BAB III

METODOLOGI PENELITIAN DAN PERANCANGAN SISTEM

3.1 Sistematika Penelitian

Penelitian "Sistem Pengenalan Wajah untuk Presensi dengan Algoritma Convolutional Neural Network" menggunakan enam buah tahap agar metodologi dan perancangan sistem dapat terpenuhi. Tahap-tahap yang digunakan dalam penelitian ini adalah studi literatur, pengumpulan data, analisa dan percangan desain sistem, implementasi, *testing* dan perbaikan, dan penulisan laporan.

1. Studi Literatur

Untuk dapat menyelesaikan tahap ini, dilakukan proses mencari dan membaca jurnal-jurnal penelitian terkait sistem pengenalan wajah. Proses membaca jurnal dilakukan untuk meningkatkan pemahaman teori terkait pengenalan wajah, citra digital, ekstraksi ciri, *deep learning*, algoritma *convolutional neural network*, dan sistem presensi untuk memenuhi landasan teori penelitian.

2. Pengumpulan Data

Data yang dikumpulkan dan digunakan untuk penelitian adalah data citra wajah manusia yang terdiri dari 15 orang yang menjadi label dalam model klasifikasi. Masing-masing dari label memiliki 60 data citra dengan perbedaan ekspresi dan arah pandang wajah tiap citra.

3. Perancangan Aplikasi

Proses perancangan aplikasi dilakukan dengan merancang alur kerja dari sistem presensi, sistem basis data, dan alur antarmuka aplikasi berdasarkan kebutuhan.

4. Implementasi

Proses implementasi terbagi menjadi menjadi tiga proses besar, yaitu melakukan pembuatan model klasifikasi dengan menggunakan bahasa pemrograman Python yang dijelaskan pada sub bab metodologi penelitian, pembuatan aplikasi android *Frontend* dengan *framework* Ionic React, dan membuat integrasi *Backend* antara model, *database*, dan aplikasi *Frontend* yang telah diciptakan dengan *framework Flask* Python.

5. Pengujian dan Perbaikan

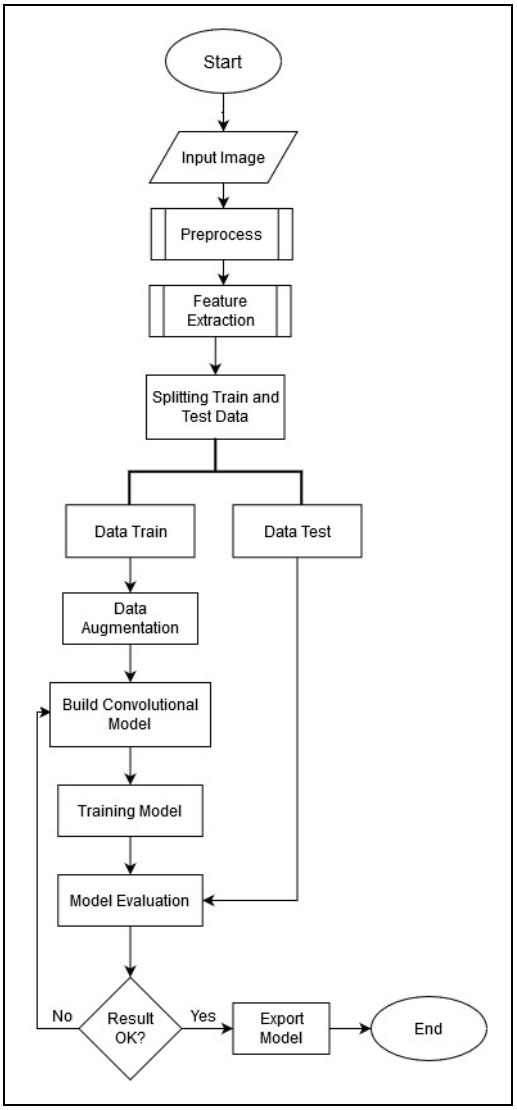
Pengujian hasil pengenalan wajah dapat dilakukan langsung ketika model telah diciptakan dengan memasukkan citra yang di tes dan mengeluarkan hasil klasifikasi berdasarkan label. Proses perbaikan model dapat dilakukan dengan melakukan perbaikan di tahap pra proses. Proses pengujian juga dilakukan terhadap aplikasi langsung dimana diuji mengeluarkan keluaran sesuai dengan yang model klasifikasikan.

6. Penulisan Laporan

Menulis laporan penelitian dari tahap studi literatur hingga pengujian sistem presensi pengenalan wajah.

3.2 Metodologi Penelitian

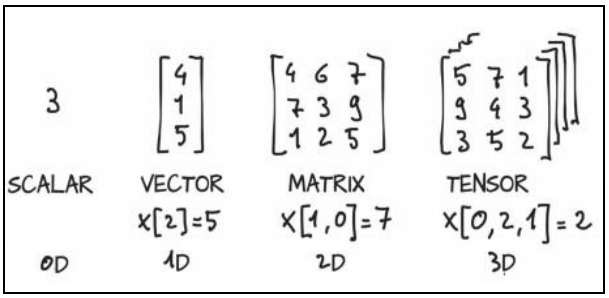
Dalam rangka menyelesaikan penelitian ini, dijelaskan mengenai prosedur sistem pengenalan untuk presensi menggunakan *Deep Learning*. Berikut adalah *flowchart* yang menggambarkan proses kerja hingga implementasi algoritma CNN menggunakan bahasa pemrograman Python yang ditunjuk pada Gambar 3.1.



Gambar 3. 1 Flowchart Alur Implementasi Algoritma CNN

3.2.1 Input Image

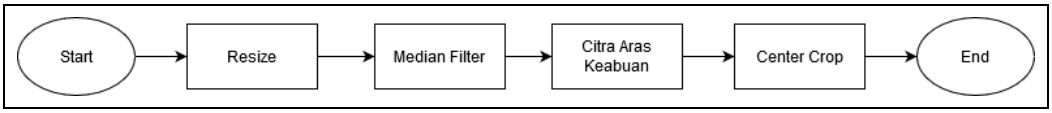
Data citra wajah yang diinput adalah data yang diperoleh dari hasil pengumpulan data. Data citra digital yang diambil dari kamera depan *smartphone* ini memiliki dimensi awal sebesar 3264x2448 *pixel* dengan format .JPG. Proses pengambilan sampel sebanyak 900 citra dari 15 sampel wajah orang Sehingga terbentuk suatu tipe data bertipe *tensor list* yang memiliki dimensi 900x3264x2448x3. *Tensor* merupakan suatu data kumpulan dari matriks yang memiliki ilustrasi komponen pembentuk yang ditunjuk pada Gambar 3.2.



Gambar 3. 2 Komponen Pembentuk *Tensor*
(Chollet, 2018)

3.2.2 Pra proses

Praproses memiliki tujuan untuk mentransformasi data menjadi suatu format dimensi yang layak untuk menjadi penyusun *neural network*. Praproses dapat dilakukan setelah citra digital telah diakuisisi. Pra proses yang dilakukan dalam penelitian kali ini terbagi menjadi empat proses, yaitu *resize image*, *median filter*, *grayscale*, dan *center crop*. Flowchart alur dari praproses ditunjuk pada Gambar 3.3.



Gambar 3. 3 Flowchart Pra proses

A. Resize

Resize merupakan salah satu teknik *downsampling*. *Downsampling* sendiri merupakan suatu teknik yang berguna untuk mereduksi ukuran citra dalam data yang bertujuan untuk mengurangi proses komputasi tanpa mengubah hasil output yang diinginkan (Shi, 2016). Setelah gambar diakuisisi dalam tensor bertipe data *list tensor* dan memiliki dimensi 3264x2448x3, selanjutnya dilakukan proses pengubahan dimensi matriks data gambar menjadi lebih kecil 4 kali dari citra awal, sehingga perubahan dimensi menjadi 816x612x3.

B. Median Filter

Median filtering merupakan suatu pengolahan citra yang mengubah suatu nilai *pixel* dengan nilai median yang himpunannya didapat dari *n*x*n* matriks dimana *n* merupakan besar *kernel* matriks filter yang menjadi parameter median filtering dan harus bernilai ganjil. Median filtering merupakan operasi filter *non-linear* dan umumnya digunakan digunakan untuk mengurangi *noise* dan mempertajam suatu tepi pada citra (Chen, dkk., 2015).

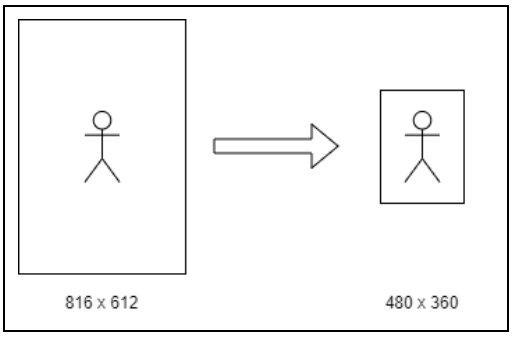
C. Citra Aras Keabuan (Grayscale)

Citra aras keabuan atau *Grayscale* merupakan suatu pengolahan citra yang tujuannya untuk mengurangi dimensi suatu citra yang awalnya tersusun dari tiga dimensi warna RGB menjadi satu. Untuk mendapatkan citra dengan format

grayscale yang sesuai, maka diperlukan suatu bobot masing-masing komponen warna citra digital yang tepat. Masing-masing komponen warna citra digital memiliki suatu intensitas yang berbeda-beda sehingga komposisi dari grayscale yang digunakan seperti W_R = 0.299, W_G= 0.587, dan W_B = 0.114. Warna hijau memiliki bobot paling tinggi sebab warna tersebut memberikan intensitas cahaya yang lebih besar dibandingkan warna merah dan biru ketika ditampilkan dalam pencahayaan yang sama (Montaña, 2017).

D. Center Crop

Pada tahap ini dilakukan pemotongan pada data citra, sebab data yang dibutuhkan berada sekitar tengah horizontal dan vertikal. Tujuannya adalah membantu proses ekstraksi ciri yang melakukan segmentasi agar *background* pada citra tidak mempengaruhi proses deteksi tepi wajah. Ukuran hasil pemotongan gambar menjadikan gambar berukuran 480x360. Gambar 3.4 menunjukkan gambaran hasil proses crop center.

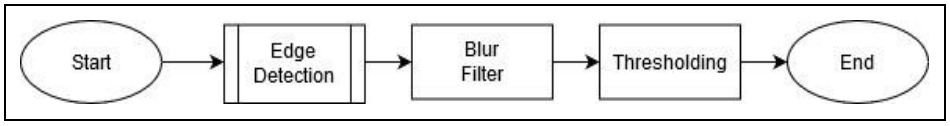


Gambar 3. 4 Ilustrasi Crop Center

3.2.3 Ekstraksi Ciri

Ekstraksi ciri merupakan proses untuk melakukan reduksi pada data namun tidak mengurangi ciri unik dan mengurangi informasi yang redundansi

pada data. Ekstraksi yang digunakan untuk melakukan implementasi terbagi menjadi tiga proses, yaitu *edge detection*, *blur filter*, dan *thresholding*. Flowchart alur dari ekstraksi ciri ditunjuk pada Gambar 3.5.



Gambar 3. 5 Flowchart Ekstraksi Ciri

A. Edge Detection

Untuk mendapatkan mengolah suatu data citra yang dapat diolah dalam proses training model, dibutuhkan suatu data yang ringan agar proses komputasi training model tidak begitu lama. Salah satu cara untuk mengolah data menjadi lebih ringan ini adalah dengan melakukan deteksi tepi. Hasil dari proses ini adalah mengembalikan data gambar dengan nilai biner. Pada penelitian ini, deteksi tepi yang digunakan adalah deteksi tepi Canny.

Proses dari deteksi tepi Canny telah dibahas sebelumnya di bab sebelumnya. Untuk menerapkan deteksi tepi ini dibutuhkan dua parameter dengan tipe data integer yang mempengaruhi hasil deteksi tepi. Dua nilai yang akan dimasukkan adalah nilai minimum dan maksimum diterapkan ketika proses *hysteresis thresholding*. Nilai yang dimasukkan adalah 31 dan 95 dan hasilnya mampu melakukan deteksi tepi hingga keseluruhan wajah tersegmentasi dengan baik.

B. Blur Filter

Blur filter merupakan salah satu proses pengolahan citra yang digunakan untuk membuang *noise* pada citra dan detail kaku pada citra. Salah satu tekniknya

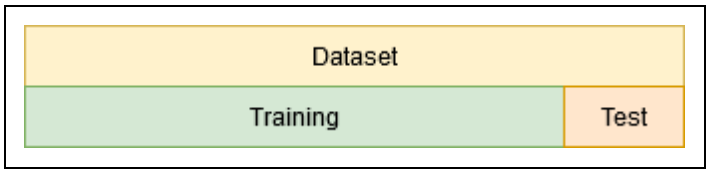
adalah dengan menggunakan *Gaussian Filter*. Tujuan diterapkannya *filter* untuk dapat memperbagus hasil deteksi tepi. Untuk dapat menerapkan *Gaussian Filter* ini dibutuhkan suatu masukan parameter yang tipe datanya integer, masukkan merupakan bilangan desimal dengan nilai ganjil, yang menjadi nilai besar *kernel Gaussian Blur* diterapkan ke suatu citra, sehingga hasil data tidak lagi bernilai biner, dan perlu dilakukan *thresholding* lagi.

C. Thresholding

Thresholding merupakan salah satu tahap akhir dalam pengolahan citra dimana tujuannya mengubah data komponen citra yang sebelumnya berbentuk bilangan desimal dengan *range* 0 sampai 255 menjadi nilai yang 0 atau 255. *Thresholding* yang digunakan adalah *otsu thresholding* sebab telah banyak digunakan dan sukses untuk pemecahan masalah *thresholding* yang seimbang.

3.2.4 Pembagian Data Train Test

Dalam ranah *machine learning*, untuk dapat melakukan prediksi, dibutuhkan model klasifikasi yang berpedoman dari dataset yang telah diolah dan label dari satuan dataset. Dataset yang digunakan untuk pembentukkan model dibagi menjadi dua bagian berdasarkan tujuannya yaitu, data train dan test. Ilustrasi dari pembagian data ditunjuk dalam Gambar 3.6.



Gambar 3. 6 Splitting Data

Agar pembagian data dan label dapat dilakukan secara berpasangan dan sesuai, terdapat suatu fungsi yang berasal dari *library sklearn* yang diimplementasikan. Fungsi tersebut bernama train_test_split menerima tipe data seperti, *list, numpy array*, dan *pandas dataframes*. Berikutnya train_test_split memiliki lima parameter yang mempengaruhi hasil pembagian data yaitu sebagai berikut.

A. Test Size

Parameter yang menerima bilangan *float* yang nilainya antara 0.0 dan 1.0, nilai ini merupakan representasi proporsi dari dataset yang dibagikan menjadi data test. Jika tidak diisi maka nilai *test size* jadi 0.25.

B. Train Size

Parameter yang menerima bilangan *float* yang nilainya antara 0.0 dan 1.0, nilai ini merupakan representasi proporsi dari dataset yang dibagikan menjadi data training. Jika tidak diisi maka terisi bernilai komplemen dari ukuran data test.

C. Random State

Parameter yang menerima secara optional tipe data integer, yang digunakan untuk men-*generate* angka random, dalam pembagian data, sehingga nilai random state ini dapat menjadi referensi jika ingin kembali ke suatu hasil terbaik.

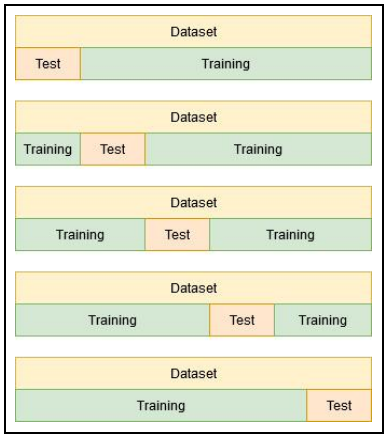
D. Shuffle

Parameter boolean, melakukan pengacakan dataset jika true, jika bernilai false maka parameter *Stratify* harus kosong.

E. Stratify

Parameter ketika melakukan pembagian dataset, data beserta class label diacak sebagai satu kesatuan.

Kemduain untuk dapat memperbaiki proses pembagian data, terdapat salah satu teknik *tuning* atau evaluasi hasil model dengan melakukan *K-Fold Cross Validation*. Teknik ini dapat menjadi suatu mencari pembagian data untuk model terbaik dengan mengetes hasil dari tiap kombinasi pembagian data (*fold*) sebanyak nilai parameter bernama *n_split*, yang menandakan jumlah kombinasi *folding* pembagian data dilakukan proporsi dari data *test* yang akan dibagi berjumlah sebanyak total data dibagi nilai *n_split*. Pada penelitian ini jumlah *folding* yang digunakan adalah 10. Ilustrasi dari proses *K-Fold Cross Validation* ditunjuk dalam Gambar 3.7



Gambar 3. 7 Ilustrasi *K-Fold Cross Validation*

3.2.5 Penyusunan Layer Convolution

Penyusunan *Layer Convolution* bertujuan untuk dapat menciptakan model yang di-*train*. Proses penyusunan *layer* dilakukan dengan membentuk suatu variabel model sekuensial dengan *library keras*. Model yang tersusun ini terbagi menjadi dua *layer* utama yaitu, *extraction layer* dan *fully connected layer*.

3.2.6 Augmentasi Data

Augmentasi data merupakan proses pengambilan suatu sampel yang dilakukan ketika *training* model. Tujuan augmentasi data adalah menciptakan data citra tambahan dari suatu data *train* dengan memodifikasi tampilan citra. Tujuannya adalah untuk ketika dilakukan *training*, model tidak mudah menemukan data sama dua kali, hal ini dilakukan agar model tercipta dari data yang menyeluruh sehingga mencegah *overfitting* dan digunakan untuk mencegah model yang di-*train* tidak kekurangan data untuk *training* model atau bisa disebut *underfitting* (Chollet, 2018).

Implementasi augmentasi data menerapkan fungsi ImageDataGenerator dari *library* Keras. Untuk dapat mengimplementasikan fungsi ini, diperlukan banyak parameter aktivasi fungsi secara acak diperlukan, sehingga mempengaruhi hasil. Beberapa parameter yang digunakan dalam implementasi ini adalah sebagai berikut:

A. Rotation Range

Menerima masukan data tipe integer dengan suatu range 0-180. Berguna untuk secara acak dengan nilai yang menjadi jangkauan yang telah dimasukkan untuk

merotasikan suatu gambar sebesar hasil random sudut. Pada implementasi kali ini, nilai *rotation range* yang dimasukkan adalah 45.

B. Width Shift and Height Shift Range

Menerima masukan data tipe *float* dengan suatu range 0-1. Berguna untuk secara acak dengan nilai jangkauan yang telah dimasukkan melakukan translasi gambar secara vertikal atau horizontal. Pada implementasi kali ini, nilai parameter yang dimasukkan adalah 0.15.

C. Shear Range

Menerima masukan data tipe *float* dengan suatu range 0-1. Berguna untuk secara acak dengan nilai jangkauan yang telah dimasukkan memodifikasi *linear map* citra dengan kemiringan sesuai nilai parameter dan melakukan suatu pemotongan citra.

D. Zoom Range

Menerima masukan data tipe *float* dengan suatu range 0-1. Berguna untuk secara acak dengan nilai jangkauan yang telah dimasukkan melakukan *zoom in* pada suatu citra.

E. Horizontal Flip

Menerima masukan suatu data tipe *boolean* dimana secara acak membalikkan citra secara horizontal atau melakukan rotasi pada citra sebesar 270^{o}.

3.2.7 Model Training

Untuk dapat melakukan train pada model, *method* fungsi pada class variable model yang perlu dipanggil adalah fit_generator yang berasal dari *library* Keras. Kemudian, untuk dapat menjalankan *method* pembangunan model ini dibutuhkan banyak parameter seperti data dan label *train*, jumlah *epochs*, data dan label *test*, *batch_size* dan masih banyak lagi.

Jumlah *epoch* yang digunakan untuk sekali training pada implementasi ini adalah 100 *epochs*, sedangkan ukuran dari *batch size* yang mampu diimplementasikan 16. Hasil dari tahap ini ada output *file* dengan format .hdf5 yang digunakan untuk pengklasifikasian atau prediksi terhadap citra.

3.2.8 Evaluasi

Metode evaluasi yang digunakan adalah metode akurasi dan *F-score*. Metode akurasi adalah metode evaluasi yang mempertimbangan seluruh hasil benar atau tidak. Metode digunakan untuk melihat hasil implementasi dengan hasil kebenaran terbagi menjadi empat nilai. Matriks penyusun empat nilai terkait adalah sebagai berikut:

Tabel 3. 1 Matriks Hasil Pengujian (Prasetyo, 2014)

		Kelas hasil prediksi	
		Positif	Negatif
Kelas asli	Positif	True Positive (TP)	False Negative (FN)
	Negatif	False Positive (FP)	True Negative (TN)

Keterangan:

- True Positive: Teridentifikasi secara benar
- True Negative: Tertolak secara salah
- False Positive: Teridentifikasi secara salah
- False Negative: Tertolak secara salah

Untuk menghitung nilai akurasi dibutuhkan suatu jumlah hasil pengujian setiap subjek. Berikut merupakan rumus dari akurasi (Alvarez, 2002).

$$Akurasi = \frac{true\ positive + true\ negative}{true\ positive + false\ positive + true\ negative + false\ negative} \quad \ldots(13)$$

Metode *F-score* merupakan metode evaluasi akurasi yang mempertimbangankan presisi dan *recall* dari tes. Presisi merupakan tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem. Sedangkan *recall*, merupakan tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi. Kemudian, nilai dari *F-score* sendiri adalah rata-rata perbandingan nilai presisi dan *recall* dari hasil. Berikut merupakan rumus dari presisi, *recall*, dan *F-score*. (Sokolova, 2006).

$$Precision = \frac{\text{true positive}}{true\ positive + false\ positive} \quad \ldots(14)$$

$$recall = \frac{\text{true positive}}{true\ positive + false\ negative} \quad \ldots(15)$$

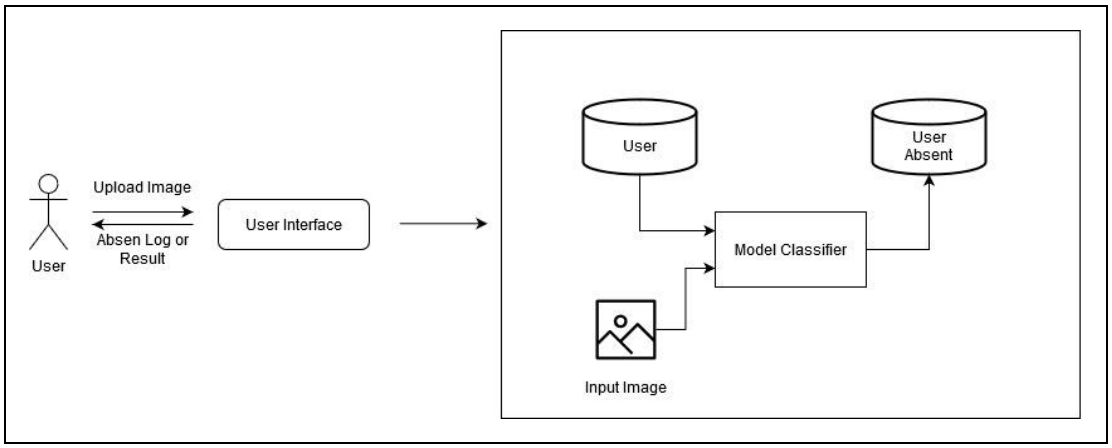
$$F\text{-}score = 2 * \frac{\text{precision} * \text{recall}}{precision + recall} \quad \ldots(16)$$

3.3 Perancangan Aplikasi

Sistem presensi pengenalan wajah dengan algoritma *convolutional neural network* sebagai pembentuk model klasifikasi memiliki perancangan dalam membangun sistem. Perancangan sistem yang digunakan untuk membangun sistem pengenalan wajah untuk presensi adalah berupa model sistem, *sitemap*, *flowchart*, *source code* , *database schema*, struktur tabel, dan desain antarmuka.

3.3.1 Model Sistem

Gambar 3.8 merupakan model sistem dari sistem pengenalan wajah untuk presensi yang dikembangkan.



Gambar 3. 8 Model Sistem Presensi Pengenalan Wajah

Adapun elemen-elemen yang terdapat pada model sistem dalam Gambar 3.9 adalah sebagai berikut.

1. User

User adalah pengguna biasa dari sistem presensi yang wajahnya telah dijadikan dataset dalam model pengklasifikasi yang ada dalam sistem.

2. User Interface

User Interface adalah tampilan halaman dari sistem presensi pengenalan wajah. Tampilan halaman dapat diakses oleh *user* untuk dapat melakukan *login*, melakukan absen, dan mengecek log *history* hasil absen. Pembuatan *User Interface* dibuat dengan *framework frontend* nodeJS Ionic-Angular.

3. Database

Database adalah basis data untuk menyimpan informasi terkait hasil absen yang telah dilakukan tiap *User* dan menyimpan data kredensial *User* seperti *username* dan *password*.

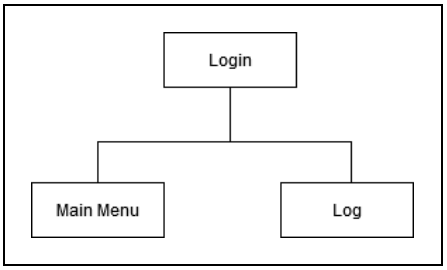
4. Input Image

Input image merupakan media citra digital yang dimasukkan *user* agar dapat dilakukan proses pengklasisikasi dari model

5. Model Classifier

Model *Classifier* merupakan suatu model hasil training dengan algoritma CNN, yang digunakan untuk melakukan prediksi dengan mengolah masukkan citra digital dan mengeluarkan output berupa label. Label yang dihasilkan digunakan untuk pencocokan terhadap *user*.

3.3.2 Sitemap



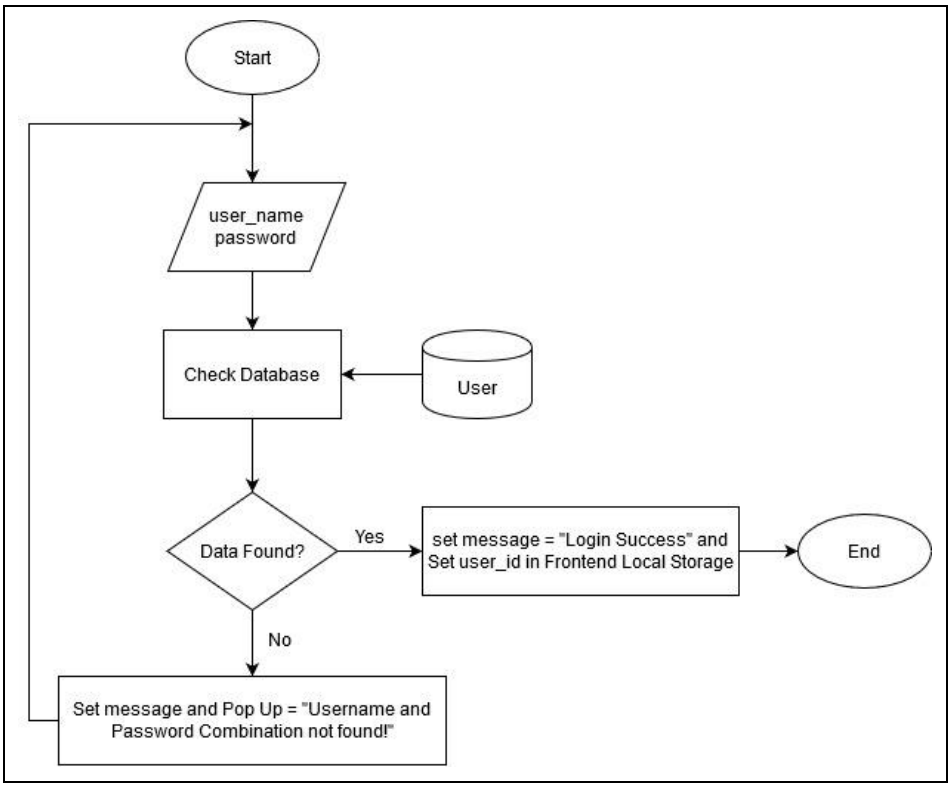
Gambar 3. 9 Sitemap User

Gambar 3.9 merupakan *sitemap* untuk pengguna. User dalam mengakses sistem presensi pengenalan wajah akan langsung diarahkan ke halaman *login*. Pada halaman *login*, *user* dapat memasukkan *username* dan *password*. Jika pasangan *username* dan *password* sesuai maka *user* diarahkan ke halaman Main Menu. Halaman main menu membuat *user* dapat melakukan absen dengan melakukan pengambilan citra melalui kamera atau folder dalam android, selain itu pada halaman ini terdapat *sidebar* yang membuat *user* dapat mengakses halaman log absen *user* sendiri dan melakukan *logout*. Pada halaman log, *user* dapat melihat hasil absen yang telah dicatat dalam *database*.

3.3.3 Flowchart

Penjelasan mengenai sistem *flowchart* dibagi menjadi tiga bagian berdasarkan jumlah fitur aplikasi sistem presensi pengenalan wajah sebagai berikut.

A. Flowchart Login

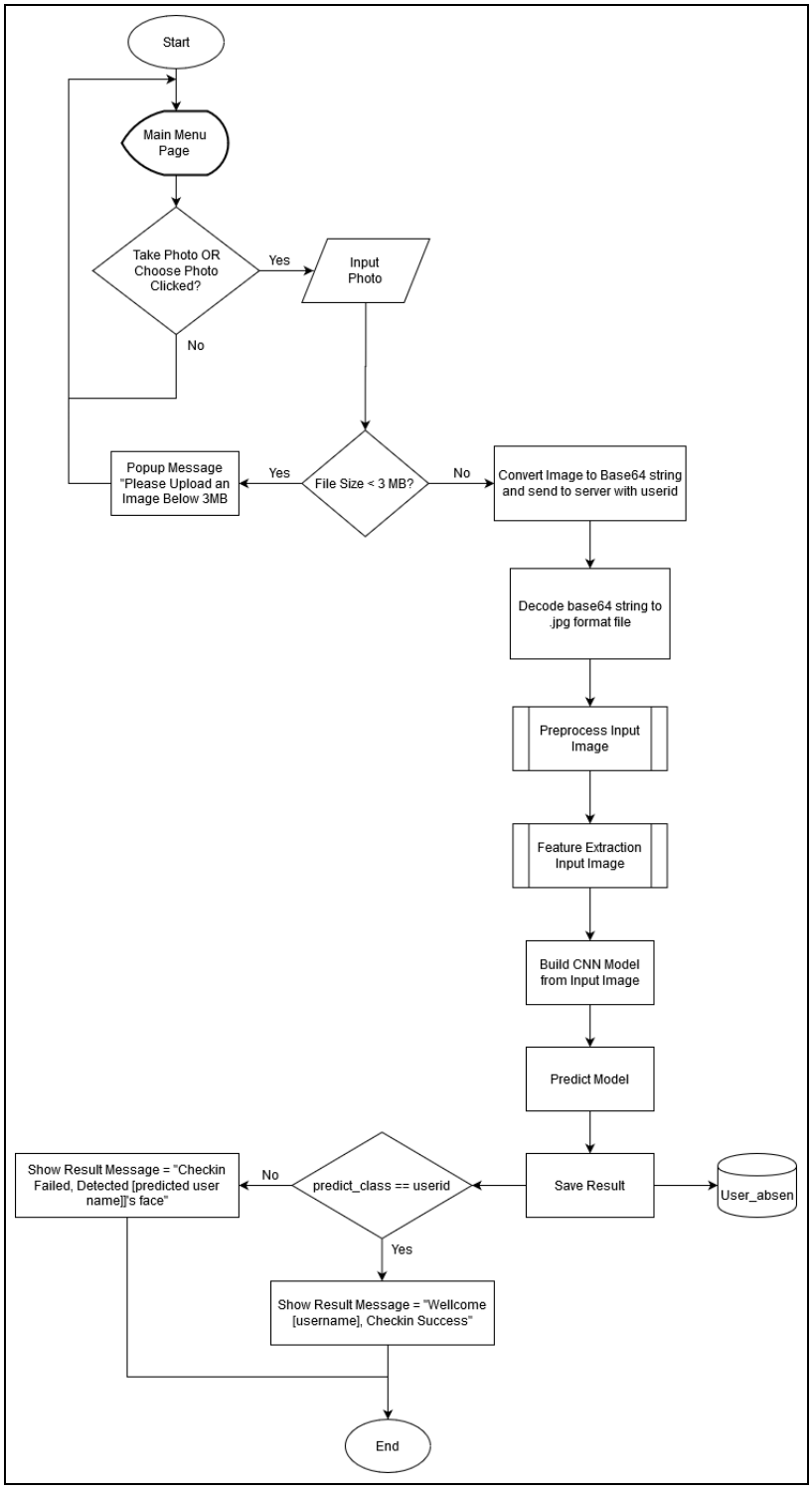


Gambar 3. 10 Flowchart Login

Gambar 3.10 adalah perancangan *flowchart* dari proses *login*. Dalam proses *login*, pengguna memasukkan data seperti *username* dan *password*. Aplikasi akan melakukan pencarian ke basis data berdasarkan kombinasi *username* dan *password* yang dimasukkan.

Aplikasi akan mengembalikan pesan respon berupa “Username and Password Combination not Found!” ketika kombinasi dar *username* dan *password* miliki suatu *user* tidak ditemukan. Jika ditemukan maka nilai yang dikembalikan adalah *userid* dari *user* untuk dapat di *set* ke dalam *Local Storage* aplikasi sebagai tanda *login* telah berhasil dan berganti halaman *Main Menu*.

B. Flowchart Main Menu

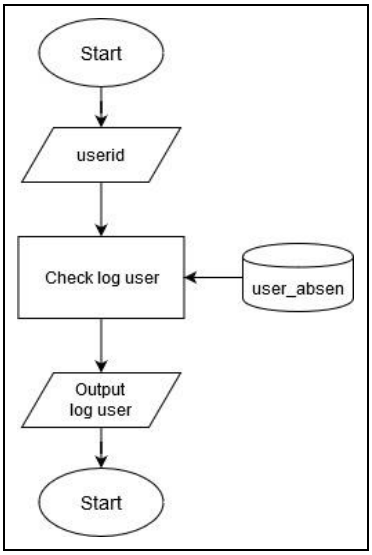


Gambar 3. 11 Flowchart Main Menu

Gambar 3.11 adalah *flowchart* dari prosedur fitur utama aplikasi yang berada dalam halaman utama berjalan. Mula-mula setelah *user* menerima antarmuka halaman, *user* dapat mengambil foto dengan dua cara yaitu, mengambil melalui *file* yang ada dalam perangkat android dan mengambil melalui kamera foto. Setelah itu, dilakukan pengecekan ukuran foto, aplikasi menolak foto jika ukuran foto diatas 3 *Megabytes*, foto yang telah diinput di-*encode* menjadi format *string base64*.

Data yang dikirimkan ke *backend* dalam proses ini adalah hasil *encode string base64* dan *userid*. Hasil string *base64* yang diterima *backend* di-*decode* ke dalam format *file* .jpg, agar dapat dilakukan pemrosesan citra seperti pra proses, ekstraksi fitur, dan pembentukan model CNN dengan cara yang sama dilakukan ketika membentuk suatu model *training* agar dapat dilakukan prediksi suatu label. Kemudian, label yang dihasilkan dari prediksi merupakan *userid*, hasilnya dapat mengeluarkan *userid* milik *user* lain, namun jika *userid* hasil prediksi dan *userid* yang diterima *backend* sama maka proses pengenalan wajah benar dan presensi tervalidasi. Terakhir dilakukan pencatatan data *timestamp* dan status presensi ke dalam basis data lalu mengembalikan pesan respon.

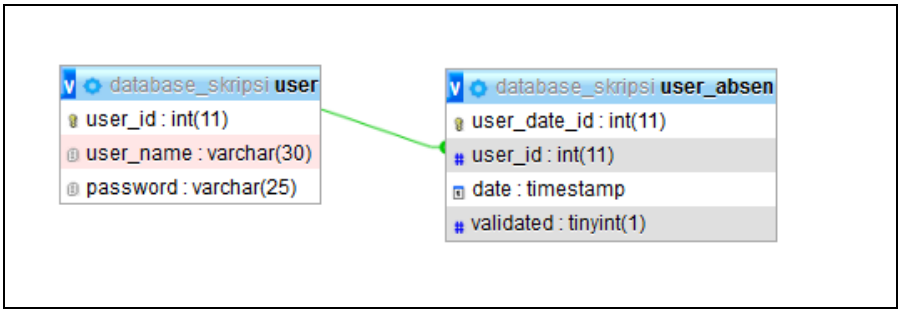
C. Flowchart User Log



Gambar 3. 12 Flowchart User Log

Gambar 3.12 adalah *flowchart* dari *user log* untuk melihat data *user* yang telah melakukan absen. Dengan mengirim suatu data *userid*,maka akan mengembalikan data *user log* sesuai dengan *user* yang sedang *login*.

3.3.4 Database Schema



Gambar 3. 13 Database Schema Sistem Presensi

Gambar 3.13 merupakan gambar *database schema* dari sistem presensi sederhana yang dikembangkan. Pada sistem presensi ini hanya terdapat suatu relasi antara *user* dan *user_absen* dengan *foreign key* kolom *user_id*. Relasinya

adalah satu *user* dapat memiliki banyak *user_date_id* atau dapat disebut *one to many*.

A. Tabel User

Tabel 3.2 merupakan struktur tabel *User*. Tabel *User* menyediakan informasi akun *user*.

Tabel 3. 2 Struktur Tabel User

No.	Nama Kolom	Tipe	Panjang	Keterangan
1	user_id	int	11	*Primary key*
2	user_name	varchar	30	Nama pengguna untuk melakukan *login*
3	password	varchar	25	*Password* pengguna

B. Tabel User Absen

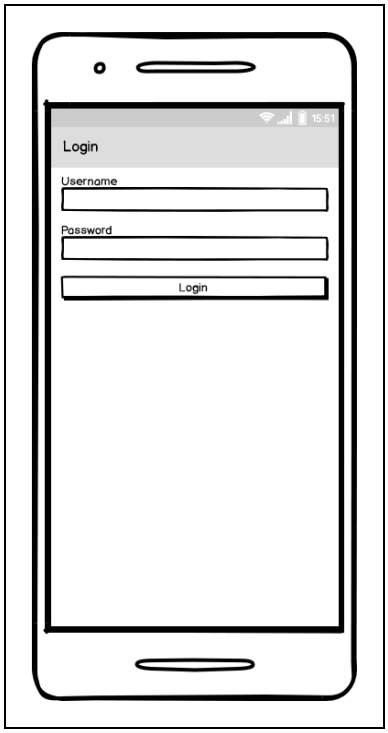
Tabel 3.3 merupakan struktur tabel *User Absen*. Tabel *User Absen* menyediakan data hasil presensi *user*.

Tabel 3. 3 Struktur Tabel User Absen

No.	Nama Kolom	Tipe	Panjang	Keterangan
1	user_date_id	int	11	*Primary key*
2	user_id	int	11	*Foreign key* untuk referensi dengan tabel *User*
3	date	timestamp		*Timestamp*
4	validated	tinyint		Status dari hasil absen, 0 jika user_id dan label hasil *predict* tidak sesuai dan 1 jika sesuai

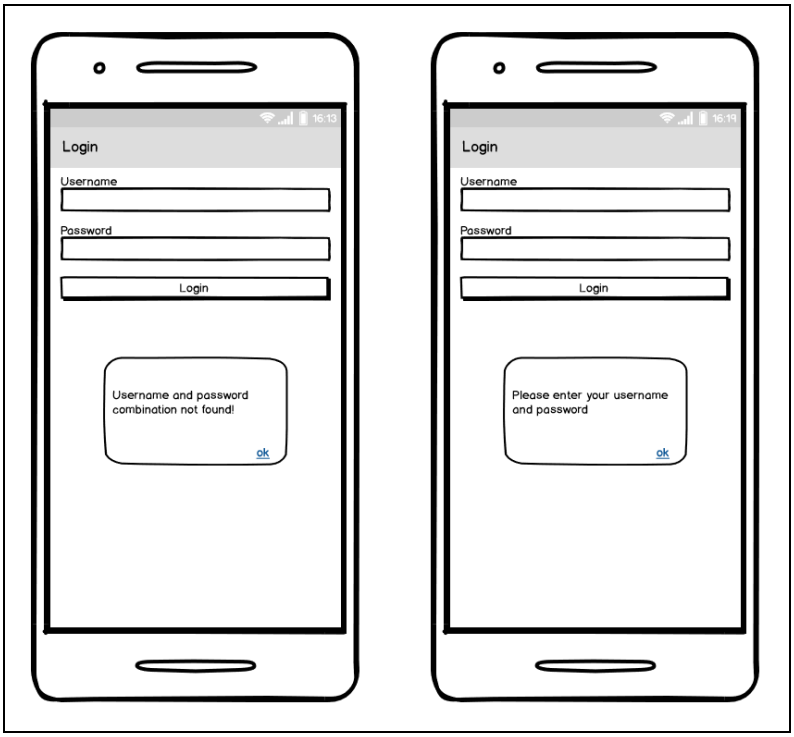
3.3.5 Desain Antarmuka

Desain antarmuka dari sistem pengenalan wajah untuk presensi merupakan salah satu perancangan dari sistem yang digunakan untuk memberikan gambaran tampilan dari aplikasi.



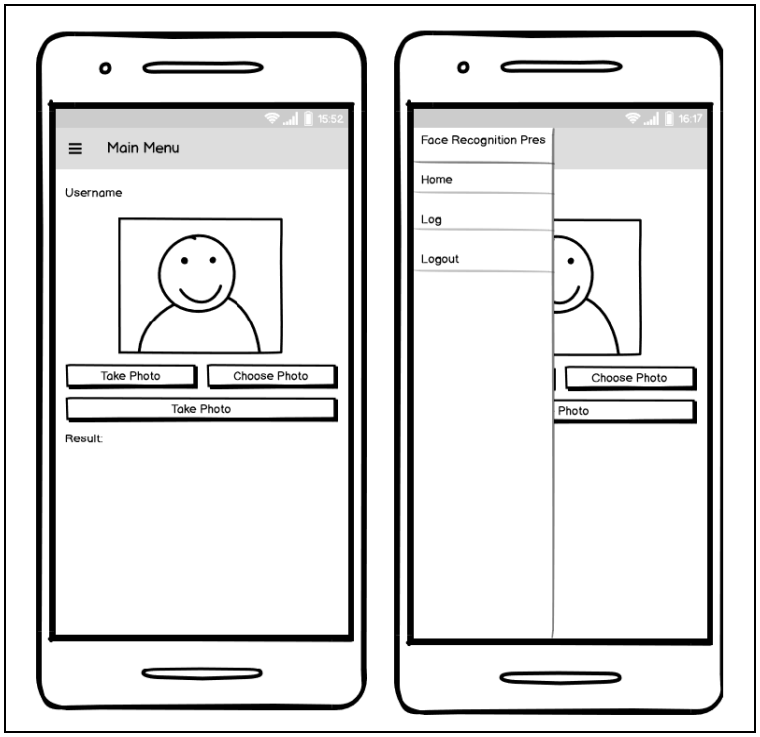
Gambar 3. 14 Desain Antarmuka Login Page

Gambar 3.14 merupakan desain antarmuka dari *login page* aplikasi. Pengguna menerima halaman ini ketika mulai menggunakan sistem. Pada *login page* ini pengguna dapat mengisi input teks *username* dan *password*. Gambar 3.16 menampilkan suatu *popup* pada *login page* bila salah satu atau kedua input teks belum diisi dan kombinasi *username* dan *password* tidak ditemukan di basis data



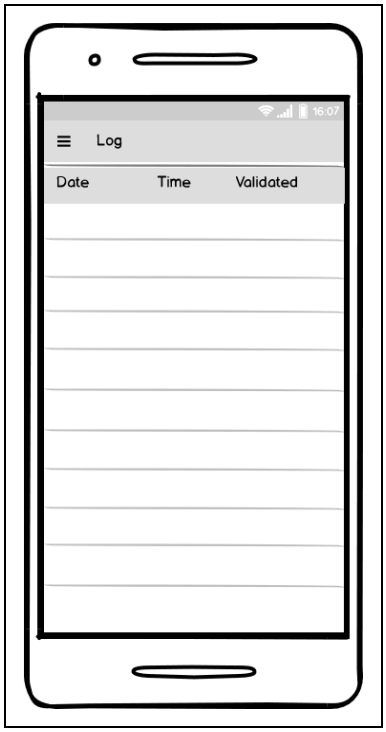
Gambar 3. 15 Desain Antarmuka Popup di Login Page

Kemudian, jika input teks diisi dengan *username* dan *password* yang sesuai maka pengguna menuju ke halaman utama, dan sistem akan memasukkan *userid* sesuai pengguna yang melakukan *login* ke dalam *local storage* yang terdapat dalam aplikasi yang digunakan untuk melakukan pengambilan data terkait satu pengguna dan validasi terhadap hasil prediksi.



Gambar 3.16 Desain Antarmuka Main Menu serta Sidebar

Gambar 3.16 merupakan desain antarmuka dari halaman utama, pengguna selain dapat mengunggah foto citra, pengguna juga dapat berpindah ke halaman *log* dan *login* dengan mengklik button yang ada di sudut kiri atas antarmuka dan akan muncul suatu *sidebar* berisi daftar halaman yang dapat diakses ketika diklik.



Gambar 3. 17 Desain Antarmuka Halaman User Log

Gambar 3.17 merupakan desain antarmuka dari halaman *user log*, pada halaman ini pengguna dapat melihat data absen pengguna itu sendiri. Data yang dicatat adalah tanggal, jam dan status tervalidasinya absen berdasarkan pencocokan foto dengan model.

.