# LEMBAR PENGESAHAN

# LEMBAR PENGESAHAN SIDANG

# PERNYATAAN KEASLIAN TUGAS AKHIR

# PERNYATAAN PERSETUJUAN PUBLIKASI

# MOTTO

# DAFTAR RIWAYAT HIDUP

# ABSTRAK

# ABSTRACK

# KATA PENGANTAR

# DAFTAR ISI

[LEMBAR PENGESAHAN i](#_Toc78361381)

[LEMBAR PENGESAHAN SIDANG ii](#_Toc78361382)

[PERNYATAAN KEASLIAN TUGAS AKHIR iii](#_Toc78361383)

[PERNYATAAN PERSETUJUAN PUBLIKASI iv](#_Toc78361384)

[MOTTO v](#_Toc78361385)

[DAFTAR RIWAYAT HIDUP vi](#_Toc78361386)

[ABSTRAK vii](#_Toc78361387)

[ABSTRACK viii](#_Toc78361388)

[KATA PENGANTAR ix](#_Toc78361389)

[DAFTAR ISI x](#_Toc78361390)

[DAFTAR TABEL xiii](#_Toc78361391)

[DAFTAR GAMBAR xiv](#_Toc78361392)

[DAFTAR LAMPIRAN xv](#_Toc78361393)

[BAB I PENDAHULUAN 1](#_Toc78361394)

[1.1 Latar Belakang 1](#_Toc78361395)

[1.2 Indentifikasi Masalah 4](#_Toc78361396)

[1.3 Batasan Masalah 5](#_Toc78361397)

[1.4 Tujuan dan Manfaat 5](#_Toc78361398)

[1.4.1 Tujuan 5](#_Toc78361399)

[1.4.2 Manfaat 6](#_Toc78361400)

[1.5 Metode dan Teknik Penelitian 7](#_Toc78361401)

[1.5.1 Metode Penelitian 7](#_Toc78361402)

[1.5.2 Teknik Penelitian 7](#_Toc78361403)

[1.6 Anggapan Dasar 8](#_Toc78361404)

[1.7 Kerangka Pemikiran 10](#_Toc78361405)

[1.8 Sistematika Penulisan 12](#_Toc78361406)

[1.9 Lokasi Penelitian dan Jadwal Penelitian 13](#_Toc78361407)

[1.9.1 Lokasi Penelitian 13](#_Toc78361408)

[1.9.2 Jadwal Penelitian 13](#_Toc78361409)

[BAB II LANDASAN TEORI 15](#_Toc78361410)

[2.1 Tinjauan Pustaka 15](#_Toc78361411)

[2.2 Teori Utama Penelitian 20](#_Toc78361412)

[*2.2.1* Kecerdasan Buatan (*Artificial Intelligence)* 20](#_Toc78361413)

[2.2.2 Local Binary Pattern Histogram (LBPH) 21](#_Toc78361414)

[2.2.3 OpenCV 26](#_Toc78361415)

[2.2.4 Computer Vision 26](#_Toc78361416)

[2.2.5 Python 27](#_Toc78361417)

[2.2.6 Amazon Web Service (AWS) 27](#_Toc78361418)

[2.3 Metode Pengembangan Perangkat Lunak 28](#_Toc78361419)

[2.3.1 Waterfall 28](#_Toc78361420)

[2.3.2 Tahapan-tahapan Waterfall 29](#_Toc78361421)

[2.4 Tools Perancangan 31](#_Toc78361422)

[2.4.1 Flowchart 31](#_Toc78361423)

[2.4.2 Flowmap 33](#_Toc78361424)

[2.4.3 Data Flow Diagram (DFD) 35](#_Toc78361425)

[2.5 Tools Perangkat Lunak 36](#_Toc78361426)

[2.5.1 Visual Studio Code 36](#_Toc78361427)

[2.5.2 AWS Lambda 36](#_Toc78361428)

[2.5.3 Amazon Simple Cloud Storage (S3) 37](#_Toc78361429)

[2.5.4 Amazon DynamoDB 38](#_Toc78361430)

[2.5.5 Amazon Elastic Compute Cloud (EC2) 38](#_Toc78361431)

[2.5.6 NodeJs 39](#_Toc78361432)

[2.5.7 React Js 39](#_Toc78361433)

[BAB III ANALISIS SISTEM 41](#_Toc78361434)

[3.1 Gambaran Organisasi 41](#_Toc78361435)

[3.1.1 Sejarah Singkat 41](#_Toc78361436)

[3.1.2 Visi dan Misi **Error! Bookmark not defined.**](#_Toc78361437)

[3.1.3 Struktur Organisasi 42](#_Toc78361438)

[3.1.4 Uraian Tugas **Error! Bookmark not defined.**](#_Toc78361439)

[3.2 Analisis Sistem Berjalan 44](#_Toc78361440)

[BAB IV PERANCANGAN SISTEM 50](#_Toc78361441)

[4.1 Desain Prosedur 50](#_Toc78361442)

[4.2 Desain Dokumen dan informasi 70](#_Toc78361443)

[4.3 Desain Aliran Data 72](#_Toc78361444)

[4.4 Desain Interface dan Struktur Menu 81](#_Toc78361445)

[4.5 Implementasi Sistem 85](#_Toc78361446)

[4.6 Pengujian 95](#_Toc78361447)

[BAB V KESIMPULAN DAN SARAN 106](#_Toc78361448)

[5.1 Kesimpulan 106](#_Toc78361449)

[5.2 Saran 106](#_Toc78361450)

[DAFTAR PUSTAKA 107](#_Toc78361451)

[LAMPIRAN-LAMPIRAN 110](#_Toc78361452)

# DAFTAR TABEL

[Tabel 1.1 Jadwal Penelitian **Error! Bookmark not defined.**](#_Toc74658413)

[Tabel 2. 1 Simbol Flowchart Program **Error! Bookmark not defined.**](#_Toc80722028)

[Tabel 2. 2 Simbol Flowchart Program **Error! Bookmark not defined.**](#_Toc80722029)

[Tabel 2. 3 Simbol Flowmap **Error! Bookmark not defined.**](#_Toc80722030)

[Tabel 2. 4 Simbol Data Flow Diagram (DFD) **Error! Bookmark not defined.**](#_Toc80722031)

[Tabel 4. 1 Tingkat Kecerahan Gambar 54](#_Toc81861080)

[Tabel 4. 2 Operator LBP 1 54](#_Toc81861081)

[Tabel 4. 3 Operator LBP 2 54](#_Toc81861082)

[Tabel 4. 4 Operator LBP 3 54](#_Toc81861083)

[Tabel 4. 5 Operator LBP 4 54](#_Toc81861084)

[Tabel 4. 6 Operator LBP 5 54](#_Toc81861085)

[Tabel 4. 7 Operator LBP 6 55](#_Toc81861086)

[Tabel 4. 8 Hasil Operator LBP 1 57](#_Toc81861087)

[Tabel 4. 9 Hasil Operator LBP 2 59](#_Toc81861088)

[Tabel 4. 10 Hasil Operator LBP 3 61](#_Toc81861089)

[Tabel 4. 11 Hasil Operator LBP 4 63](#_Toc81861090)

[Tabel 4. 12 Hasil Operator LBP 5 66](#_Toc81861091)

[Tabel 4. 13 Hasil Operator LBP 6 68](#_Toc81861092)

[Tabel 4. 14 Nilai LBP Dari Satu Bagian Gambar Wajah 68](#_Toc81861093)

[Tabel 4. 15 Skema Tabel Pengguna 79](#_Toc81861094)

[Tabel 4. 16 Skema Tabel Data Absensi 80](#_Toc81861095)

[Tabel 4. 17 Skema Model 81](#_Toc81861096)

[Tabel 4. 18 Sintaks Membuat Dataset 86](#_Toc81861097)

# DAFTAR GAMBAR

[Gambar 1. 1 Kerangka Pemikiran 11](#_Toc74658418)

[Gambar 2. 1 Operator LBP Asli 22](#_Toc74658419)

[Gambar 2. 2 Flowchart deteksi wajah 23](#_Toc74658420)

[Gambar 2. 3 Pembagian gambar 25](#_Toc74658421)

[Gambar 2. 4 Histogram LBPH 25](#_Toc74658422)

[Gambar 2. 5 Contoh gambar dalam dataset 26](#_Toc74658423)

[Gambar 2. 6 Model Pengembangan Waterfall (Pressman 2002) 29](#_Toc74658424)

[Gambar 4. 1 Hasil Deteksi Wajah 52](#_Toc81864654)

[Gambar 4. 2 Merubah Warna Gambar 53](#_Toc81864655)

[Gambar 4. 3 Pembagian Wilayah Wajah 53](#_Toc81864656)

[Gambar 4. 4 Diagram Konteks 72](file:///C:\Users\Home\Desktop\ta\ZxZX-10%20(AutoRecovered).docx#_Toc81864657)

[Gambar 4. 5 DFD Level 1 73](file:///C:\Users\Home\Desktop\ta\ZxZX-10%20(AutoRecovered).docx#_Toc81864658)

[Gambar 4. 6 Desain Halaman Login 82](#_Toc81864659)

[Gambar 4. 7 Desain Halaman Setting 83](#_Toc81864660)

[Gambar 4. 8 Desain Halaman Absensi 83](#_Toc81864661)

[Gambar 4. 9 Desain Halaman Laporan 84](#_Toc81864662)

[Gambar 4. 10 Struktur Menu Sistem 84](#_Toc81864663)

[Gambar 4. 11 Halaman Pendaftaran 103](#_Toc81864664)

[Gambar 4. 12 Halaman Login 104](#_Toc81864665)

[Gambar 4. 13 Tampilan Setting 104](#_Toc81864666)

[Gambar 4. 14 Halaman Absensi 105](#_Toc81864667)

# DAFTAR LAMPIRAN

# PENDAHULUAN

## Latar Belakang

Menurut UU. RI No. 88 Th. 2019, Kesehatan Kerja adalah upaya yang ditujukan untuk melindungi setiap orang yang berada di tempat kerja agar hidup sehat dan terbebas dari gangguan kesehatan serta pengaruh buruk yang diakibatkan dari pekerjaan. Berdasarkan undang-undang UU. RI No. 88 Th. 2019, bahwa untuk menghadirkan lingkungan kerja yang aman maka perlu dilakukan upaya Kesehatan Kerja.

Maret 2020 merupakan awal penyebaran virus Covid-19 di Indonesia. Kasus Covid-19 di Indonesia hingga saat ini masih sulit dibendung, menurut data dari *Our World In Data* per tanggal 27 April 2021kasus positif di negara kita mencapai 1,65 juta jiwa [1]. WHO memberikan pernyataan bahwa Covid-19 dapat menyebar dalam beberapa kemungkinan termasuk transmisi secara langsung, *airbone*, *droplets* (percikan), *formit, fekal-oral*, melalui darah, binatang ke manusia, dan ibu ke anak [2]. Untuk menghadapi penyebaran Covid-19 pemerintah mengeluarkan Kebijakan Permenkes Nomor 9 Tahun 2020 tentang Pembatasan Sosial Berskala Besar (PSBB). Kebijakan ini sangat memengaruhi berbagai kegiatan seperti kegiatan bekerja yang biasanya dilaksanakan di kantor sekarang harus dilaksanakan dirumah atau *work from home* (WFH).

*Work from home* yang sering disingkat dengan WFH memiliki arti bekerja dari rumah. *Work from home* digambarkan dengan kegiatan atau pekerjaan karyawan yang berada di luar kantor atau dengan kata lain bekerja dari rumah. Sistem kerja WFH memiliki fleksibilitas yang tinggi namun sulit dalam pengawasannya. Salah satu bentuk pengawasan terhadap karyawan adalah dengan menggunakan absensi. Saat ini sudah banyak teknologi yang dipakai untuk sistem absensi seperti absensi untuk memindai biometrik manusia. Teknologi biometrik berdasarkan ciri kodrati manusia yaitu ciri fisiologis dan ciri tingkah laku, seperti wajah, sidik jari, suara, telapak tangan, iris mata dan retina mata, DNA dan tanda tangan.

Wajah manusia mengungkapkan banyak informasi kepada siapa saja yang melihatnya. Wajah tidak hanya dapat memberi tahu tentang suasana hati, niat atau perhatian, tetapi juga dapat berfungsi sebagai identitas dari seseorang [3]. Tentu seseorang juga dapat dikenali selain dari wajah. Suara, bentuk tubuh, gaya berjalan, atau bahkan gaya pakaian dapat menjadi identitas dari seseorang. Tetapi dari semuanya, wajah manusia yang paling unik, manusia tidak mungkin berwajah serupa bahkan pada kasus anak kembar.

Pada era modern ini dimana teknologi berkembang dengan pesatnya, wajah digunakan sebagai bagian yang dapat dikenali oleh komputer. Pendeteksian wajah dan pengenalan wajah merupakan teknik yang digunakan untuk melakukan proses pengenalan wajah pada komputer. Saat ini sudah banyak teknologi yang digunakan untuk mendeteksi kehadiran yaitu teknologi absensi sidik jari dan juga wajah. Melalui teknologi ini seseorang dapat diketahui waktu kehadirannya pada suatu tempat dengan cara melakukan proses absen pada mesin absensi. Pada sistem yang memanfaatkan sidik jari, pengguna wajib menyentuh detektor sidik jari pada proses absensi. Namun untuk mesin absensi wajah, pengguna wajib melakukan absensi melalui proses perekam wajah dengan cara berdiri didepan detektor kamera.

Ada banyak metode untuk pengenalan wajah, diantaranya Local Binary Pattern (LBP) dan Local Binary Pattern Histogram (LBPH). Pada LBP ada yang disebut LBP Operator yang digunakan untuk menggambarkan nilai kontras suatu piksel dan piksel didekatnya. LBP asli mendefinisikan ukuran piksel 3 \* 3. Menggunakan nilai piksel median sebagai ambang, ini membandingkan dengan nilai abu-abu dari 8 piksel yang berdekatan. Jika nilai piksel yang didekatnya lebih besar atau sama dibandingkan dengan nilai piksel median, maka nilai posisi piksel diberi tanda 1, jika tidak diberi tanda (0) seperti pada Gambar 2.1.

Metode LBPH membagi citra wajah menjadi beberapa sel, dan setiap sel diberikan operasi LBP sehingga menghasilkan sebuah nilai biner. Dalam metode LBPH semua nilai yang dihasilkan oleh setiap sel akan disatukan menjadi sebuah histogram yang kemudian dapat dibaca dan dijadikan referensi wajah oleh mesin.

Dalam pengenalan wajah akurasi adalah hal penting yang harus diperhatikan. Menurut riset yang dilakukan oleh Qadrisa Mutiara Detila dan Eri Prasetyo Wibowo, LBPH merupakan metode pengenalan wajah paling akurat disbanding dengan metode *Eigenface* dan *Fisherface* dengan akurasi rata-rata 83% [4].

Berdasarkan uraian tersebut penerapan sistem pengenalan wajah menggunakan metode LBPH dapat menjadi solusi yang baik untuk membuat sistem absensi online. Oleh karena itu penulis membuat tugas akhir dengan judul “**METODE LOCAL BINARY PATTERN HISTOGRAM (LBPH) PENGENALAN WAJAH PADA SISTEM ABSENSI ONLINE KARYAWAN RADAR CIREBON**”. Dengan sistem absensi online yang dibuat diharapkan dapat membantu dan berguna bagi pihak yang membutuhkan.

## Indentifikasi Masalah

Dari latar belakang tersebut, penulis mengindentifikasikan masalah penelitian sebagai berikut:

Karyawan yang sedang bekerja diluar seperti wartawan tidak dapat melakukan absensi, karena alat detektor sidik jari berada di kantor.

Tidak menerapkan protokol kesehatan saat pandemi secara maksimal karena karyawan harus menempelkan jari di alat detektor sidik jari.

## Batasan Masalah

Dalam pengembangan sistem absen menggunakan pengenalan wajah penulis membuat batasan masalah agar tujuan yang dari sistem yang dibuat dapat tercapai. Adapun batasan masalahnya adalah sebagai berikut:

1. Sistem absen ini hanya mengambil data wajah dan lokasi pengguna.
2. Sistem tidak memperhitungkan jarak antara pengguna dan lokasi kantor, lokasi pengguna hanya akan disimpan di database.
3. Sistem ini tidak menghasilkan laporan apapun dalam bentuk pdf/csv seperti laporan absensi harian/bulanan, laporan tiap karyawan. Sistem hanya menyimpan data dalam *database*.
4. *Training* dataset dan proses pengenalan wajah dilakukan secara otomastis ketika sistem berjalan di Amazon Web Service.

## Tujuan dan Manfaat

### Tujuan

Tujuan utama dari penelitian ini adalah:

Membuat sistem absensi *online* yang mampu mengenali wajah sebagai media verifikasinya sehingga dapat membuat karyawan yang memiliki mobilitas tinggi dapat melakukan absensi dimanapun dan menerapkan protokol kesehatan dengan baik.

Membuat lingkungan yang menerapkan protokol kesehatan dengan baik, karena mengurangi interaksi antar karyawan.

### Manfaat

Setelah mengetahui tujuan penelitian tersebut di atas, maka diharapkan penelitian ini dapat dikembangkan dan diamalkan dengan baik. Maka manfaat penelitian antara lain, yaitu:

Bagi Penulis

Dapat memahami bagaimana dasar-dasar proses pengenalan wajah menggunakan bahasa pemrograman python dan menerapkan ke aplikasi *serverless* berbasis website menggunakan Amazon Web Service.

Bagi Lembaga Pendidikan

Manfaat penelitian ini bagi lembaga pendidikan diharapkan dapat menjadi bahan pembelajaran dan referensi bagi rekan-rekan yang akan melakukan penelitian lebih lanjut dengan topik yang berhubungan dengan judul penelitian di atas.

Bagi Perusahaan

Sistem absensi online yang dibagun dapat memudahkan pengguna ketika melakukan absensi di manapun secara *online* sehingga memungkinkan pengguna tidak perlu lagi ke kantor hanya untuk melakukan absensi.

## Metode dan Teknik Penelitian

### Metode Penelitian

Metode yang digunakan dalam penelitian adalah metode penelitian dan pengembangan (*Research and Development*). Menurut Sugiyono metode penelitian dan pengembangan adalah metode penelitian yang digunakan untuk menghasilkan produk dan mengujinya [5]. Agar dapat menghasilkan produk digunakan penelitian yang bersifat analisis kebutuhan dan untuk menguji keefektifan produk tersebut agar dapat berfungsi di masyarakat luas, maka diperlukan penelitian untuk menguji keefektifan produk tersebut.

Salah satu keuntungan yang merupakan alasan penulis memilih metode penelitian dan pengembangan (*Research and Development*) adalah mampu memenuhi dan mengatasi kebutuhan yang yang sifatnya mendesak melalui pengembangan solusi dari sebuah masalah sekaligus menghasilkan pengetahuan yang berguna bagi masa yang akan datang.

### Teknik Penelitian

Teknik penelitian yang digunakan penulis untuk perancangan dan pengembangan sistem diantaranya adalah:

Observasi

Observasi salah satu kegiatan ilmiah yang berdasarkan pada fakta-fakta lapangan maupun dokumen, melalui pengalaman sendiri tanpa menggunakan manipulasi apapun [6].

Dalam penelitian ini penulis melakukan observasi di Radar Cirebon guna mendapatkan data-data pendukung yang diperlukan dalam penelitian ini seperti bagaimana sistem absensi yang sedang berjalan.

Studi kepustakaan

Studi kepustakaan adalah suatu studi yang digunakan untuk mengumpulkan informasi dan data dengan bantuan berbagai macam material yang ada di perpustakaan seperti dokumen, buku, majalah, kisah-kisah sejarah, dan lainnya [7].

Dalam penelitian ini penulis melakukan studi kepustakaan dengan cara mencari berbagai informasi yang berkaitan dengan penelitian pada buku atau jurnal maupun mencari dokumentasi *software* melalui internet.

## Anggapan Dasar

Penulis melakukan studi kepustakaan guna mengumpulkan teori-teori dari buku maupun penelitian terdahulu. Teori-teori tersebut dinamakan anggapan dasar dari sebuah penelitian, anggaran dasar adalah sesuatu yang sudah jelas kebenarannya.

Pengenalan wajah merupakan salah satu ilmu yang berhubungan dengan *computer vision*, dimana komputer menganalisis citra wajah yang ada pada citra tersebut dan membandingkannya dengan data pada citra wajah tersebut, dimana komputer dapat menemukan identitas atau data pribadi dari gambar wajah dalam *database*. Biasanya, pengenalan wajah dilakukan dari depan dengan menerangi 10 wajah secara merata. Namun, ada beberapa masalah seperti posisi wajah, ukuran wajah dan jarak, orientasi, usia dan ekspresi wajah.

Menurut Vicki Bruce dan Andy Young wajah manusia mengungkapkan banyak informasi kepada siapa saja yang melihatnya. Wajah tidak hanya dapat memberi tahu tentang suasana hati, niat atau perhatian, tetapi juga dapat berfungsi sebagai identitas dari seseorang. Tentu seseorang juga dapat dikenali selain dari wajah. Suara, bentuk tubuh, gaya berjalan, atau bahkan gaya pakaian dapat menjadi identitas dari seseorang. Tetapi dari semuanya, wajah manusia yang paling unik, manusia tidak mungkin berwajah serupa bahkan pada kasus anak kembar [3].

Dalam buku yang berjudul “*Handbook of Face Recognition”* yang ditulis oleh Stan Z. Li Anil K. Jain mengemukakan bahwa pengenalan wajah atau *face recognition* adalah tugas yang dilakukan manusia secara rutin dan mudah dalam kehidupan sehari-hari. Ketersediaan kumputasi komputer yang luas dan mudah memunculkan minat yang besar untuk pemrosesan otomatis gambar dan video digital di sejumlah aplikasi termasuk otentikasi biometrik, pengawasan, interaksi manusia dengan komputer dan lainnya [8].

## Kerangka Pemikiran

Pada Gambar 1.1 memaparkan pemikiran penulis secara kasar mengenai penelitian yang dilakukan untuk membuat aplikasi absensi *online* menggunakan pengenalan wajah. Sistem ini diharapkan dapat mengenali objek wajah secara akurat. Objek gambar tersebut di ambil dan direkam menjadi sebuah citra/gambar. Metode Local Binary Patterm Histogram(LBPH)diterapkan untuk mengambil tekstur dari objek gambar. Hasil pengambilan tekstur dari LBPHadalah sebuah histogram yang terdiri dari gabungan beberapa histogram kecil yang berisi informasi nilai operasi operator Local Binary Pattern(LBP*)*.



Gambar 1. 1 Kerangka Pemikiran

Tahapan-tahapan pembangunan atau pengembangan perangkat lunak (*software*) sebagai berikut:

* + - 1. Menentukan perangkat lunak yang akan digunakan.
      2. Membuat desain tampilan program.
      3. Membuat kode program.
      4. Melakukan pengujian dan pemeliharaan.

Setelah aplikasi selesai dibuat maka dilakukan pengujian dengan cara pengguna mencoba aplikasi secara langsung. Hasil dari penelitian ini berupa *website* absensi *online* berbasis pengenalan wajah.

## Sistematika Penulisan

Sistematika penulisan merupakan langkah-langkah yang digunakan untuk menyusun laporan penelitian, penulis menggunakan sistematika sebagai berikut:

**BAB I PENDAHULUAN**

Bab ini merupakan pendahuluan yang berisi latar belakang, identifikasi masalah, batasan masalah, tujuan dan manfaat penelitian, metode dan teknik penelitian, kerangka pemikiran, anggapan dasar, sistematika penulisan serta lokasi penelitian dan jadwal penelitian.

**BAB II LANDASAN TEORI**

Dalam BAB II akan dijelaskan mengenai landasan teori sebagai acuan primer yang digunakan penulis dalam penelitian ini, teori pendukung serta tinjauan pustaka yang ditambahkan untuk menjelaskan informasi secara lebih detail.

**BAB III ANALISIS SISTEM**

Dalam BAB III membahas tentang gambaran organisasi serta analisis dari sistem yang sedang berjalan, baik dalam bentuk uraian, diagram atau keduanya.

**BAB IV PERANCANGAN SISTEM**

Dalam BAB IV akan dijelaskan tentang prosedur yang akan berjalan untuk sistem yang baru, desain *interface*, implementasi maupun pengujian pada sistem tersebut.

**BAB V KESIMPULAN DAN SARAN**

Dalam BAB V akan dijelaskan tentang kesimpulan dan saran-saran dari penulis tentang penulisan skripsi ini.

**DAFTAR PUSTAKA**

**LAMPIRAN**

## Lokasi Penelitian dan Jadwal Penelitian

### Lokasi Penelitian

Penelitian ini di lakukan di Radar Cirebon yang beralamat di Grha Pena Radar Cirebon, Jl. Perjuangan No.9 Kota Cirebon 45135 Telp. (0231) 483531, 483532 Fax: (0231) 483533, Indonesia dengan jadwal penelitian dari Maret – Agustus 2021.

### Jadwal Penelitian

Pada Tabel 1.1 menjelaskan waktu penelitian yang dilakukan penulis yaitu dari bulan Maret 2021 hingga bulan Agustus 2021.

| Tabel 1.1 Jadwal Penelitian | | | | | | | | | | | | | | | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Prosedur Penelitian | 2021 | | | | | | | | | | | | | | | | | | | | | | | |
| Maret | | | | April | | | | Mei | | | | Juni | | | | Juli | | | | Agustus | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 1 | Persiapan |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | Analisis masalah dan pengumpulan data |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | Penyusunan BAB I - BAB III |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | Pembuatan aplikasi dan menyusun BAB IV - BAB V |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 | Evaluasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# LANDASAN TEORI

## Tinjauan Pustaka

Dalam penelitian ini penulis menggunakan beberapa penelitian terdahulu yang membahas pengenalan wajah dengan metode Local Binary Pattern(LBP)atau Local Binary Pattern Histogram (LBPH). Untuk menambahkan gambaran penelitian penulis dibandingkan dengan penelitian yang sudah dilakukan, maka dibawah ini diuraikan tentang penelitian terdahulu yang relevan. Beberapa penelitian yang berkaitan dengan masalah penelitian ini adalah:

**PERBANDINGAN METODE EIGENFACE, FISHERFACE, DAN LBPH PADA SISTEM PENGENALAN WAJAH.**

Qadrisa Mutiara Detila dan Eri Prasetyo Wibowo, Jurnal Ilmiah KOMPUTASI, Volume 18 No. 4, Desember 2019. Penelitian ini menggunakan *dataset* yang diperoleh dari pengambilan citra menggunakan *webcam* dan *Haar cascade classifier* dalam bentuk xml file untuk deteksi wajah dan proses pengenalan wajah. Tahap awal dalam sistem pengenalan wajah adalah proses deteksi wajah, citra yang diperlukan dalam penelitian diambil dari kamera yang menghasilkan citra dinamis kemudian di konversi menjadi citra statis, lalu disimpan ke dalam folder *dataset* untuk dijadikan citra *training*.

Qadrisa Mudiara Detila dkk. melakukan perbandingan dengan cara membandingkan gambar wajah dengan 4 kondisi yang berbeda yaitu tingkat kecerahan rendah, tingkat kecerahan normal, wajah tidak berekspresi, dan wajah tersenyum. Hasil dari jurnal ini adalah nilai persentase keberhasilan dari 4 kondisi yang ditentukan sebelumnya dengan rata-rata 46% untuk *Eigenface*, 54,25% untuk *Fisherface* dan 88,75% untuk LBPH. Dilihat dari nilai persentase keberhasilan dari tiap kondisi yang ditentukan, Metode LBPH dapat mengenali wajah lebih baik dibandingkan metode *Eigenface* dan *Fisherface*. Berdasarkan hasil tersebut, metode LBPH juga lebih unggul dibandingkan dengan kedua metode lainnya dalam mengenali wajah yang datar atau tanpa ekspresi. Sedangkan untuk ekspresi senyum, metode *Eigenface* masih sulit untuk mengenali wajah dibandingkan kedua metode lainnya yaitu *Fisherface* dan LBPH yang masih bisa mengenali wajah apabila terjadi perubahan ekspresi.

**SISTEM PENGENALAN WAJAH DENGAN METODE LOCAL BINARY PATTERN HISTOGRAM PADA FIREBASE BERBASIS OPENCV**.

Fajar Setiawan dan Dewi Agushinta, Seminar Nasional Teknologi Informasi dan Komunikasi STI&K (SeNTIK) Volume 4 Nomor 1, 23 September 2020. Penelitian ini menggunakan LBPH sebagai metode untuk pengolahan fitur wajah dan *Haar Cascade* untuk pengambilan frame wajah. Metode LBPH mengubah tekstur dari suatu citra wajah menjadi nilai biner, dan nilai tersebut mewakili bagian dari piksel-piksel suatu wajah yang membentuk sebuah lingkaran dan memiliki pusat sebagai acuan terhadap nilai-nilai tersebut. *Database* yang digunakan adalah *database* *online* dari firebase yang dinaungi oleh Google, sehingga dengan terciptanya *dataset* wajah secara online, pengoperasian pengenalan wajah dapat lebih efisien.

Fajar Setiawan dkk. melakukan penelitian untuk mengukur tingkat keberhasilam metode LBPH dalam mengenali wajah. Hasil yang diperoleh dari penelitian ini sistem pengenalan wajah yang memiliki tingkat keberhasilan 93,5%. Hasil tersebut diperoleh dengan cara menguji sistem untuk mengenali 62 citra wajah.

**FACIAL RECOGNITION SYSTEM USING LBPH FACE RECOGNIZER FOR ANTI-THEFT AND SURVEILLANCE APPLICATION BASED ON DRONE TECHNOLOGY.**

Li Wang dan Ali Akbar Siddique, *Original Paper Measurement and Control* (*United Kingdom*), 15 Mei 2020. Penelitian ini mengusulkan sistem pengenalan wajah menggunakan Local Binary Patterns Histogram (LBPH) *Face recognition* yang dipasang pada teknologi *drone*. Selama bertahun-tahun, para peneliti telah menggunakan algoritma yang berbeda untuk pengenalan wajah seperti *Haar cascade*, fitur TPLBP/HOG, *DBN Depth Model* dan *Fisherface* untuk keamanan mesin ATM atau untuk autentikasi seseorang. Peneliti juga menggunakan *neural network* bersama dengan *sparse auto-encoder* untuk melatih model pengenalan wajah.

Pengenal wajah LBPH juga dapat melacak dan mengidentifikasi tindakan manusia beserta fitur dominan dan klasifikasinya. Makalah ini mengusulkan sistem tertanam berbasis teknologi *drone*, yang mampu mengidentifikasi seseorang menggunakan pengenal wajah LBPH. Kamera yang terpasang pada drone menangkap bingkai secara *real time* dan mengidentifikasi orang yang menjadi target. Kecanggihan Bagian ini menawarkan beberapa penelitian mutakhir tentang teknologi pengenalan wajah menggunakan *drone*. Studi ini berkonsentrasi pada pengenalan wajah orang-orang dengan latar belakang kriminal. Jika drone mengidentifikasi orang yang menjadi target, *drone* akan menginformasikan otoritas terkait dengan mengirimkan gambar yang menampilkan orang yang diidentifikasi. Karena, model yang diusulkan menggunakan pengenalan wajah algoritma pembelajaran mesin (LBPH), ini juga merupakan aplikasi berbasis *AI*. Model perangkat lunak Aplikasi berbasis *Machine learning* seperti pengenalan wajah memerlukan kumpulan data yang sangat besar untuk pelatihan sebelum kami mengimplementasikannya. Hasil dari penelitian tersebut adalah sistem pengenalan wajah yang memiliki tinggat akurasi sebesar 89,1%.

**LBPH BASED IMPROVED FACE RECOGNITION AT LOW RESOLUTION.**

Aftab Ahmed, Jiandong Guon Fayaz Ali, Farha Deeba, dan Awais Ahmed, *International Conference on Artificial Intelligence and Big Data, 2018.* Penelitian ini menggunakan metode algoritma Local Binary Patterns Histogram (LBPH) untuk mengatasi pengenalan wajah manusia secara *real time* pada tingkat resolusi yang rendah. Ekstraksi Fitur Operator LBP diterapkan untuk menggambarkan informasi kontras suatu piksel ke piksel tetangganya lalu mengubahnya menjadi angka desimal sehingga diperoleh nilai LBP dari titik piksel tengah. Algoritma LBPH saat ini menggunakan operator LBP melingkar yang ditingkatkan. Algoritma ini membuat operator LBP tidak lagi terbatas pada radius dan lingkungan tetap dan dapat memenuhi kebutuhan fitur ukuran dan tekstur yang lebih berbeda. Untuk setiap piksel gambar, ia menghitung nilai *eigen* LBP-nya kemudian nilai *eigen* ini dapat membentuk spektrum fitur LBP. Algoritme LBPH menggunakan histogram spektrum karakteristik LBP sebagai vektor fitur untuk klasifikasi dan membagi gambar menjadi beberapa *sub-region*, kemudian mengekstrak fitur LBP dari setiap piksel *sub-region*, membentuk histogram statistik spektrum karakteristik LBP di setiap *sub-region*, sehingga setiap *sub-region* dapat menggunakan histogram statistik untuk gambarkan keseluruhan gambar melalui sejumlah komponen histogram.

Penelitian ini menggunakan *dataset* LR500 dan menguji dengan 2500 gambar yang diambil oleh *webcam* beresolusi 1080px. Dari 2500 gambar 98.8% berhasil dikenali. Untuk hasil akurasi ukuran wajah 45px sebesar 94% dan 35px sebesar 90%.

**SISTEM PENGENALAN WAJAH DENGAN ALGORITMA HAAR CASCADE DAN LOCAL BINARY PATTERN HISTOGRAM.**

Sayeed Al-Aidid dan Daniel S. Pamungkas, Jurnal Rekayasa Elektrika, Volume 1 No.1, April 2018. Penelitian ini mengunakan webcam dan mampu membedakan objek wajah dan bukan wajah, serta mengenai orang yang telah terdaftar pada database secara *real time*. Sayeed Al-Aidid dan Daniel S. Pamungkas menggunakan *Haar cascade classifier* dan metode Local Binary Pattern Histogram (LBPH) untuk mengenali citra wajah. Metode LBPH digunakan karena mampu mendeteksi wajah dari sebuah cirta gambar meskipun di sekitar wajah manusia tertapat gangguan objek. Ada beberapa tahapan yang dilakukan peneliti untuk membuat sistem pengenalan wajah, diantaranya deteksi wajah, pembuatan *database,* dan pengenalan wajah. Dalam penelitian ini juga dibahas seberapa jauh sistem dapat mengenali wajah. Hasil dari penelitian Sayeed Al-Aidid dkk. adalah sistem pengenalan wajah dapat mendeteksi wajah meskipun terdapat gangguan objek lain dan dapat mendeteksi wajah paling jauh sekitar 150cm.

## Teori Utama Penelitian

### Kecerdasan Buatan (Artificial Intelligence)

Kecerdasan buatan adalah cabang ilmu komputer yang menjelaskan bagaimana komputer dapat meniru pemikiran manusia. Kecerdasan buatan memungkinkan komputer untuk menarik kesimpulan dan mengidentifikasi masalah dengan cara manusia. Dalam buku yang ditulis oleh Dr. Hendra Jaya, S.Pd., M.T., dkk. Dengan judul “Kecerdasan Buatan” mengemukakan bahwa kecerdasan buatan merupakan salah satu bidang ilmu komputer yang membuat mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik apa yang manusia lakukan [9].

Menurut Dr. Hendra Jaya, S.Pd., M.T., dkk. Kecerdasan buatan dapat dipandang dalam 4 sudut pandang yaitu sudut pandang kecerdasan, penelitian, bisnis, dan pemrograman. Berikut beberapa contoh penerapan kecerdasan buatan (*Artificial Intelligence),* diantaranya:

Sistem kemudi otomatis pada mobil, contohnya TESLA pabrikan mobil dari Amerika Serikat memproduksi mobil *autopilot*.

Asisten virtual seperti Siri pada Apple, Bixbi pada Samsung, dan Alexa pada Amazon.

Sistem rekomendasi, seperti rekomendasi produk di *e-commerce* dan rekomendasi iklan pada youtube.

### Local Binary Pattern Histogram (LBPH)

Untuk memahapi apa itu Local Binary Pattern Histogram (LBPH) kita harus memahami dahulu apa itu Local Binary Pattern (LBP) karena LBPH merupakan gabungan nilai yang dihasilkan LBP operator dari sebuah citra gambar. Operator LBP diterapkan untuk menggambarkan kontras informasi dari suatu piksel ke piksel tetangganya. Operator LBP asli didefinisikan di jendela 3\*3. Menggunakan nilai piksel median sebagai ambang jendela, dibandingkan dengan nilai abu-abu dari 8 piksel yang berdekatan. Jika nilai piksel tetangga lebih besar atau sama dengan nilai median piksel, nilai posisi piksel ditandai sebagai 1, sebaliknya ditandai sebagai (0) [9]. Fungsi didefinisikan seperti yang ditunjukkan pada persamaan 1. Dapat diilustrasikan seperti pada Gambar 2.1.



Gambar 2. 1 Operator LBP Asli

Sumber: Ahmed et al (2018)

Setelah diubah menjadi nilai biner maka akan menghasilkan deret biner yang kemudian jika diubah menjadi nilai desimal akan menghasilkan nilai baru untuk nilai tengah piksel selanjutnya, deret biner tersebut yang disebut dengan kode LBP. Kode LBP yang dihasilkan tersebut direpresentasikan dengan histogram. Histogram akan menunjukkan frekuensi kejadian dari berbagai nilai LBP. Secara umum perhitungan LBP ini bisa dilakukan seperti persamaan 1.1, hasil dari persamaan 1.1 kemudian dihitung menggunakan persamaan dua sehingga hasil akhirnya angka nol atau satu yang dapat dilihat pada persamaan 1.2 [10]:

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) |

Pada persamaan 1.1 menjelaskan operator LBP asli dengan adalah nilai piksel dari titk pusat (), sedangkan merupakan nilai piksel tetangga dari . Persamaan 1.2 dengan x merupakan nilai LBP. Berikut adalah alur pengenalan wajah menggunakan metode LBPH [11]:

Deteksi Wajah (*Face Detection)*

Untuk deteksi wajah, penulis menggunakan OpenCV dan *Haar cascade classifier.*



Gambar 2. 2 Flowchart deteksi wajah

Sumber: (A.Singh et al 2019)

Pada Gambar 2.2 menjelaskan tentang alur dari deteksi wajah. Langkah pertama yaitu menbaca gambar kemudian dirubah ke gambar abu-abu, setelah itu lakukan deteksi wajah menggunakan *Haar cascade classifier* hingga wajah ditemukan dan disimpan menjadi gambar wajah.

Ekstraksi Fitur (*Feature Extraction*)

Untuk mengekstrak fitur wajah dari citra digunakan operasi LBP yang membandingkan nilai intensitas setiap komponen dengan nilai 8 piksel tetangga terdekat. Jika nilai piksel tetangga lebih besar dari nilai piksel tengah, itu akan menetapkan 1 ke piksel tetangganya, jika tidak maka akan menetapkan 0. Untuk setiap piksel, tugas ini menyediakan string 8-bit. Nilai desimal dari string piksel 8-bit menentukan nilai LBP. Gambar 2.1 menunjukkan operasi ini.

Gambar input dibagi menjadi banyak sub-gambar kecil seperti pada Gambar 2.3. Setelah penerapan operator LBP dan histogram nilai LBP dari setiap sub-gambar diekstraksi. Kemudian semua histogram dihubungkan seperti pada Gambar 2.4 untuk membuat vektor fitur yang merepresentasikan citra dan digunakan untuk melatih pengklasifikasi pengenalan wajah.



Gambar 2. 3 Pembagian Gambar LBP

Sumber: (Li Wang dan Ali Akbar Siddique, 2020)



Gambar 2. 4 Histogram LBPH

Sumber: (Li Wang dan Ali Akbar Siddique, 2020)

Membuat Dataset

Penulis merancang sendiri *dataset* yang akan dibuat, *dataset* berisi 3 gambar untuk setiap wajah. *Dataset* akan dilatih setiap ada pengguna yang mendaftarkan wajahnya ke sistem absen ini. Gambar 2.5 menunjukan contoh gambar dalam *dataset*.



Gambar 2. 5 Contoh gambar dalam dataset

Sumber: (A.Singh et al, 2019)

Pengenalan Wajah (*Face Recognition*)

Dalam langkah ini gambar akan diproses menggunakan operator LBP seperti pada Gambar 2.1. Setelah perhitungan menggunakan operator LBP selesai dan menghasilkan Histogram, maka akan dibandingkan dengan *dataset* yang sudah ada.

### OpenCV

OpenCV adalah kependekan dari *opensource computer vision.* OpenCV adalah sebuah kumpulan *tools, library,* dan *module* yang berkaitan dengan *computer vision*. Dalam OpenCV terdapat kumpulan algoritma yang dinamakan “*Haar Cascade Classifier*” untuk mendeteksi objek khususnya wajah manusia. Dengan menggunakan *Haar Cascade Classifier* penulis tidak kesulitan untuk mendeteksi wajah dalam sebuah gambar atau video.

### Computer Vision

*Computer Vision* adalah bagian dari komputer yang menggambarkan bagaimana komputer terlihat layaknya manusia, dan karena ini terkait erat dengan penglihatan, cahaya juga merupakan faktor penting dalam hal ini. [12].

### Python

Python adalah bahasa pemrograman berorientasi objek yang ditafsirkan, interaktif. Ini menggabungkan modul, pengecualian, pengetikan dinamis, tipe data dinamis tingkat sangat tinggi, dan kelas. Ini mendukung beberapa paradigma pemrograman di luar pemrograman berorientasi objek, seperti pemrograman prosedural dan fungsional. Python menggabungkan kekuatan yang luar biasa dengan sintaks yang sangat jelas. Ini memiliki antarmuka ke banyak panggilan sistem dan perpustakaan, serta berbagai sistem jendela, dan dapat dikembangkan dalam C atau C ++. Ini juga dapat digunakan sebagai bahasa ekstensi untuk aplikasi yang membutuhkan *programmable* antarmuka. Terakhir, Python bersifat portable, ia berjalan di banyak varian Unix termasuk Linux dan macOS, dan di Windows.

### Amazon Web Service (AWS)

Ada banyak definisi dan interpretasi komputasi awan yang dapat ditemukan dari berbagai sumber. Istilah "komputasi awan" itu sendiri kemungkinan berasal dari diagram jaringan di mana *cloud* bentuk digunakan untuk menggambarkan jenis jaringan tertentu, baik internet atau jaringan internal. Beberapa sumber menyebut komputasi awan sebagai sekumpulan aplikasi yang dikirimkan sebagai gabungan layanan dengan perangkat keras dan perangkat lunak pusat data yang memungkinkan aplikasi. Yang lain mengatakan awan itu komputasi adalah model bisnis daripada teknologi atau layanan tertentu [13].

Amazon Web Service (AWS) adalah penyedia layanan *cloud computing*. Menurut situs resminya yaitu <https://aws.amazon.com/>, AWS menyediakan lebih dari 200 service seperti penyimpanan, *compute, internet of things, machine learning,* dan masih banyak lainnya.

## Metode Pengembangan Perangkat Lunak

### Waterfall

Pengembangan perangkat lunak meliputi metode, proses, dan alat yang memudahkan dalam proses pengembangan perangkat lunak dari tahap satu ke tahap lainnya agar menghasilkan perangkat lunak yang berkualitas [14].

Model pengembangan perangkat lunak merupakan gambaran dari proses membuat suatu *software.* Menurut Pressman [14] Model Waterfall merupakan suatu model pengembangan secara sekuensial. Model Waterfall bersifat sistematis dan berurutan dalam membangun sebuah perangkat lunak. Proses pembuatannya mengikuti alur dari mulai analisis, desain, kode, pengujian dan pemeliharaan. Model pengembangan waterfall memiliki beberapa kelebihan, antara lain: dapat mudah dipahami dan dapat diterapkan dalam proses pengembangan perangkat lunak.Gambar 2.6. merupakan model pengembangan Waterfall.



Gambar 2. 6 Model Pengembangan Waterfall (Pressman 2002)

Sumber: (Pressman, 2002)

### Tahapan-tahapan Waterfall

Tahapan-tahapan dalam metode waterfall adalah sebagai berikut [14]:

Analisis kebutuhan perangkat lunak.

Tahap ini merupakan tahap untuk mengumpulkan kebutuhan perangkat lunak. Tujuan dari tahap ini yaitu untuk mengetahui apa saja kebutuhan untuk membuat sebuah perangkat lunak atau *software.* Analisis kebutuhan pertangkat lunak ini perlu didokumentasikanuntuk dilihat kembali oleh pengguna apakah kebutuhan sudah sesuai atau tidak.

Desain

Desain merupakan tahapan yang berfokus pada perangkat lunak yang meliputi arsitektur, struktur data, antarmuka perangkat lunak, dan prosedur perngkodean. Ada beberapa pemodelan perangkat lunak yang umum digunakan, salah satu pemodelan perangkat lunak yang digunakan dalam tahapan ini yaitu Unified Modeling Language (UML), yang merupakan gambaran mengenai perangkat lunak yang akan dibuat. UML dibuat untuk mempermudah pengembang dalam membuat suatu perangkat lunak.

Pembuatan Kode

Dalam tahap ini desain yang sebelumnya telah dibuat akan diterapkan kedalam sebuah kode. Penerjemahan desain ke dalam kode haruslah lengkap dan berjalan dengan baik, Hasil dari tahap ini adalah sebuah sistem informasi yang sesuai dengan desain yang telah dibuat.

Pengujian

Tahapan biasanya ini merupakan tahapan yang paling panjang. Sistem dipasang dan digunakan secara nyata. pengujian melibatkan pembetulan kesalahan-kesalahan yang tidak ditemukan pada tahapan sebelumnya, meningkatkan implementasi dari unit sistem, dan meningkatkan layanan sistem sebagai kebutuhan baru.

## Tools Perancangan

### Flowchart

*Flowchart* menurut Drs. Suarga, M.Sc, Math, Ph.D, adalah untaian simbol atau gambar yang menunjukkan aliran dari data. Dalam bukunya juga yang berjudul ALGORITMA (LOGIKA PEMROGRAMAN), Drs. Suarga, M.Sc, Math, Ph.D, membagi symbol *flowchart* menjadi 2 jenis [15], yaitu:

* + - 1. Simbol *flowchart* program

*Flowchart* Program merupakan simbol-simbol *flowchart* yang digunakan untuk menggambarkan logika dari pemrosesan terhadap data ditunjukan pada Tabel 2.1.

| Tabel 2. 1 Simbol Flowchart Program | | |
| --- | --- | --- |
| **Simbol** | **Nama Simbol** | **Keterangan** |
|  | terminator | Menunjukan awal/akhir sistem |
|  | proses | Menunjukan operasi yang dilakukan sistem |
|  | *Input/output* | Menerima input atau menampilkan output |
|  | pilihan | Memilih aliran berdasarkan syarat |
|  | *Predefined process* | Menunjukan fungsi atau *sub-program* |
|  | penghubung | Menunjukan penghubung pada halaman yang sama |
|  | penghubung | Menunjukan penghubung pada halaman yang lain |

Sumber: (Suarga, 2002)

* + - 1. Simbol *flowchart* sistem

*Flowchart* sistem merupakan simbol-simbol peralatan sistem komputer yang digunakan untuk menyatakan proses pengolahan data ditunjukan pada Tabel 2.2.

| Tabel 2. 2 Simbol Flowchart Program | |
| --- | --- |
| **Simbol** | **Nama Simbol** |
|  | *keyboard* |
|  | *printer* |
|  | *File/storage* |
|  | *Display* |
|  | *Magnetic tape* |
|  | *Magnetic disk* |
|  | *sorting* |
|  | *extract* |
|  | *merge* |

Sumber: (Suarga, 2002)

### Flowmap

*Flowmap* adalah campuran peta dan *flowchart* yang menunjukkan pergerakan benda dari satu lokasi ke lokasi lain [16]. *Flowmap* sendiri sering disebut diagram alir atau aliran data yang berbentuk dokumen dalam sebuah aktifitas yang saling berkaitan antara kebutuhan data dan informasi. *Flowmap* dapat membantu programmer untuk memecahkan masalah yang besar ke dalam segmen atau bagian yang lebih kecil.

Fungsi *flowmap* adalah mendefinisikan hubungan antara bagian (pelaku proses), proses (manual/berbasis komputer) dan aliran data (dalam bentuk dokumen masukan dan keluaran) [16]. Simbol Flowmap dalam Tabel 2.3.

| Tabel 2. 3 Simbol Flowmap | |
| --- | --- |
| **Simbol** | **Keterangan** |
|  | *Terminator Symbol* (Simbol yang menunjukan awal/akhir dari sistem) |
|  | *Process* *Symbol* (Simbol yang menunjukan pengolahan yang dilakukan oleh komputer) |
|  | Simbol *Document* (Simbol yang menyatakan input berasal dari dokumen dalam bentuk kertas atau output dicetak kekertas) |
|  | *Decision Symbol* (Simbol untuk kondisi yang akan menghasilkan beberapa kemungkinan jawaban atau aksi) |
|  | *Predefined process* Symbol (Simbol untuk menjelaskan proses dalam *chart* tersendiri) |
|  | *Connector Symbol* (Simbol untuk keluar atau masuk prosedur atau proses dalam lembar atau halaman) |
|  | *Off-line Connector* *Symbol* (Simbol untuk keluar atau masuk proses pada halaman yang lain) |
|  | *Input/output Symbol* (Simbol yang dapat menerima input atau menampilkan output) |

Sumber: (Haidar Bagir1 & Bramantiyo Eko Putro, 2018)

### Data Flow Diagram (DFD)

Pada akhir 1970an diagram aliran data (DFD) diperkenalkan dan dipopulerkan untuk analisis dan desain terstruktur (Gane dan Sarson 1979). DFD menunjukkan aliran data dari entitas eksternal ke dalam sistem, menunjukkan bagaimana data dipindahkan dari satu proses ke proses lainnya, serta penyimpanan logisnya [17].

Diagram konteks merupakan level tertinggi dari DFD yang menggambarkan seluruh input ke sistem atau output dari sistem. Sistem dibatasi oleh boundary. Dalam diagram konteks hanya ada satu proses. Tidak boleh ada *store* dalam diagram konteks [16]. Diagram konteks terdiri dari beberapa simbol [16], yang ditunjukan pada Tabel 2.4

| Tabel 2. 4 Simbol Data Flow Diagram (DFD) | |
| --- | --- |
| **Simbol** | **Keterangan** |
|  | *External entity* adalah sumber atau tujuan aliran data suatu sistem |
|  | *Data store* adalah tempat penyimpanan data |
|  | *Process,* menggambarkan bagaimana *input* diubah menjadi *output* |
|  | *Boundary,* merupakan Batasan dari sebuah sistem |
|  | Aliran data, menggambarkan aliran dari suatu proses ke proses lainnya |

Sumber: (Suarga, 2002)

## Tools Perangkat Lunak

### Visual Studio Code

Visual Studio Code adalah *text editor* yang dikembangkan oleh Microsoft yang dapat berjalan di Windows, Linux dan MacOS. Ini termasuk dukungan untuk *debugging*, GIT *Control* yang disematkan, penyorotan sintaks, penyelesaian kode cerdas, cuplikan, dan kode *refactoring*. Hal ini juga dapat disesuaikan, sehingga pengguna dapat mengubah tema *editor*, *shortcut keyboard*, dan preferensi. Visual Studio Code gratis dan *opensource*, meskipun unduhan resmi berada di bawah lisensi MIT.

### AWS Lambda

AWS Lambda atau biasa disebut Lambda merupakan layanan dari Amazon Web Service yang memungkinkan kita menjalankan kode program tanpa harus memikirkan infrastruktur dari server seperti tidak perlu memilih Sistem Operasi. Lambda berjalan oleh sebuah *trigger,* *trigger* tersebut dapat berupa *event* dari *endpoint, event* dari S3 *bucket,* maupun dari sumber lainnya.

### Amazon Simple Cloud Storage (S3)

Amazon S3 atau biasa disebut S3 adalah kependekan dari *Simple Cloud Storage* yang merupakan layanan penyimpanan dari Amazon Web Service. Amazon S3 memiliki kelebihan yaitu memiliki ketahanan yang tinggi, sangat mudah digunakan dan memiliki penyimpanan dan *bandwidth* yang tidak terbatas [18].

Amazon Simple Storage Service (Amazon S3) adalah layanan penyimpanan objek yang menawarkan skalabilitas, ketersediaan data, keamanan, dan kinerja terdepan di industri. Ini berarti pelanggan dari semua ukuran dan industri dapat menggunakannya untuk menyimpan dan melindungi data dalam jumlah berapa pun untuk berbagai kasus penggunaan, seperti data lake, situs web, aplikasi seluler, pencadangan dan pemulihan, arsip, aplikasi perusahaan, perangkat IoT, dan besar analitik data. Amazon S3 menyediakan fitur manajemen yang mudah digunakan sehingga Anda dapat mengatur data dan mengonfigurasi kontrol akses yang disesuaikan untuk memenuhi persyaratan bisnis, organisasi, dan kepatuhan spesifik Anda. Amazon S3 dirancang untuk ketahanan 99,999999999% (11 9), dan menyimpan data untuk jutaan aplikasi untuk perusahaan di seluruh dunia.

### Amazon DynamoDB

Amazon DynamoDB atau biasa disebut Dynamodb merupakan layanan penyimpanan data milik AWS. DynamoDB merupakan database NoSql yang memiliki ketahanan yang tinggi. Amazon DynamoDB adalah database nilai kunci dan dokumen yang memberikan kinerja milidetik satu digit pada skala apa pun. Ini adalah database yang terkelola sepenuhnya, multi-wilayah, multi-aktif, dan tahan lama dengan keamanan bawaan, pencadangan dan pemulihan, dan penyimpanan dalam memori untuk aplikasi skala internet. DynamoDB dapat menangani lebih dari 10 triliun permintaan per hari dan dapat mendukung puncak lebih dari 20 juta permintaan per detik.

### Amazon Elastic Compute Cloud (EC2)

Amazon EC2 atau biasa disebut EC2 adalah kependekan dari *Elastic Compute Cloud ­*yang merupakan layanan dari Amazon Web Service yang memungkinkan kita memiliki paket komputer lengkap dalam *cloud* dalam hitungan detik [18].

Amazon Elastic Compute Cloud (Amazon EC2) adalah layanan web yang menyediakan kapasitas komputasi yang aman dan dapat diubah ukurannya di cloud. Ini dirancang untuk membuat komputasi awan skala web lebih mudah bagi pengembang. Antarmuka layanan web sederhana Amazon EC2 memungkinkan Anda memperoleh dan mengonfigurasi kapasitas dengan gesekan minimal. Ini memberi Anda kendali penuh atas sumber daya komputasi Anda dan memungkinkan Anda berjalan di lingkungan komputasi Amazon yang telah terbukti.

### NodeJs

Node.JS adalah sistem perangkat lunak yang didesain untuk pengembangan aplikasi web. Node.JS dapat juga disebut sebagai *runtime environment*. Aplikasi ini ditulis dalam campuran Bahasa C++ dan juga JavaScript, mempunyai model *event driven* (basis event) dan *asynchrounous* I/O. Tidak seperti kebanyakan bahasa JavaScript yang dijalankan pada web browser, Node.JS dieksekusi sebagai aplikasi server. Node.JS dapat berjalan di server karena dukungan dari V8 *Engine* buatan Google dan beberapa modul bawaan yang terintegrasi seperti modul http, modul *filesystem*, modul *security* dan beberapa modul penting lainnya.

### React Js

React.js adalah salah satu web *framework* paling popular di dunia javascript, Dokumentasinya yang lengkap dan penggunaannya yang cukup mudah, dapat membuat kita mengembangkan berbagai produk seperti aplikasi web ataupun RESTful API dan dapat digunakan menjadi pijakan untuk membangun web *framework* yang lebih kompleks.

# ANALISIS SISTEM

## Gambaran Organisasi

### Sejarah Singkat

Radar Cirebon adalah surat kabar harian pagi yang terbit di Cirebon, Jawa Barat. Harian ini masih satu grup dengan Jawa Pos. Lahir dan terbentuk 20 Desember 1999. Radar Cirebon memiliki sirkulasi di Cirebon, Indramayu, Kuningan, dan Majalengka. Saat ini memiliki 28 halaman, terbit setiap hari. kecuali hari minggu terbit dengan 20 halaman.

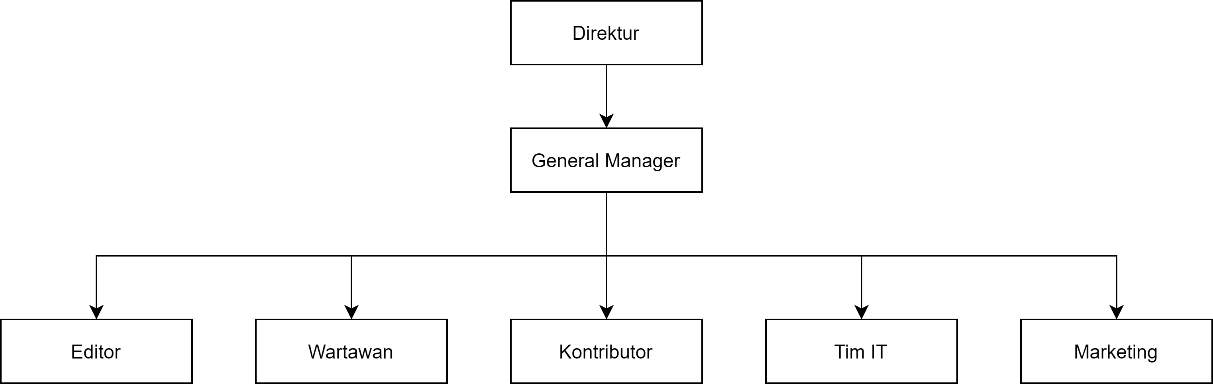
Tahun 2011, Radar Cirebon membangun kantor baru dengan sebutan Grha Pena yang bermarkas di Jalan Perjuangan No 9 Kota Cirebon. Saat ini Radar Cirebon mengembangkan Radar Indramayu, Radar Kuningan dan Radar Majalengka, guna menyajikan berita-berita lokal yang lebih banyak.

Selain koran harian Radar Cirebon juga memiliki stasiun tv lokal dengan sebutan Radar Cirebon TV atau lebih dikenal dengan RCTV, dengan jangkauan wilayah siar Kota/Kabupaten Cirebon dan Indramayu. Untuk masyarakat yang berada di luar wilayah edar Radar Cirebon dapat juga membaca berita-berita wilayah Cirebon dan sekitarnya secara online di www.radarcirebon.com.

### Struktur Organisasi dan Uraian Tugas

* + - 1. Struktur Organisasi

Berikut adalah struktur organisasi dari Radar Cirebon .com yang dapat dilihat pada Gambar 3.1:



Gambar 3. 1 Struktur Organisasi Radar Cirebon .com

* + - 1. Uraian Tugas
         1. Direktur

Menyusun strategi bisnis.

Melakukan evaluasi.

Melakukan rapat rutin.

Implementasi dan melaksanakan Visi dan Misi perusahaan.

* + - * 1. General Manager

Mengatur sekaligus memimpin jalannya operasional kantor.

Mengawasi dan membina manager di tingkat yang lebih rendah.

* + - * 1. Editor

Membuat dan memastikan berita yang akan ditayangkan memiliki struktur yang baik.

* + - * 1. Wartawan

Melaporkan suatu peristiwa kepada tim editor.

Melakukan wawancara dengan narasumber.

* + - * 1. Kontributor

Tugas dari kontributor sama dengan karyawan, yang membedakan hanyalah kontributor berdomisili di daerah-daerah.

* + - * 1. Tim IT

Mengelola dan memastikan *Server* Radar Cirebon berjalan dengan semestinya.

Mengelola dan memastikan *website* radarcirebon.com dan jaringannya berjalan dengan semestinya.

* + - * 1. Marketing

Merencanakan promosi.

Meningkatkan *Search Engine Optimization* (SEO).

## Analisis Sistem Berjalan

### Uraian Prosedur

Analisa prosedur bertujuan untuk mengetahui cara kerja dari sistem yang sedang berjalan maupun sistem yang sedang dalam pengembangan dan perawatan sehingga kelebihan serta kelemahan sistem tersebut dapat diketahui, hal ini tentu sangat penting dalam proses pengembangan atau perawatan sistem tersebut.

Adapun uraian prosedur sistem absensi di Radar Cirebon dalam penelitian ini adalah sebagai berikut:

* + - 1. Prosedur Daftar Karyawan

Tahapan untuk prosedur ini adalah sebagai berikut:

* + - * 1. Admin masuk ke aplikasi khusus admin.
        2. Admin memasukan data karyawan yang terdaftar kedalam *database*. Data yang diinputkan meliputi data personal karyawan dan sidik jari.
        3. Admin mengklik tombol simpan.
      1. Prosedur Absensi

Tahapan untuk prosedur ini adalah sebagai berikut:

* + - * 1. Karyawan menempelkan ibu jari pada mesin deteksi.
        2. Mesin deteksi akan memindai sidik jari.
        3. Mesin deteksi akan mencocokan sidik jari karyawan dengan sidik jari yang ada di *database.*
        4. Jika sidik jari cocok, maka mesin akan berwarna hijau
        5. Jika sidik jari tidak terdaftar maka mesin akan berwarna merah.
      1. Prosedur Membuat Laporan

Tahapan untuk prosedur ini adalah sebagai berikut:

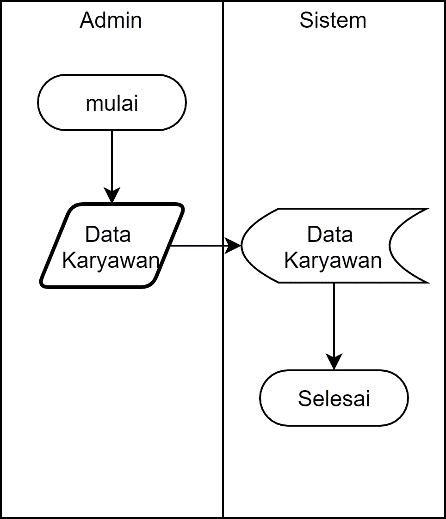
* + - * 1. Admin masuk ke aplikasi khusus admin.
        2. Admin mencetak laporan absensi tiap karyawan.
        3. Laporan diberikan ke bagian lain untuk pertimbangan pemberian gaji.

### Diagram Sistem Prosedur

Diagram sistem prosedur yang ada pada sistem absensi di Radar Cirebon adalah sebagai berikut:

* + - 1. Flowmap Daftar Karyawan

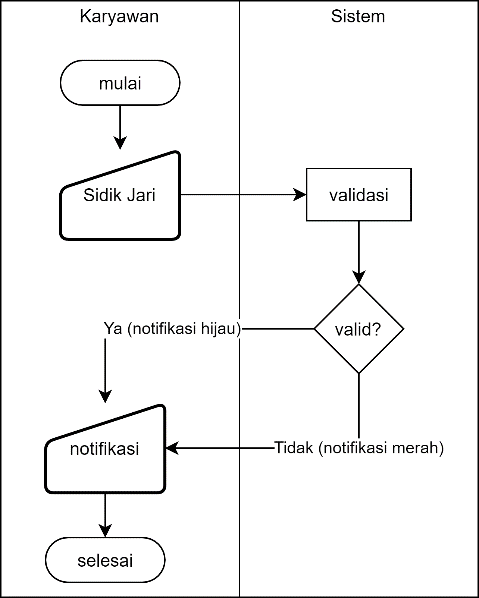
Berikut adalah *Flowmap* Daftar Karyawan yangdapat dilihat pada Gambar 3.2.



Gambar 3. 2 Flowmap Daftar Karyawan

* + - 1. Flowmap Absensi

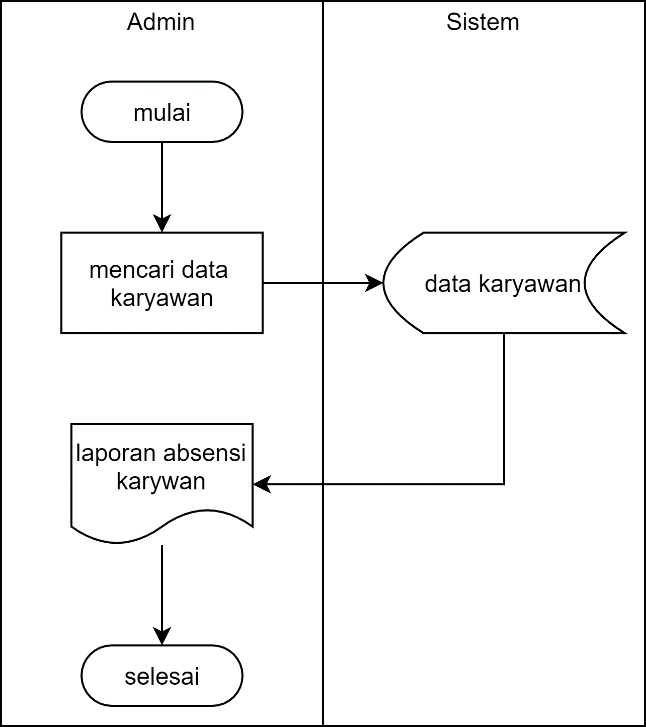
Berikut *Flowmap* Absensi yang dapat dilihat pada Gambar 3.3.



Gambar 3. 3 Flowmap Absensi

* + - 1. Flowmap Membuat Laporan

Berikut adalah *Flowmap* Membuat Laporan yang dapat dilihat pada Gambar 3.4.



Gambar 3. 4 Flowmap Membuat Laporan

### Analisa Dokumen

Analisa dokumen memiliki tujuan yaitu untuk mempelajari dokumen yang terdapat pada sistem, dalam hal ini adalah sistem absensi di Radar Cirebon.

Adapun dokumen yang terdapat pada sistem absensi di Radar Cirebon yang sedang berjalan adalah sebagai berikut:

* + - 1. Dokumen Data Karyawan

|  |  |  |
| --- | --- | --- |
| Nama dokumen | : | Data Karyawan |
| Fungsi | : | Untuk mencatat data karyawan |
| Sumber | : | Admin |
| Bentuk | : | Formulir |
| Atribut | : | nama, *email*, nomor *handphone,* id dan sidik jari |

* + - 1. Dokumen Absensi

|  |  |  |
| --- | --- | --- |
| Nama dokumen | : | Data Absensi |
| Fungsi | : | Untuk mencatat data absensi karyawan |
| Sumber | : | Pengguna |
| Bentuk | : | Formulir |
| Atribut | : | Id, *timestamp,* status, nama |

### Desain Informasi

Hasil Analisa dari dokumen yang ada menghasilkan informasi sebagai berikut:

* + - 1. Informasi Data Karyawan

|  |  |  |
| --- | --- | --- |
| Nama dokumen | : | Data Karyawan |
| Fungsi | : | Untuk mencatat data karyawan |
| Sumber | : | Admin |
| Bentuk | : | Formulir |
| Atribut | : | nama, *email*, nomor *handphone,* id dan sidik jari |

* + - 1. Informasi Absensi

|  |  |  |
| --- | --- | --- |
| Nama dokumen | : | Data Absensi |
| Fungsi | : | Untuk mencatat data absensi karyawan |
| Sumber | : | Pengguna |
| Bentuk | : | Formulir |
| Atribut | : | Id, *timestamp,* status, nama |

# PERANCANGAN SISTEM

## Desain Prosedur

Berdasarkan uraian pada bab sebelumnya, untuk mengatasi permasalahan absensi dimasa pandemi covid-19 di Radar Cirebon, maka pada penelitian ini penulis akan mengimplementasikan pengenalan wajah menggunakan metode Local Binary Pattern Histogram berbasis website untuk mempermudah karyawan Radar Cirebon melakukan absensi sekaligus menerapkan protokol kesehatan.

### Prosedur Pendaftaran Pengguna

* + - 1. Pengguna melakukan pendaftaran dengan menggunakan *email* dan *password*.
      2. Pengguna akan menerima kode *verifikasi* yang akan dikirimkan melalui *email* yang digunakan untuk pendaftaran, masukan kode *verifikasi* ke *form* yang tersedia.
      3. Setelah pendaftaran akun selesai pengguna dapat masuk ke aplikasi.

### Prosedur Login Pengguna

* + - 1. Pengguna melakukan login dengan memasukan alamat *email* yang sudah terdaftar dan *password*.
      2. Jika alamat *email*/*password* salah, silahkan masukan alamat *email/password* yang benar.
      3. Jika pengguna berhasil *login* maka pengguna sudah bisa menggunakan aplikasi

### Prosedur Kelola Akun Pengguna

* + - 1. Pengguna memilih menu untuk melengkapi data diri pengguna.
      2. Pengguna mengisi *form* yang tersedia.
      3. Pengguna mengisi nama, alamat, nomor *handphone*, dan video yang berisi wajah pengguna.
      4. Pengguna menekan tombol *send data* untuk menyimpan perubahan ke *database*.

### Prosedur Melatih Model

* + - 1. Pengguna memilih video yang berisi gambar wajahnya untuk di unggah ke server.
      2. Membuat dataset wajah dari video menggunakan *library OpenCV.*
         1. Langkah pertama yaitu mengunduh file video yang disimpan di S3.
         2. Langkah selanjutnya adalah menanbahkan nama video ke database sesuai dengan data pengguna.
         3. Kemudian muat video menggunakan perintah ‘*cv2.CaptureVideo*’ yang bertujuan untuk memisahkan video *perframe* sehingga dapat menghasilkan foto dari sebuah video.
      3. Membuat model dari *dataset* wajahmenggunakan metode Local Binary Pattern Histogram.
         1. Langkah pertama adalah deteksi wajah.

Pada Langkah ini penulis mendeteksi wajah menggunakan bantuan dari *library OpenCV.* Gambar dari dataset akan dibandingkan dengan *Cascade Classifier* sehingga dapat diketahui bagian wajahnya. Gambar 4.1 merupakan contoh hasil yang diperoleh dari deteksi wajah.



Gambar 4. 1 Hasil Deteksi Wajah

Setelah wajah terdeteksi maka gambar tersebut akan dirubah menjadi gambar abu-abu agar dapat dilatih menjadi sebuah model seperti pada gambar 4.2.



Gambar 4. 2 Merubah Warna Gambar

Langkah selanjutnya yaitu membagi wajah menjadi 64 bagian atau 8x8 seperti terlihat pada Gambar 4.3.



Gambar 4. 3 Pembagian Wilayah Wajah

* + - * 1. Langkah kedua yaitu ekstraksi fitur.

Pada Langkah ini Gambar 4.3 akan dirubah menjadi bentuk array yang terdiri dari kepadatan/kecerahan sebuah piksel atau bila dituliskan dalam angka akan dihasilkan angka antara 0-255. Tabel 4.1 merupakan contoh salah satu bagian dari 64 bagian yang ada.

Tabel 4. 1 Tingkat Kecerahan Gambar

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 85 | 170 | 227 | 28 | 57 | 227 | 28 | 85 | 227 |
| 57 | 0 | 227 | 227 | 170 | 227 | 0 | 57 | 57 |
| 170 | 0 | 28 | 227 | 227 | 0 | 170 | 170 | 113 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 227 | 0 |
| 0 | 0 | 57 | 0 | 0 | 0 | 57 | 0 | 28 |
| 28 | 0 | 0 | 0 | 57 | 227 | 142 | 57 | 57 |

Langkah selanjutnya akan dicari nilai LBP menggunakan Operator LBP yang telah dijelaskan pada BAB II. Tabel 4.1 akan dibagi lagi menjadi tabel kecil yang berukuran 3x3 seperti Tabel 4.2 sampai Tabel 4.7.

Tabel 4. 2 Operator LBP 1

|  |  |  |
| --- | --- | --- |
| 85 | 170 | 227 |
| 57 | 0 | 227 |
| 170 | 0 | 28 |

Tabel 4. 3 Operator LBP 2

|  |  |  |
| --- | --- | --- |
| 28 | 57 | 227 |
| 227 | 170 | 227 |
| 227 | 227 | 0 |

Tabel 4. 4 Operator LBP 3

|  |  |  |
| --- | --- | --- |
| 28 | 85 | 227 |
| 0 | 57 | 57 |
| 170 | 170 | 113 |

Tabel 4. 5 Operator LBP 4

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 57 |
| 28 | 0 | 0 |

Tabel 4. 6 Operator LBP 5

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 57 | 227 |

Tabel 4. 7 Operator LBP 6

|  |  |  |
| --- | --- | --- |
| 0 | 227 | 0 |
| 57 | 0 | 28 |
| 142 | 57 | 57 |

Menghitung nilai LBP pada Tabel 4.2

Nilai 85 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.3 dan persamaan 1.4.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 170 akan dihitung menggunakan persamaan 1.1 kemudian persamaan 1.2 sehingga menghasilkan persamaan 1.5 dan persamaan 1.6.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.7 dan persamaan 1.8.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.9 dan persamaan 1.10.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 28 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.11 dan persamaan 1.12.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.13 dan persamaan 1.14.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 170 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.15 dan persamaan 1.16.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 57 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.17 dan persamaan 1.18.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Dari persamaan 1.3 sampai persamaan 1.18 dihasilkan nilai LBP berturut-turut yaitu 1,1,1,1,1,1,1,1 yang bila diubah ke bentuk biner maka senilai dengan 255 yang merupakan nilai asli atau nilai tengah. Hasil perhitungan LBP yang diperoleh akan dikalikan dengan bilangan biner antara 1 hingga 128 yang hasilnya dapat dilihat pada Tabel 4.8.

Tabel 4. 8 Hasil Operator LBP 1

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 4 |
| 8 | 255 | 16 |
| 32 | 64 | 128 |

Menghitung nilai LBP pada Tabel 4.3

Nilai 28 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.19 dan persamaan 1.20.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 57 akan dihitung menggunakan persamaan 1.1 kemudian persamaan 1.2 sehingga menghasilkan persamaan 1.21 dan persamaan 1.22.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.23 dan persamaan 1.24.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.25 dan persamaan 1.26.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.27 dan persamaan 1.28.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.29 dan persamaan 1.30.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.31 dan persamaan 1.32.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.33 dan persamaan 1.34.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Dari persamaan 1.19 sampai persamaan 1.34 dihasilkan nilai LBP berturut-turut yaitu 0,0,1,1,0,1,1,1 yang bila diubah ke bentuk biner maka senilai dengan 55 yang merupakan nilai asli atau nilai tengah. Hasil perhitungan LBP yang diperoleh akan dikalikan dengan bilangan biner antara 1 hingga 128 yang hasilnya dapat dilihat pada Tabel 4.9.

Tabel 4. 9 Hasil Operator LBP 2

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 4 |
| 8 | 55 | 16 |
| 32 | 64 | 0 |

Menghitung nilai LBP pada Tabel 4.4

Nilai 28 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.35 dan persamaan 1.36.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 85 akan dihitung menggunakan persamaan 1.1 kemudian persamaan 1.2 sehingga menghasilkan persamaan 1.37 dan persamaan 1.38.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.39 dan persamaan 1.40.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.41 dan persamaan 1.42.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 113 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.43 dan persamaan 1.44.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 170 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.45 dan persamaan 1.46.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 170 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.47 dan persamaan 1.48.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.49 dan persamaan 1.50.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Dari persamaan 1.35 sampai persamaan 1.50 dihasilkan nilai LBP berturut-turut yaitu 0,1,1,1,1,1,1,0 yang bila diubah ke bentuk biner maka senilai dengan 126 yang merupakan nilai asli atau nilai tengah. Hasil perhitungan LBP yang diperoleh akan dikalikan dengan bilangan biner antara 1 hingga 128 yang hasilnya dapat dilihat pada Tabel 4.10.

Tabel 4. 10 Hasil Operator LBP 3

|  |  |  |
| --- | --- | --- |
| 0 | 2 | 4 |
| 0 | 126 | 0 |
| 32 | 64 | 128 |

Menghitung nilai LBP pada Tabel 4.5

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.51 dan persamaan 1.52.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 kemudian persamaan 1.2 sehingga menghasilkan persamaan 1.53 dan persamaan 1.54.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.55 dan persamaan 1.56.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 57 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.57 dan persamaan 1.58.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.59 dan persamaan 1.60.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.61 dan persamaan 1.62.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 28 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.63 dan persamaan 1.64.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.65 dan persamaan 1.66.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Dari persamaan 1.51 sampai persamaan 1.66 dihasilkan nilai LBP berturut-turut yaitu 0,0,0,1,0,0,1,0 yang bila diubah ke bentuk biner maka senilai dengan 18 yang merupakan nilai asli atau nilai tengah. Hasil perhitungan LBP yang diperoleh akan dikalikan dengan bilangan biner antara 1 hingga 128 yang hasilnya dapat dilihat pada Tabel 4.11.

Tabel 4. 11 Hasil Operator LBP 4

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 18 | 16 |
| 32 | 0 | 0 |

Menghitung nilai LBP pada Tabel 4.6

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.67 dan persamaan 1.68.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 kemudian persamaan 1.2 sehingga menghasilkan persamaan 1.69 dan persamaan 1.70.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.71 dan persamaan 1.72.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.73 dan persamaan 1.74.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.75 dan persamaan 1.76.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 57 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.77 dan persamaan 1.78.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.79 dan persamaan 1.80.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.81 dan persamaan 1.82.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Dari persamaan 1.67 sampai persamaan 1.82 dihasilkan nilai LBP berturut-turut yaitu 0,0,0,0,1,1,0,0 yang bila diubah ke bentuk biner maka senilai dengan 12. Hasil perhitungan LBP yang diperoleh akan dikalikan dengan bilangan biner antara 1 hingga 128 yang hasilnya dapat dilihat pada Tabel 4.12.

Tabel 4. 12 Hasil Operator LBP 5

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 12 | 0 |
| 0 | 64 | 128 |

Menghitung nilai LBP pada Tabel 4.7

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.83 dan persamaan 1.84.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 227 akan dihitung menggunakan persamaan 1.1 kemudian persamaan 1.2 sehingga menghasilkan persamaan 1.85 dan persamaan 1.86.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 0 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.87 dan persamaan 1.88.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 28 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.89 dan persamaan 1.90.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 57 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.91 dan persamaan 1.92.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 57 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.93 dan persamaan 1.94.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 142 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.95 dan persamaan 1.96.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Nilai 57 akan dihitung menggunakan persamaan 1.1 dan 1.2 sehingga menghasilkan persamaan 1.97 dan persamaan 1.98.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |
|  |  | (1.) | |

Dari persamaan 1.83 sampai persamaan 1.98 dihasilkan nilai LBP berturut-turut yaitu 0,1,0,1,1,1,1,1 yang bila diubah ke bentuk biner maka senilai dengan 95. Hasil perhitungan LBP yang diperoleh akan dikalikan dengan bilangan biner antara 1 hingga 128 yang hasilnya dapat dilihat pada Tabel 4.13.

Tabel 4. 13 Hasil Operator LBP 6

|  |  |  |
| --- | --- | --- |
| 0 | 2 | 0 |
| 8 | 95 | 16 |
| 32 | 64 | 128 |

Tabel 4.8 hingga Tabel 4.13 dapat digabungkan kembali menjadi satu tabel namun hanya titik pusatnya saja yang digunakan dan dapat dilihat pada Tabel 4.14.

Tabel 4. 14 Nilai LBP Dari Satu Bagian Gambar Wajah

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 4 | 0 | 0 | 4 | 0 | 2 | 4 |
| 8 | 255 | 16 | 8 | 55 | 16 | 0 | 126 | 0 |
| 32 | 64 | 128 | 32 | 64 | 0 | 32 | 64 | 128 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 |
| 0 | 18 | 16 | 0 | 12 | 0 | 8 | 95 | 16 |
| 32 | 0 | 0 | 0 | 64 | 128 | 32 | 64 | 128 |

Langkah diatas diulang hingga telah diketahui nilai LBP dari setiap bagiannya. Selanjutnya Nilai LBP dari setiap bagian disatukan hingga membentuk histogram seperti pada Gambar 2.4.

* + - * 1. Membuat *dataset* yang berisi nilai LBP dari setiap gambar wajah.

Dalam tahap ini penulis menggunakan *library OpenCV* untuk membuat dataset yang berisi nilai LBP dari setiap gambar wajah yang ada di *dataset* wajah.

### Prosedur Memprediksi Wajah

* + - 1. Pengguna mengambil gambar wajah.
      2. Pengguna mengunggah foto wajah ke s3.
      3. Sistem akan mengunduh gambar yang dikirimkan pengguna.
      4. Pengenalan wajah menggunakan metode Local Binary Pattern Histogram.
         1. Mendeteksi wajah.

Pada Langkah ini penulis mendeteksi wajah menggunakan bantuan dari *library OpenCV.* Gambar yang diunduh akan dibandingkan dengan *Cascade Classifier* sehingga dapat diketahui bagian wajahnya. Gambar 4.1 merupakan contoh hasil yang diperoleh dari deteksi wajah.

Setelah wajah terdeteksi maka gambar tersebut akan dirubah menjadi gambar abu-abu agar dapat dilatih menjadi sebuah model seperti pada gambar 4.2. Langkah selanjutnya yaitu membagi wajah menjadi 64 bagian atau 8x8 seperti terlihat pada Gambar 4.3. Langkah selanjutnya sama seperti Langkah-langkah ekstraksi fitur pada Prosedur Memprediksi Wajah.

Selanjutnya setelah melakukan ekstraksi fitur dan mendapatkan nilai Histogram LBP maka dilakukan pengukuran jarak antara histogram LBP dengan histogram dari dataset yang sudah dilatih sebelumnya. Mengukur jarak dari dua histogram dapat menggunakan persamaan *euclidean distance* seperti pada persamaan 1.99.

|  |  |  |
| --- | --- | --- |
|  |  | (1.) |

Hasil dari persamaan 1.99 merupakan jarak antara kedua histogram, nilai ini tidak memiliki batas namun semangin kecil (mendekati 0) nilainya maka tingkat kemiripan semakin tinggi maupun sebaliknya jika nilainya semakin besar (menjauhi 0) maka tingkat kemiripannya semakin kecil.

## Desain Dokumen dan informasi

### Desain Dokumen

Desain dokumen baru pada *website* website Absensi Online Radar Cirebon setelah diimplementasikan dengan teknik pengenalan wajah adalah sebagai berikut:

* + - 1. Dokumen pengguna

|  |  |  |
| --- | --- | --- |
| Nama dokumen | : | Data Pengguna |
| Fungsi | : | Untuk melengkapi data pengguna |
| Sumber | : | Pengguna |
| Bentuk | : | Formulir |
| Atribut | : | nama, *email*, nomor *handphone,* id wajah dan alamat |

* + - 1. Dokumen absensi

|  |  |  |
| --- | --- | --- |
| Nama dokumen | : | Data Absensi |
| Fungsi | : | Untuk bukti absensi |
| Sumber | : | Pengguna/ Sistem |
| Bentuk | : | Formulir |
| Atribut | : | *\_typename, confidence, face\_id, time, user\_id, year, dan location.* |

### Desain Informasi

* + - 1. Informasi Data Pengguna

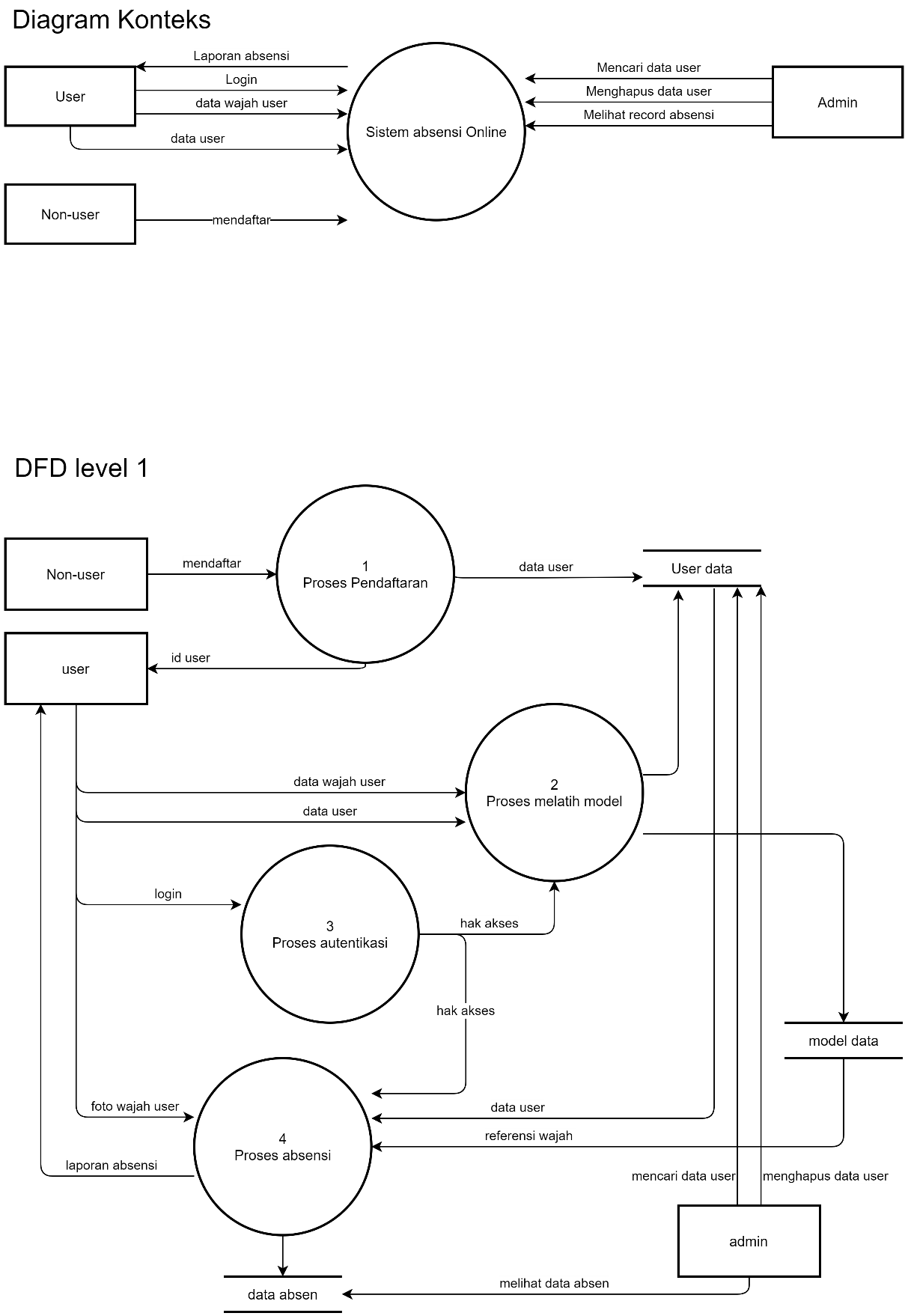
|  |  |  |
| --- | --- | --- |
| Nama dokumen | : | Data Pengguna |
| Fungsi | : | Untuk melengkapi data pengguna |
| Sumber | : | Pengguna |
| Bentuk | : | Formulir |
| Atribut | : | nama, *email*, nomor *handphone,* id wajah dan alamat |

* + - 1. Informasi Absensi

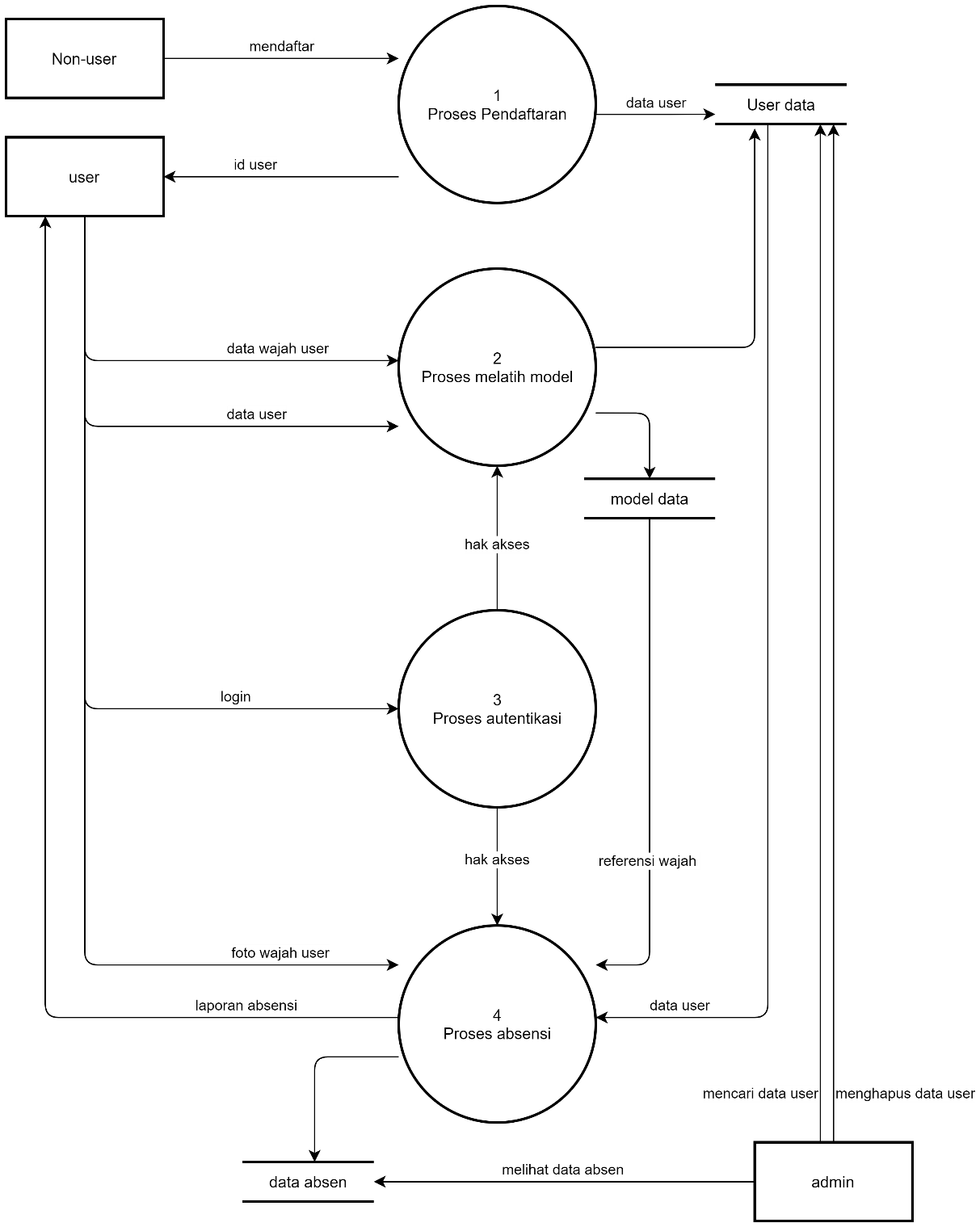
|  |  |  |
| --- | --- | --- |
| Nama dokumen | : | Data Absensi |
| Fungsi | : | Untuk bukti absensi |
| Sumber | : | Pengguna/ Sistem |
| Bentuk | : | Formulir |
| Atribut | : | *\_typename, confidence, face\_id, time, user\_id, year, dan location.* |

## Desain Aliran Data

### Data Flow Diagram

Berikut adalah Diagram Konteksyang penulis buat pada penelitian ini yang dapat dilihat pada Gambar 4.4.

Gambar 4. 4 Diagram Konteks

Dari Gambar 4.4 dapat diturunkan menjadi *Data Flow Diagram* level 1 yang dapat dilihat pada Gambar 4.5.

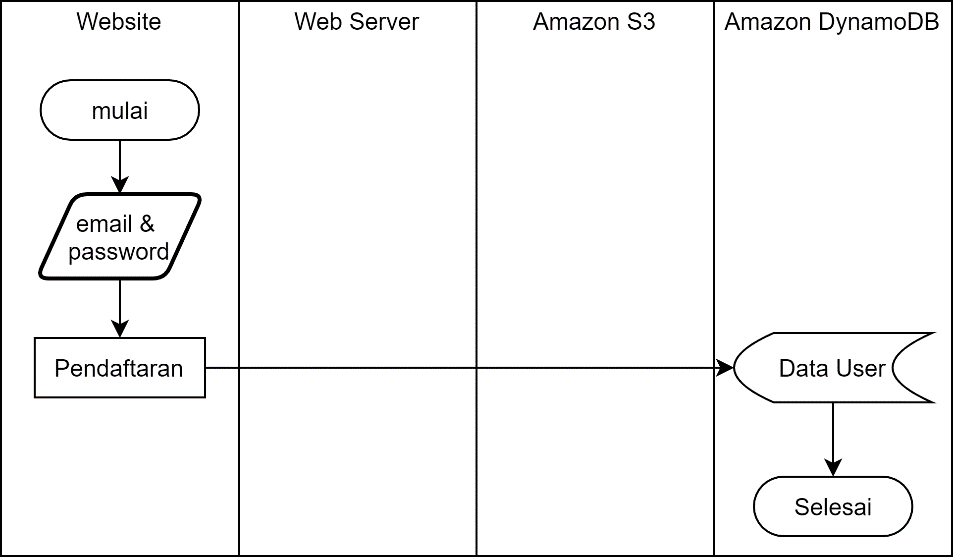
Gambar 4. 5 DFD Level 1

### Flowmap

Untuk mempermudah dalam pembuatan program penulis menggunakan *flowmap* sebagai acuan aliran data dalam sebuah program. *Flowmap* aliran data program dibagi berdasarkan prosedur sebagai berikut:

* + - 1. Flowmap Pendaftaran

Berikut adalah *Flowmap* Pendaftaran yangdapat dilihat pada Gambar 4.6.



Gambar 4. 6 Flowmap Pendaftaran

* + - 1. Flowmap Login

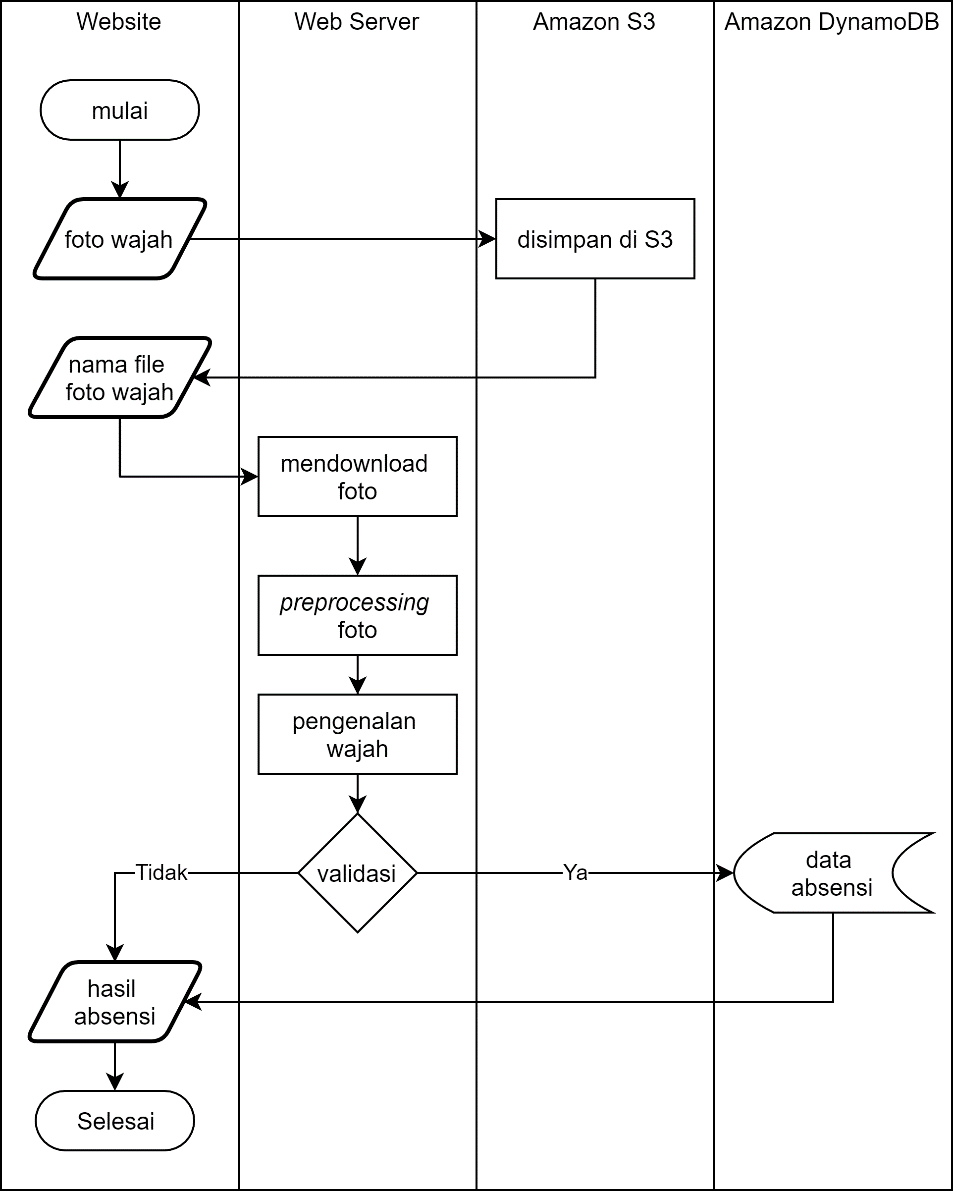
Berikut adalah *flowmap login* yang dapat dilihat pada Gambar 4.7.



Gambar 4. 7 Flowmap Login

* + - 1. Flowmap Memprediksi Wajah (Absensi)

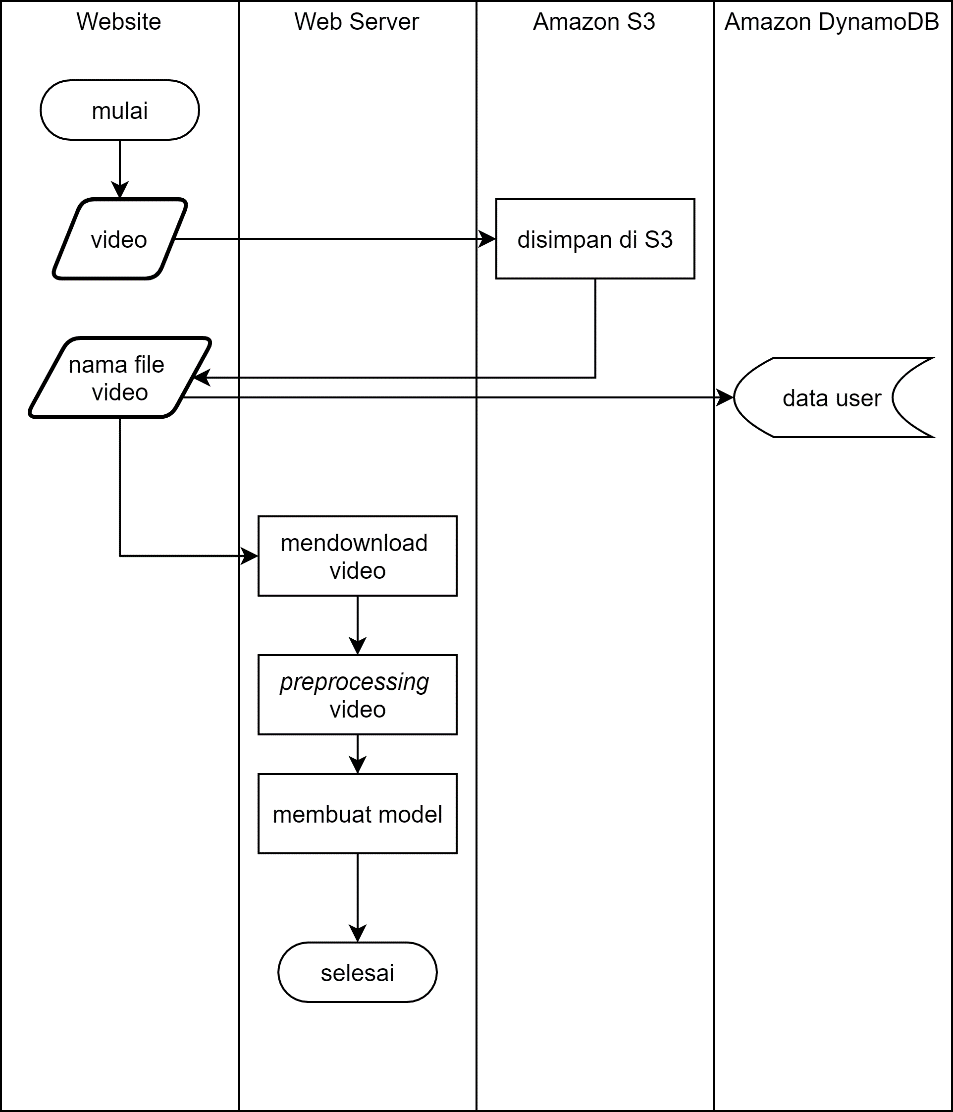
Berikut adalah *flowmap* memprediksi wajah yang dapat dilihat pada Gambar 4.8.



Gambar 4. 8 Flowmap Memprediksi Wajah

* + - 1. Flowmap Melatih Model

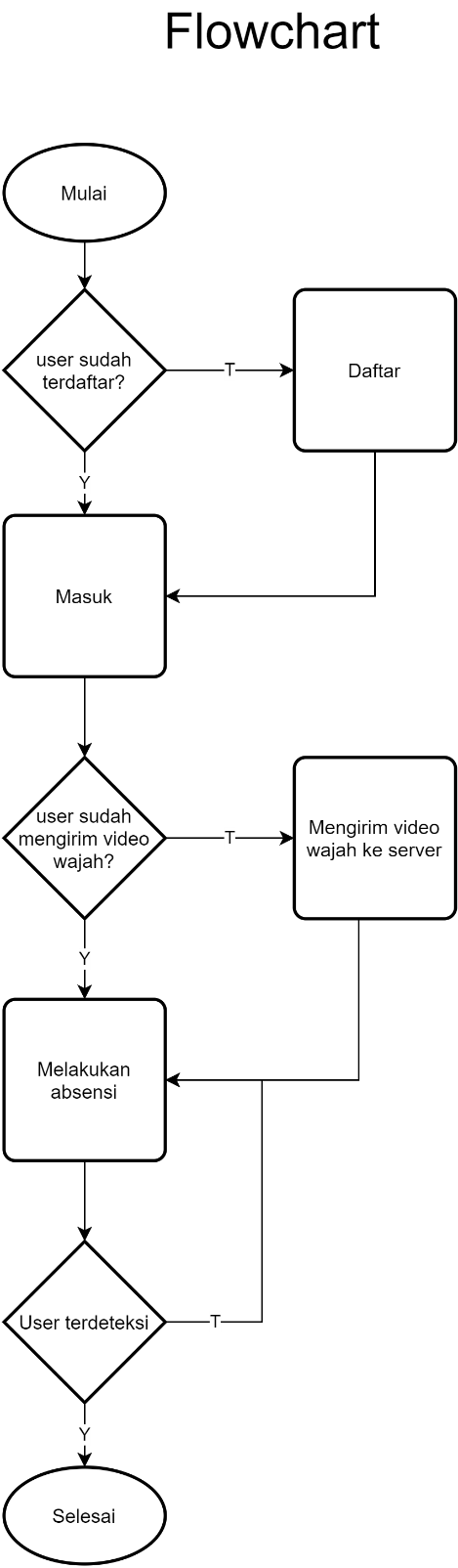
Berikut adalah *flowmap* melatih model yang dapat dilihat pada Gambar 4.9.



Gambar 4. 9 Flowmap Melatih Model

### Flowchart

Untuk mempermudah dalam pembuatan program penulis menggunakan *flowchart* sebagai acuan alur program. *Flowchart* alur program ditunjukan oleh Gambar 4.10 berikut:



Gambar 4. 10 Flowchart Program

### Skema Database

*Database nosql* adalah *database* yang tidak memiliki perintah SQL dan konsep penyimpanannya semistuktural atau tidak struktural dan tidak harus memiliki relasi layaknya tabel-tabel MySQL. Dalam penelitian ini penulis memilih DynamoDB sebagai *database nosql* karena DynamoDB memiliki kecepatan baca yang konstan sehingga cocok untuk aplikasi *realtime*.

* + - 1. Skema Tabel Pengguna

Berikut adalah skema yang penulis buat dalam bentuk json untuk tabel pengguna yang dapat dilihat pada Tabel 4.15.

Tabel 4. 15 Skema Tabel Pengguna

|  |
| --- |
|  |

* + - 1. Skema Tabel Data Absensi

Berikut adalah skema yang penulis buat dalam bentuk json untuk tabel data absensi yang dapat dilihat pada Tabel 4.16.

Tabel 4. 16 Skema Tabel Data Absensi

|  |
| --- |
|  |

* + - 1. Skema Model

Berikut adalah skema yang dihasilkan oleh OpenCV dalam bentuk yaml yang dapat dilihat pada Tabel 4.17.

Tabel 4. 17 Skema Model

|  |
| --- |
|  |

## Desain Interface dan Struktur Menu

### Desain Interface

Desain adalah sesuatu yang sangat dibutuhkan dalam membuat sebuah sistem, tujuannya agar sistem tersebut mudah untuk digunakan oleh pengguna. Adapun desain *interface* untuk *website* Absensi Online Radar Cirebon setelah diimplementasikan dengan teknik pengenalan wajah adalah sebagai berikut:

* + - 1. Desain Halaman Login

Berikut adalah Desain Halaman *Login* yang dapat dilihat pada Gambar 4.11:



Gambar 4. 11 Desain Halaman Login

* + - 1. Desain Halaman Dashboard

Berikut adalah Desain Halaman Setting yang dapat dilihat pada Gambar 4.12:



Gambar 4. 12 Desain Halaman Dashboard

* + - 1. Desain Halaman Setting

Berikut adalah Desain Halaman *Setting* yang dapat dilihat pada Gambar 4.13:



Gambar 4. 13 Desain Halaman Setting

* + - 1. Desain Halaman Absensi

Berikut adalah Desain Halaman Absensi yang dapat dilihat pada Gambar 4.14:



Gambar 4. 14 Desain Halaman Absensi

* + - 1. Desain Halaman Laporan

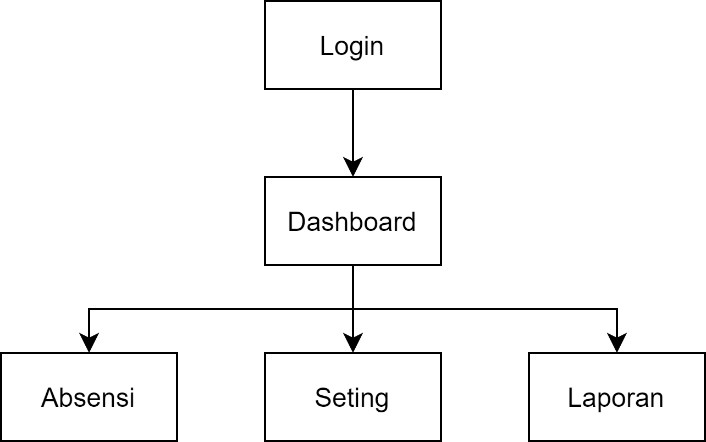
Berikut adalah Desain Halaman Laporan yang dapat dilihat pada Gambar 4.15:



Gambar 4. 15 Desain Halaman Laporan

### Struktur Menu

Berikut adalah struktur dari website Absensi Online Radar Cirebon setelah diimplementasikan dengan teknik pengenalan wajah dapat dilihat pada Gambar 4.16:



Gambar 4. 16 Struktur Menu Sistem

## Implementasi Sistem

### Konfigurasi Perangkat Lunak

Kebutuhan perangkat lunak untuk mengelola server pengenalan wajah guna mendukung *website* Unit Robotika STIKOM Poltek Cirebon yang dijalankan dalam Amazon EC2 adalah sebagai berikut:

* + - 1. Linux Ubuntu 18.04.
      2. Nginx*.*
      3. Flask.
      4. Gunicorn.

Sedangkan perangkat lunak yang digunakan untuk pengembangan website adalah Visual Studio Code dan Postman.

### Konfigurasi Perangkat Keras

Dalam pembuatan *website* Absensi Online Radar Cirebon penulis hanya menggunakan perangkat keras pada saat pengembangan *website*, karena penulis menggunakan Amazon Web Service sebagai sarana *hosting* sehingga tidak membutuhkan perangkat keras secara langsung. Berikut perangkat keras yang digunakan pada saat pengembangan *website*.

* + - 1. Processor Intel Core i3 7100.
      2. Ram 4 GB.
      3. Penyimpanan 250 GB.
      4. Mouse, keyboard, monitor, dan webcam.

### Implementasi Program

* + - 1. Sintaks Pembuatan Dataset

Berikut adalah Sintaks Pembuatan Dataset yang dapat dilihat pada Tabel 4.18:

| Tabel 4. 18 Sintaks Membuat Dataset |
| --- |
| import cv2  import os  import numpy as np  import time  import matplotlib.pyplot as plt  import sys  import argparse as arg  class Train():    def \_\_init\_\_(self, face\_cascade,config,username):  self.username = username  # self.current\_dir = current\_dir  self.\_Face\_Cascade = cv2.CascadeClassifier(face\_cascade)  self.dataset\_path("dataset/")  self.recognizer = cv2.face.LBPHFaceRecognizer\_create(config[0], config[1], config[2], config[3],config[4])  def dataset\_path(self, path):  dir = os.path.dirname(path)  if not os.path.exists(dir):  os.makedirs(dir)  def ReadName(self):  NAME = []  with open ("users.txt", "r") as f:  for line in f:  NAME.append(line.split(",")[1].rstrip())  return NAME  def AddUser(self):  # Name = input('\n[INFO] Masukan nama user : ')  Name = self.username  info = open('users.txt', "a+")  ID = len(open("users.txt").readlines( ))  info.write(str(ID) + "," + Name + "\n")  print("\n[INFO] Tambah user berhasil, ID:" + str(ID))  info.close  return ID  def getImageWithLabels(self,path):  imagePaths = [os.path.join(path, f) for f in os.listdir(path)]  faceSamples = []  ids = []  for imagePath in imagePaths:  img = cv2.imread(imagePath, 0)  img\_numpy = np.array(img, 'uint8')  id = int(os.path.split(imagePath)[- 1].split('.')[1])  faceSamples.append(img\_numpy)  ids.append(id)  return faceSamples, ids  def train(self, path, file\_name):  # os.chdir('tmp')  # real\_path = '{0}/{1}'.format(self.current\_dir,path)  print("\n[INFO] Training wajah sedang dimulai...")  time.sleep(1)  faces, ids = self.getImageWithLabels(path)  self.recognizer.update(faces,np.array(ids))  self.recognizer.write(file\_name)  print("\n[INFO] Model sukses melatih user ID: {0}".format(len (np.unique (ids))))  print("\n[INFO] Menutup program")  return len(np.unique (ids))  def create\_Rect(self, Image, face, color):  x,y,w,h = face  cv2.line(Image, (x, y), (int(x + (w/5)),y), color, 2)  cv2.line(Image, (int(x+((w/5)\*4)), y), (x+w, y), color, 2)  cv2.line(Image, (x, y), (x,int(y+(h/5))), color, 2)  cv2.line(Image, (x+w, y), (x+w, int(y+(h/5))), color, 2)  cv2.line(Image, (x, int(y+(h/5\*4))), (x, y+h), color, 2)  cv2.line(Image, (x, int(y+h)), (x + int(w/5) ,y+h), color, 2)  cv2.line(Image, (x+int((w/5)\*4), y+h), (x + w, y + h), color, 2)  cv2.line(Image, (x+w, int(y+(h/5\*4))), (x+w, y+h), color, 2)    def createDataset(self,samples,cam,dataset\_name):  fig, axs = plt.subplots(10,5,figsize=(20,20), facecolor='w', edgecolor='k')  fig.subplots\_adjust(hspace=.5, wspace=.001)  self.dataset\_path(dataset\_name)  count = 0  face\_id = self.AddUser()  print('\n[INFO] Membuat dataset')  while(True):  success, image = cam.read()  # convert image to grayscale  gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)  faces = self.\_Face\_Cascade.detectMultiScale(gray, scaleFactor = 1.098, minNeighbors = 6, minSize = (50, 50))  if(len(faces)> 1):  print('\n[WARNING] Terdeteksi lebih dari 1 wajah')  continue  try:  for \_,face in enumerate(faces):  x, y, w, h = face  gray\_chunk = gray[y-30: y + h + 30, x-30: x + w + 30]  image\_chunk = image[y: y + h, x: x + w]  self.create\_Rect(image, face, [0,255,0])  # cv2.imshow("Video", image)  #get center image  # image\_center = tuple(np.array(gray\_chunk.shape) / 2)  # rot\_mat = cv2.getRotationMatrix2D(image\_center, angle\_degree, 1.0)  # rotated\_image = cv2.warpAffine(gray\_chunk, rot\_mat, gray\_chunk.shape, flags=cv2.INTER\_LINEAR)  print("\n[INFO] Adding image number {} to the dataset".format(count))  # Save image  cv2.imwrite("dataset/User." + str(face\_id) + '.' + str(count) + ".jpg " ,  image)  axs[int(count/5)][count%5].imshow(image,cmap='gray', vmin=0, vmax=255)  axs[int(count/5)][count%5].set\_title("Person." + str(face\_id) + '.' + str(count) + ".jpg ",  fontdict={'fontsize': 15,'fontweight': 'medium'})  axs[int(count/5)][count%5].axis('off')  count += 1  except Exception as e:  print(e)  print('[WARNING] Ada error')  continue  if cv2.waitKey(1) & 0xff == 27:  break  elif count >= samples:  break  print('\n[INFO] Dataset berhasil dibuat')  # cam.release()  # cv2.destroyAllWindows()  # plt.show()  def Arg\_Parse():  Arg\_Par = arg.ArgumentParser()  Arg\_Par.add\_argument("-v", "--video",  help = "path of the video or if not then webcam")  Arg\_Par.add\_argument("-c", "--camera",  help = "Id of the camera")  arg\_list = vars(Arg\_Par.parse\_args())  return arg\_list  if \_\_name\_\_ == "\_\_main\_\_":  if len(sys.argv) == 1:  print("Masukan argumen yang valid!")  sys.exit()  Arg\_list = Arg\_Parse()  face\_cascade = 'lib/haarcascade\_frontalface\_default.xml'  if not (os.path.isfile(face\_cascade)):  raise RuntimeError("%s: not found" % face\_cascade)  samples = 2  dataset\_name = 'dataset/'  file\_name = 'train.yml'  radius = 1  neighbour = 8  grid\_x = 8  grid\_y = 8  treshold = 140  var = list([radius,neighbour,grid\_x,grid\_y,treshold])  model = Train(face\_cascade,var)  if Arg\_list["video"] != None :  video = cv2.VideoCapture(Arg\_list["video"])  #create a dataset for further model training  print('{0} {1} {2}'.format(samples,video,dataset\_name))  model.createDataset(samples,video,dataset\_name)  #Training the model  model.train(dataset\_name,file\_name)  if Arg\_list["camera"] != None :  camera = cv2.VideoCapture(eval(Arg\_list["camera"]))  camera.set(3, 640)  camera.set(4, 480)  model.createDataset(samples,camera,dataset\_name)  #Training the model  model.train(dataset\_name,file\_name) |

* + - 1. Sintaks Membuat Model

Berikut adalah Sintaks Membuat Model yang dapat dilihat pada Tabel 4.19:

| Tabel 4. 19 Sintaks Membuat Model |
| --- |
| import numpy as np  import cv2  import os  #Face detection is done  def faceDetection(test\_img):  gray\_img=cv2.cvtColor(test\_img,cv2.COLOR\_BGR2GRAY)  face\_haar=cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')  faces=face\_haar.detectMultiScale(gray\_img, scaleFactor=1.2,  minNeighbors=4,  minSize=(30, 30))  return faces,gray\_img  # labeling dataset  def labels\_for\_training\_data(directory):  faces=[]  faceID=[]  file\_count = 0  for path,subdirnames,filenames in os.walk(directory):  for filename in filenames:  if filename.startswith("."):  print("skipping system file")  continue  file\_count += 1  print(f'{filename} with id: {filename.split(".")[1]}')  id = filename.split(".")[1]  image = cv2.imread(f'dataset/{filename}')  if image is None:  print(f'{filename} not exist!')  continue  faces\_rect,gray\_img=faceDetection(image)  if len(faces\_rect)!=1:  print(f'{filename} -> no/multiple face detected')    continue  (x,y,w,h)=faces\_rect[0]  tiny\_face=gray\_img[y:y+w,x:x+h]  faces.append(tiny\_face)    faceID.append(int(id))  print(f'{file\_count=}')  return faces,faceID  #Here training Classifier is called  def train\_classifier(faces,faceID):  face\_recognizer=cv2.face.LBPHFaceRecognizer\_create()  face\_recognizer.train(faces,np.array(faceID))  face\_recognizer.write(f'{os.path.dirname(os.path.realpath(\_\_file\_\_))}/model.yml')  return face\_recognizer  #Drawing a Rectangle on the Face Function  def draw\_rect(test\_img,face):  (x,y,w,h)=face  cv2.rectangle(test\_img,(x,y),(x+w,y+h),(0,255,0),thickness=3)  #Putting text on images  def put\_text(test\_img,text,x,y):  cv2.putText(test\_img,text,(x,y),cv2.FONT\_HERSHEY\_DUPLEX,3,(255,0,0),6) |

* + - 1. Sintaks Pengenalan Wajah

Berikut adalah Sintaks Pengenalan Wajah yang dapat dilihat pada Tabel 4.20:

| Tabel 4. 20 Sintaks Pengenalan Wajah |
| --- |
| import cv2  import numpy as np  import csv  import os  import logging  logging.basicConfig(filename='predict.log', format='%(asctime)s %(levelname)-8s %(message)s',level=logging.DEBUG, datefmt='%Y-%m-%d %H:%M:%S')  # global recognizer = cv2.face.LBPHFaceRecognizer\_create()  # recognizer.read('train.yml')  # global faceCascade = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')  # global font = cv2.FONT\_HERSHEY\_SIMPLEX  class Predict():  def \_\_init\_\_(self, image):  self.image = image  # self.recognizer = cv2.face.LBPHFaceRecognizer\_create()  # self.recognizer.read('train.yml')  # self.faceCascade = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')  # self.font = cv2.FONT\_HERSHEY\_SIMPLEX  def print(self):  return os.path.join('dataset', self.image)  def predict(self):  logging.info('-----------------------------')  logging.info('Predict Start')  # print(self.image)  # print(os.path.join('dataset', self.image))  # print('{} '.format(self.image))  # logging.info('{}'.format(os.path.join('dataset', self.image)))  # logging.info('{}'.format('dataset/{} '.format(self.image)))    # img = cv2.imread('image/59b53e2d-0c03-44ec-a4a9-23d496caf6e0.jpg')  img = cv2.imread(str(self.image))  # img = cv2.imread(os.path.join('dataset', self.image)+ " ")  logging.info('Recognizer start')  recognizer = cv2.face.LBPHFaceRecognizer\_create()  recognizer.read('model.yml')  faceCascade = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')  font = cv2.FONT\_HERSHEY\_SIMPLEX  ID = None  CONF = None  # recognizer = cv2.face.LBPHFaceRecognizer\_create()  # recognizer.read('train.yml')  # faceCascade = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')  # font = cv2.FONT\_HERSHEY\_SIMPLEX  # img = cv2.imread('/home/ubuntu/skripsi/dataset/User.3.0.jpg ')  # img = cv2.imread(self.image)  ### resize image  imgs = cv2.resize(img,(0,0),None,0.25,0.25)  # convert to gray scale  gray = cv2.cvtColor(imgs, cv2.COLOR\_BGR2GRAY)  ### detect the face  faces = faceCascade.detectMultiScale(gray,scaleFactor=1.05,minNeighbors=4,minSize=(30, 30))  treshold = cv2.face\_LBPHFaceRecognizer.getThreshold  print(treshold)  for (x,y,w,h) in faces:  Id,conf = recognizer.predict(gray[y:y+h,x:x+w])  # cv2.rectangle(imgS, (x,y), (x+w, y+h), (255,0,0), 2)  # cv2.rectangle(img, (x, y), (x + w, y + h), (0, 260, 0), 2)  # cv2.putText(imgS, str(Id), (x,y-40),font, 2, (255,255,255), 3)  print('ID {0}, confidence {1}'.format(Id, conf))  ID = Id  CONF = conf  logging.info('Predict result: {} with {} distance'.format(ID, CONF))  print('ID {0}, confidence {1}'.format(ID, CONF))  return ID,CONF |

* + - 1. Sintaks Server Pengenalan Wajah

Berikut adalah Sintaks Server Pengenalan Wajah yang dapat dilihat pada Tabel 4.21:

| Tabel 4. 21 Sintaks Server Pengenalan Wajah |
| --- |
| from flask import Flask, request, make\_response  from flask\_cors import CORS  from file\_downloader import downloader, getData, updateData, updateDataNew, putData  import cv2  from train import Train  from predict import Predict  import uuid  import os  import datetime  import json  import logging  import math  import recognize as re  logging.basicConfig(filename='skripsi.log', format='%(asctime)s %(levelname)-8s %(message)s', level=logging.INFO, datefmt='%Y-%m-%d %H:%M:%S')  app = Flask(\_\_name\_\_)  cors = CORS(app, resources={r"/\*": {"origins": "\*"}})  @app.route('/')  def test():  current\_time = datetime.datetime.now()  response = make\_response('<h1>Success at {} (server time)</h1>'.format(current\_time))  response.headers['Content-Security-Policy'] = 'upgrade-insecure-requests'  return response  @app.route('/recognize/predict', methods=['POST'])  def predict():  logging.info('halo')  request\_data = request.get\_json()      if request\_data:  image = request\_data['image']  expect\_username = request\_data['username']  location = request\_data['location']  username = None  conf = None    debug\_id = None  debug\_sim = None  debug\_dis = None  debug\_username = None  result\_id = None  result\_sim = None  result\_dis = None  result\_username = None  debug\_user\_data = None  result\_user\_data = None  minSim = 40 # 75%    image\_path = 'image/{}.jpg'.format(uuid.uuid4())  # download video from s3  downloader\_client = downloader(key=image,bucket='dannynurdin', destination=image\_path)  downloader\_client.download()    model = Predict(image\_path)    res = model.print()  id,c = model.predict()  logging.info('DATA => id from model == ', id)  print('DATA => id from model == ', id)  logging.info('success {} - {}'.format(id, c))  if c:  disMax = 140.0  simMax = 100  similarity = simMax - simMax/disMax\*c  conf = similarity      with open('users.txt') as f:  datafile = f.readlines()  for line in datafile:  Id = line.split(',')[0]  print('DATA => Id from model == ', Id)  logging.info('DATA => Id from model == ', Id)  if Id == id:  dump= line.split(',')[1]  debug\_username = dump.split('\n')[0]  debug\_id = id  debug\_sim = conf  debug\_dis = c    dynamodb\_client = getData(id = debug\_username)  debug\_response = dynamodb\_client.get()  debug\_user\_data = debug\_response['Item'] or None  # add record to database test-v2  record = putData(id = debug\_username,face\_id=debug\_id,conf=debug\_sim,location='test location')  record.put()    if conf and conf >= minSim:  result\_username = dump.split('\n')[0]  result\_sim = conf  result\_id = id  result\_dis = c  dynamodb\_client = getData(id = result\_username)  result\_response = dynamodb\_client.get()  result\_user\_data = result\_response['Item'] or None      response = make\_response({  "debug": {  "id": debug\_id,  "dis": debug\_dis,  'sim': debug\_sim,  "username": debug\_username,  "user\_data": debug\_user\_data,  'expect\_user': expect\_username  },  'result': {  "username": result\_username,  "dis": result\_dis,  "sim": result\_sim,  "id": result\_id,  "user\_data": result\_user\_data  },    })  response.headers['Content-Security-Policy'] = 'upgrade-insecure-requests'  return response    return 'kosong'    # if image is None:  # value = {  # statusCode : 400,  # message : 'Image required!',  # }  # return json.dumps(value)  # model = Predict(image)  # id, conf = model.predict()  # response = {  # statusCode : 200,  # data : {  # id : id,  # conf : conf,  # status : success,  # }  # }  # return json.dumps(response)  @app.route('/recognize/train', methods=['POST'])  def train():  # grab data from post request  request\_data = request.get\_json()  key = None  username = None  # config  face\_cascade = 'lib/haarcascade\_frontalface\_default.xml'  dataset\_name = 'dataset/'  samples = 15  file\_name = 'model.yml'  video\_path = 'video/{}.mp4'.format(uuid.uuid4())      # LBPH variables  radius = 1  neighbour = 8  grid\_x = 8  grid\_y = 8  treshold = 140  var = list([radius,neighbour,grid\_x,grid\_y,treshold])  if request\_data:  if 'key' in request\_data:  key = request\_data['key']    if 'username' in request\_data:  username = request\_data['username']  if 'from' in request\_data:  status = request\_data['from']  if status =='development':  bucket\_name = 'skripsi200053-dev'  else:  bucket\_name = 'skripsi132739-prod'    # download video from s3  downloader\_client = downloader(key=key,bucket=bucket\_name, destination=video\_path)  downloader\_client.download()  # get data user  dynamodb\_client = updateData(id = username, key = key)  ress = dynamodb\_client.update()  print(ress)  # start recognize using opencv  model = Train(face\_cascade,var,username) # create instance train  video = cv2.VideoCapture(video\_path) # load video  model.createDataset(samples,video,dataset\_name) # create dataset  # id = model.train(dataset\_name,file\_name)  faces,faceID = re.labels\_for\_training\_data('dataset')  face\_recognizer=re.train\_classifier(faces,faceID)  face\_recognizer.save(f'{os.path.dirname(os.path.realpath(\_\_file\_\_))}/model.yml')  print(f'faces: {len(faces)} , id: {len(faceID)}')  response = {  "success": True,  "face\_id": id,  "username": username  }  response = make\_response({  "success": True,  "face\_id": id,  "username": username  })  response.headers['Content-Security-Policy'] = 'upgrade-insecure-requests'  return response  else:  return 'Key required!'  @app.route('/update', methods=['POST'])  def update():  request\_data = request.get\_json()  AttrName = {}  AttrValue = {}  Expression = []  if request\_data:  if 'id' in request\_data:  id = request\_data['id']    if 'name' in request\_data:  # key = request\_data['name']  AttrName['#N'] = 'name'  AttrValue[':N'] = { 'S' : request\_data['name']}  Expression.append('#N = :N')  if 'phone' in request\_data:  # status = request\_data['phone']  AttrName['#P'] = 'phone\_number'  AttrValue[':P'] = { 'S' : request\_data['phone']}  Expression.append('#P = :P')  if 'address' in request\_data:  # status = request\_data['address']  AttrName['#A'] = 'address'  AttrValue[':A'] = { 'S' : request\_data['address']}  Expression.append('#A = :A')    expression = 'Set ' + ','.join([str(elem) for elem in Expression])  dynamodb\_client = updateDataNew(id)  ress = dynamodb\_client.update(AttrName,AttrValue, expression)  response = make\_response({  "res": request\_data,  'AttrName': json.dumps(AttrName),  'AttrValue': json.dumps(AttrValue),  'Expression': expression    })  response.headers['Content-Security-Policy'] = 'upgrade-insecure-requests'  return response  if \_\_name\_\_ == '\_\_main\_\_':  # run app in debug mode on port 5000  app.run(host='0.0.0.0.0',port=80) |

### Pedoman Pengoprasian Perangkat Lunak

Pedoman pengoperasian berfungsi tatacara untuk membantu pengguna dalam menggunakan sebuah sistem, pedoman lebih membatu pengguna dibandingkan harus mencoba semua menu yang ada. Adapun pedoman dalam menggunakan *website* Absensi Online Radar Cirebon setelah diimplementasikan dengan teknik pengenalan wajah adalah sebagai berikut:

* + - 1. Proses Pendaftaran
         1. Masukan *email* yang akan didaftarkan.
         2. Masukan *password* yang akan anda gunakan, pastikan *password* harus merupakan *alphanumeric.*
         3. Setelah pendaftaran pengguna harus mengkonfirmasi akun menggunakan kode yang diberikan ke *email* yang terdaftar.

Berikut adalah tampilan halaman pendaftaranyang dapat dilihat pada Gambar 4.17:



Gambar 4. 17 Halaman Pendaftaran

* + - 1. Proses *Login*
         1. Masukan *email* dan *password* yang telah terdaftar.
         2. Klik tombol masuk
         3. Jika *email* dan *password* benar, maka akan dialihkan ke halaman *dashboard*.

Berikut adalah tampilan halaman *login* yang dapat dilihat pada Gambar 4.18:



Gambar 4. 18 Halaman Login

* + - 1. Proses Menambahkan Informasi Pengguna

Untuk menambahkan informasi pengguna maka pilih menu *setting.* Pada halaman *setting* berisi informasi dari perngguna seperti nama, alamat *email,* alamat dan nomer telepon. Di halaman ini juga pengguna bisa memasukan data wajahnya yang berupa file video. Tampilan halaman *setting* dapat dilihat pada Gambar 4.19 berikut:



Gambar 4. 19 Tampilan Setting

* + - 1. Proses Absensi

Untuk melakukan absensi maka pilih menu absensi*.* Pada halaman absensi tedapat tombol untuk mengunggah foto dan kamera yang bisa digunakan untuk menagkap gambar wajah pengguna. Tampilan halaman absensi dapat dilihat pada Gambar 4.20 berikut:



Gambar 4. 20 Halaman Absensi

# KESIMPULAN DAN SARAN

## Kesimpulan

## Saran

# DAFTAR PUSTAKA

[1] “Indonesia: Coronavirus Pandemic Country Profile - Our World in Data.” https://ourworldindata.org/coronavirus/country/indonesia (accessed Apr. 30, 2021).

[2] WHO, “Transmisi SARS-CoV-2: implikasi terhadap kewaspadaan pencegahan infeksi,” pp. 1–10, 2020.

[3] V. Bruce and A. Young, “Understanding face recognition,” *Br. J. Psychol.*, vol. 77, no. 3, pp. 305–327, 1986, doi: 10.1111/j.2044-8295.1986.tb02199.x.

[4] Q. Mutiara and E. Prasetyo, “Perbandingan Metode Eigenface, Fisherface, dan LBPH pada Sistem Pengenalan Wajah,” *J. Ilm. Komputasi*, vol. 18, no. 4, 2019, doi: 10.32409/jikstik.18.4.2675.

[5] Sugiyono, *Metode Penelitian Kuantitatif, Kualitatif, dan Tindakan*. 2012.

[6] H. Hasanah, “TEKNIK-TEKNIK OBSERVASI (Sebuah Alternatif Metode Pengumpulan Data Kualitatif Ilmu-ilmu Sosial),” *At-Taqaddum*, vol. 8, no. 1, p. 21, 2017, doi: 10.21580/at.v8i1.1163.

[7] A. Mirzaqon, “Studi Kepustakaan Mengenai Landasan Teori Dan Praktik Konseling Expressive Writing Library,” *J. BK UNESA*, no. 1, pp. 1–8, 2018.

[8] S. Z. Li and A. K. Jain, *Handbook of Face Recognition*. 2005.

[9] H. Jaya *et al.*, *Kecerdasan Buatan*, vol. 53, no. 9. 2018.

[10] A. Ahmed, F. Ali, and A. Ahmed, “LBPH based improved face recognition at low resolution LBPH Based Improved Face Recognition At Low Resolution,” *2018 Int. Conf. Artif. Intell. Big Data*, no. October 2019, pp. 144–147, 2018, doi: 10.1109/ICAIBD.2018.8396183.

[11] A. P. Singh, S. S. Manvi, P. Nimbal, and G. K. Shyam, “Face recognition system based on LBPH algorithm,” *Int. J. Eng. Adv. Technol.*, vol. 8, no. 5 Special Issue, pp. 26–30, 2019.

[12] D. A. Wahyudi and I. H. Kartowisastro, “Menghitung Kecepatan Menggunakan Computer Vision,” *J. Tenik Komput.*, vol. 19, no. 2, pp. 89–101, 2011.

[13] E. Gorelik, “Cloud Computing Models,” 2013.

[14] R. S. Pressman, *Software Quality Engineering: A Practitioner’s Approach*, vol. 9781118592. 2014.

[15] D. S. M. Sc, M. Math, and D. Ph, “LOGIKA PEMROGRAMAN Disusun,” no. September, 2002.

[16] H. Bagir and B. E. Putro, “Analisis Perancangan Sistem Informasi Pergudangan di CV. Karya Nugraha,” *J. Media Tek. dan Sist. Ind.*, vol. 2, no. 1, p. 30, 2018, doi: 10.35194/jmtsi.v2i1.274.

[17] “Data Flow Diagram (DFD)s: An Agile Introduction.” http://www.agilemodeling.com/artifacts/dataFlowDiagram.htm (accessed Jun. 14, 2021).

[18] D. Vassallo and J. Pschorr, “The Good Parts of AWS.”

# LAMPIRAN-LAMPIRAN