya, jarak, posisi serta ekspresi sesuai dengan gambar yang tersimpan sebagai data *training*.
 - d. Aplikasi tidak dapat membedakan wajah yang bukan pas foto dan wajah yang pas foto.
 - e. Aplikasi tidak dapat memberikan notifikasi saat wajah yang berhasil diidentifikasi adalah wajah yang sudah pernah melakukan presensi kelas ataupun wajah yang tidak terdaftar dikelas yang dipilih.
6. Tingkat akurasi aplikasi *multiple face recognition* untuk presensi kelas yang dilakukan oleh Algoritma *Local Binary Patterns Histograms* adalah 62%.
7. Besarnya ukuran *file* gambar akan mempengaruhi ruang kapasitas penyimpanan, maka dari itu penghapusan data foto presensi dapat dilakukan dengan membuka *folder* penyimpanan foto presensi dan mengurutkan tanggal pengambilan yang tersimpan sebagai nama foto.

5.2 Saran

Penulis menyadari bahwa masih banyak kekurangan yang terdapat pada pembuatan aplikasi *multiple face recognition* untuk presensi kelas di teknik informatika Universitas Darma Persada, untuk itu sangat diperlukan adanya perbaikan terhadap aplikasi ini. Saran - saran yang dapat penulis berikan adalah :

1. Penambahan jumlah data *training* agar menghasilkan hasil *training* dengan tingkat akurasi prediksi mencapai nilai maksimal.
2. Pemakaian webcam yang lebih baik dari *Logitech c170* karena hasil deteksi dan pengenalan wajah akan semakin baik.

3. Pencahayaan saat deteksi wajah dan pengenalan wajah dalam kondisi cukup, tidak kurang dan tidak berlebih, karena dapat menyebabkan program tidak berjalan semestinya.
4. Perlu adanya fitur back-up data agar data-data tetap terjaga dengan baik dan aman.
5. Perlu dilakukan pengembangan aplikasi lebih lanjut agar aplikasi *multiple face recognition* untuk presensi kelas di teknik informatika dapat membedakan wajah yang bukan pas foto dan wajah pas foto.