PERANCANGAN PERANGKAT LUNAK PENGENALAN WAJAH DENGAN METODE *CONVOLUTIONAL NEURAL NETWORK* UNTUK SISTEM KEHADIRAN

TUGAS AKHIR
Diajukan untuk Memenuhi Salah Satu Syarat Kelulusan
Program Pendidikan Sarjana

Oleh:
Caesar Evan Hans Christian
2020130055

JURUSAN INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER – LIKMI
BANDUNG
2022

PERANCANGAN PERANGKAT LUNAK PENGENALAN WAJAH DENGAN METODE *CONVOLUTIONAL NEURAL NETWORK* UNTUK SISTEM KEHADIRAN

Oleh:
Caesar Evan Hans Christian
2020130055

Bandung, 1 Juli 2022
Menyetujui,

Dr. Hery Heryanto, S.Kom., M.Kom.	Dhanny Setiawan, S.T., M.T.
Pembimbing	Ketua Jurusan

JURUSAN INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER – LIKMI
BANDUNG
2022

ABSTRAK

Perkembangan teknologi digital yang sangat pesat dan pandemi Covid-19 yang melanda dunia membuat tata cara kehidupan berubah, terutama mengurangi intensitas untuk bersentuhan secara langsung maupun tidak langsung. Akhirnya sesuatu yang membantu kita untuk tidak melakukan sentuhan akan lebih efektif seperti identifikasi wajah dalam sistem kehadiran. Maka dilakukan penelitian ini pengenalan wajah pada sistem kehadiran menggunakan metode CNN (*Convolutional Neural Network*). Penelitian ini bermaksud untuk mengimplementasikan metode CNN ke dalam sistem kehadiran.

Sistem kehadiran yang menggunakan pengenalan wajah dibuat dengan bahasa pemrograman *python*. Metode CNN ini akan melakukan proses *training* dari beberapa lapisan yaitu *Conv2D*, *MaxPooling2d*, *Flatten*, dan *Dense* serta menggunakan *Haar Cascade* dengan dibantu *Library OpenCV* untuk mendeteksi wajah. Citra wajah yang digunakan untuk melakukan perekaman kehadiran dengan metode CNN ini diambil secara langsung menggunakan *webcam* dengan percobaan citra wajah diambil dari layar *handphone* yang terdapat wajah yang akan melakukan perekaman kehadiran. Jumlah citra wajah yang digunakan sebanyak 3.732 gambar wajah yang berasal dari 30 himpunan wajah dan menghasilkan tingkat akurasi sebesar 93.904%.

Kata Kunci: *Convolutional Neural Network*, *Deep Learning*, *Face Recognition*, Sistem Kehadiran.

KATA PENGANTAR

Salam Sejahtera.

Terima kasih kepada Tuhan yang Maha Esa karena dengan bimbingan-Nya dan karunia-Nya, penulis dapat menyelesaikan Tugas Akhir yang berjudul: “PERANCANGAN PERANGKAT LUNAK PENGENALAN WAJAH DENGAN METODE *CONVOLUTIONAL NEURAL NETWORK* UNTUK SISTEM KEHADIRAN”, sebagai salah satu syarat kelulusan program studi strata satu di Sekolah Tinggi Manajemen Informatika dan Komputer LIKMI.

Dalam penyusunan Tugas Akhir ini tidak lepas dari bantuan dan dukungan dari berbagai pihak. Maka pada kesempatan ini penulis ingin mengucapkan terima kasih kepada:

1. Bapak Dr. Hery Heryanto, S.Kom., M.Kom. sebagai dosen pembimbing yang bersedia memberikan bimbingan dan arahannya dengan sepenuh hati selama penulis menyelesaikan Tugas Akhir ini.
2. Bapak Dhanny Setiawan, S.T, M.T. selaku Ketua Jurusan Informatika yang telah memberikan banyak ilmu pengetahuan kepada penulis dalam menyelesaikan Tugas Akhir ini.
3. Para staf pengajar STMIK LIKMI yang turut membantu secara tidak langsung dalam memberikan ilmu pengetahuan kepada penulis selama mengikuti masa perkuliahan ini.
4. Keluarga tercinta yang senantiasa memberi dukungan, pengorbanan secara moril maupun materil, serta doa yang tulus tanpa pamrih sehingga penulis mampu menyelesaikan Tugas Akhir ini.
5. Yosef Yunawan, Daniel William, Priska Yuliani yang telah memberikan motivasi, saran, bantuan, maupun dukungan selama penyusunan Tugas Akhir ini.
6. Seluruh sahabat, teman-teman, dan mentor yang selalu memberikan semangat dan motivasi selama penulis menjalani perkuliahan juga dukungan selama penyusunan Tugas Akhir ini.
7. Semua pihak yang tidak dapat disebutkan satu per satu yang telah membantu penulis selama proses penyusunan Tugas Akhir ini.

Penulis menyadari penulisan Tugas Akhir ini masih jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun dari semua pihak.

Bandung, 1 Juli 2022

Penulis

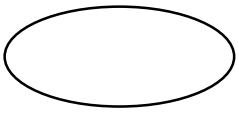
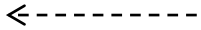
DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SIMBOL

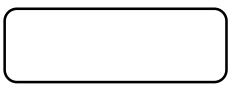
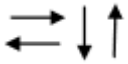
Use case diagram

Nama Simbol	Simbol	Keterangan
Use Case		Sebuah kebiasaan atau fungsi utama yang dilakukan oleh sistem perangkat lunak
Asosiasi		Hubungan antara aktor dengan *use case* yang menunjukkan adanya interaksi aktor dengan sistem
Aktor		Entitas eksternal yang berhubungan dengan sistem
Extend	<<*extend*>> 	Menunjukkan satu *use case* merupakan tambahan fungsional jika *use case* lain terpenuhi

Class diagram

Nama Simbol	Simbol	Keterangan
Asosiasi		Menunjukkan hubungan antar kelas

Activity diagram

Nama Simbol	Simbol	Keterangan
Initial Node		Status awal diagram
Activity final node		Status Akhir sebuah diagram
Action State		Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
Decision		Menggambarkan suatu keputusan atau tindakan diambil pada kondisi tertentu
Line Connector		Menghubungkan satu simbol ke simbol lain

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pada masa ini banyak kemudahan yang bisa didapatkan terutama pada bidang teknologi digital, ditambah pandemi yang membuat banyak pekerjaan dilakukan secara *online* melalui *digital platform* seperti *zoom*, *google meet*, *skype* dan lain-lain; bahkan munculnya *meta verse*. Tidak sedikit juga pekerjaan yang juga tetap dilakukan secara tatap muka sehingga perusahaan atau kantor harus tetap melakukan perekaman kehadiran untuk yang bekerja secara *online* dan yang bekerja tatap muka.

Perekaman kehadiran menggunakan kartu kehadiran cukup ketinggalan zaman, sedangkan perekaman kehadiran menggunakan jari atau *fingerprint* higenitas yang menjadi pertimbangan karena mungkin saja bisa menjadi titik penularan virus atau penyebaran kuman. Maka dari itu, dalam penelitian ini diaplikasikan sistem kehadiran dengan pengenalan wajah, yang membantu untuk mengefektifkan perekaman kehadiran. Perekaman kehadiran menggunakan wajah akan menjadi sebuah solusi sederhana yang membuat perekaman kehadiran menjadi tidak ketinggalan zaman dengan higenitas yang lebih terjamin. Pada kenyataannya juga tidak dapat dipungkiri bahwa cara yang lebih efektif membuat manusia jauh lebih terbantu pada masa ini.

Untuk mengetahui pengenalan wajah pada sistem kehadiran maka dibutuhkan *Computer Vision* untuk mengenali objek wajah yang menjadi fokus objeknya. Objek wajah dipakai menjadi sarana penelitian ini karena setiap wajah manusia mempunyai keunikan tersendiri dari mata, hidung, mulut. Objek wajah juga dipakai sebagai sarana, karena dalam zaman teknologi ini pengenalan wajah sudah banyak diterapkan untuk memudahkan suatu pekerjaan. Penelitian kali ini menggunakan metode *Convolutional Neural Networks* (CNN) dengan korteks mamalia visual sel sederhana dan kompleks. Model ini dapat mengurangi parameter bebas dan dapat menangani deformasi gambar input seperti translasi, rotasi, dan skala. CNN mempunyai kemampuan klasifikasi yang diperuntukan untuk data gambar, sehingga CNN digunakan sebagai pengenalan wajah pada tugas akhir ini.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijabarkan, maka rumusan masalah yang diangkat dalam penyusunan tugas akhir ini:

1. Bagaimana menerapkan metode *Convolutional Neural Networks* pada sistem kehadiran dengan pengenalan wajah?
2. Berapa akurasi penggunaan metode *Convolutional Neural Networks* dalam mengenali wajah?

1.3 Tujuan Penelitian

Tujuan Penelitian ini antara lain:

1. Dapat mengenali objek wajah.
2. Dapat diimplementasi dalam perancangan perangkat lunak.
3. Mendapatkan hasil akurasi deteksi objek wajah yang baik dalam mengenali wajah.

1.4 Batasan Masalah

Agar pembahasan tidak terlalu luas, batasan terhadap masalah yang dipakai:

1. Objek yang dilakukan pendeteksian berupa objek wajah.
2. Citra yang digunakan diambil secara langsung dari gambar wajah melalui *handphone*.
3. Citra yang digunakan diambil dari *website www.pinterest.com*.

1.5 Kegunaan Hasil

Kegunaan dari penelitian ini adalah:

1. Meningkatkan kinerja kerja sistem kehadiran dengan pengenalan wajah.

1.6 Sistematika Penulisan

Pada penelitian ini peneliti menyusun berdasarkan sistematika penulisan sebagai berikut:

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, kegunaan hasil dan sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini menjelaskan teori-teori Rekayasa Perangkat Lunak, UML, *Face Recognition* dan metode *Convolutional Neural Network* yang menunjang penelitian ini.

BAB III ANALISIS DAN PERANCANGAN

Bab ini berisi tentang analisa dan perancangan perangkat lunak yang meliputi analisis data, perancangan proses, dan perancangan perangkat lunak.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Bab ini berisi mengenai pengumpulan data, implementasi metode yang telah dianalisis, dan hasil pengujian terhadap banyak citra yang diuji.

BAB V KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dari penelitian ini serta saran pengembangan untuk penelitian selanjutnya.

BAB II
LANDASAN TEORI

2.1 Rekayasa Perangkat Lunak

Menurut Janner, rekayasa perangkat lunak adalah sebuah profesi yang dilakukan oleh suatu sistem dan pendekatan yang dapat dihitung untuk mengembangkan dan pemeliharaan suatu aplikasi perangkat lunak dengan menerapkan teknologi dan praktik ilmu komputer, manajemen proyek, dan bidang-bidang lainnya (Simarmata, 2010:1).

Menurut Sommverville, rekayasa perangkat lunak adalah suatu disiplin teknik yang berkaitan dengan semua aspek produksi perangkat lunak dari tahap awal spesifikasi sistem sampai pemeliharaan sistem setelah suatu sistem di gunakan (Sommerville, 2011:7).

Berdasarkan teori-teori di atas, maka dapat disimpulkan pengertian dari rekayasa perangkat lunak adalah disiplin ilmu dari teori, metode, dan semua yang berkaitan dengan pembuatan dan pemeliharaan aplikasi perangkat lunak. Rekayasa perangkat lunak dibutuhkan pengalaman dan pelatihan agar menghasilkan hasil yang baik.

2.1.1 Model Pengembangan Perangkat Lunak

Metodologi pengembangan perangkat lunak adalah suatu strategi pengembangan yang memadukan proses, metode, dan perangkat. Metode-metode rekayasa perangkat lunak memberikan teknik untuk membangun perangkat lunak. Berkatian dengan serangkaian tugas yang luas yang menyangkut analisis kebutuhan, konstruksi program, desain, pengujian, dan pemeliharaan (Pressman, 2010:21).

Proses model *software* adalah sebuah representasi yang disederhanakan dari proses *software*. Setiap proses model merupakan proses dari perspektif tertentu, dan dengan demikian hanya menyediakan informasi parsial mengenai proses tersebut. *Generic model* ini tidak menentukan deskripsi dari proses perangkat lunak. Sebaliknya, *generic model* adalah suatu abstraksi dari proses yang dapat digunakan untuk menjelaskan suatu pendekatan yang berbeda untuk mengembangkan perangkat lunak. *Generic model* sebagai kerangka proses yang dapat diperpanjang dan disesuaikan untuk membuat proses rekayasa perangkat lunak yang lebih spesifik (Sommerville, 2011:29).

2.1.2 Pemrograman Berorientasi Objek

Pemrograman berorientasi objek adalah suatu pendekatan pemrograman yang menggunakan *object* dan *class*. Pemrograman berorientasi objek ini memberikan kemudahan dalam pembuatan sebuah program.

Tidak seperti desain terstruktur yang berfokus pada dekomposisi fungsional dari sistem, desain orientasi objek berfokus pada dekomposisi objek. Secara formal, IEEE mendefinisikan desain berorientasi objek sebagai sebuah strategi desain di mana sistem atau komponen dinyatakan dalam bentuk objek dan hubungan antara objek-objeknya (Otero, 2012:32).

Menurut Simarmata, pengembangan berorientasi objek adalah berbeda dengan pemrograman tradisional yang memisahkan data dan kontrol. Pemrograman berorientasi objek dilakukan berdasarkan objek, dengan sekumpulan data yang ditetapkan dan sekumpulan operasi-operasi yang dapat dilakukan dalam data tersebut (Simarmata, 2010:74).

2.2 *Unified Modeling Language* (UML)

Menurut Carlos E. Otero, dalam buku yang berjudul "*Software Engineering Design Theory and Practice*", definisi UML adalah bahasa visual dengan serangkaian fitur yang luas yang sesuai untuk merancang suatu sistem perangkat lunak pada sebuah aplikasi.

"*The Unified Modelling Language (UML) is a standard language for writing software blueprints. UML may be used to visualize, specify, construct, and document the artifact of a software-intensive system."*(Pressman, 2015:869).

UML sama seperti *blueprint* yang digunakan untuk membuat bangunan, UML digunakan sebagai alat bantu untuk membangun sebuah *software.*

Menurut Stephens, UML membantu untuk menentukan bagaimana seharusnya bagian-bagian dari sistem bekerja. UML bukanlah sebuah kesatuan bahasa, tetapi UML menggunakan bermacam-macam diagram untuk mempresentasikan berbagai bagian yang berbeda dari suatu sistem (Stephens, 2015:77).

Berdasarkan teori-teori di atas maka dapat disimpulkan bahwa UML adalah sebuah bahasa visual dan bukan sebuah kesatuan bahasa dengan fitur yang luas untuk

merancang sistem perangkat lunak dan menentukan bagaimana seharusnya bagian-bagian dari sistem bekerja.

Diagram UML dibagi ke dalam beberapa diagram lebih tepatnya dibagi menjadi tiga kategori diagram dan terdapat macam-macam diagram di setiap kategori tersebut. Pertama, *structure diagrams*, yaitu *class, object, component, composite structure, package,* dan *deployments diagrams*. Kedua, be*havior diagrams*, yaitu *use case, activity,* dan *state machine diagrams*. Ketiga, *interaction diagrams*, yaitu *sequence, communication, timing,* dan *interaction overview diagrams* (Roger Y. Lee, 2013:41).

1.2.1. Use Case Diagram

Menurut Foster dalam buku yang berjudul "*Software Engineering: A Methodical Approach*" *use case* adalah representasi dari semua kemungkinan interaski dalam sebuah sistem oleh satu atau lebih faktor dalam menanggapi stimulus awal oleh aktor tersebut (Foster, 2014:417).

Menurut Carlos E. Otero, dalam buku yang berjudul "*Software Engineering Design: Theory and Practice*", definisi diagram *use case* adalah perilaku diagaram digunakan untuk menangkap, menentukan, dan memvisualisasikan perilaku sistem yang diperlukan. Unsur-unsur utama dari diagram *use case* adalah aktor, *use cases*, dan hubungan yang menghubungkan semuanya (Otero, 2012:55).

Menurut Stephen R. Schach dalam buku yang berjudul "*Object-Oriented Software Engineering*" definisi diagram *use case* adalah model interaksi antara pengguna eksternal dari produk perangkat lunak (aktor) dan produk perangkat lunak itu sendiri. Lebih tepatnya, aktor adalah pengguna yang memainkan peran tertentu. Diagram *use case* adalah kumpulan dari beberapa *use case* (Schach, 2010:488).

Kesimpulan dari beberapa definisi yang menjelaskan tentang *use case* bahwa *use case diagram* adalah sebuah diagram yang menangkap, menentukan, dan memvisualisasikan perilaku sistem juga interaksi antara aktor-aktor.

2.2.1 Class Diagram

Menurut Sommerville (Sommerville, 2011:129), *class diagram* digunakan ketika mengembangkan model sistem berorientasi objek untuk menunjukan kelas dalam sistem dan hubungan antara kelas-kelas. *Object class* dapat dianggap sebagi definisi umum dari satu jenis objek sistem. Sebuah asosiasi merupakan hubungan antara kelas yang menunjukkan bahwa ada hubungan antara kelas-kelas. Akibatnya, setiap kelas mungkin harus memiliki pengetahuan kelas yang terkait.

Class diagram menunjukkan bagaimana entitas yang berbeda (orang, benda, dan data) berhubungan satu sama lain, dengan kata lain menunjukkan struktur statis dari sistem. Sebuah diagram kelas dapat digunakan untuk menampilkan kelas logis yang biasanya hal-hal yang orang-orang bisnis dalam pembicaraan tentang organisasi *band rock*, CD, radio, hipotek rumah, kredit mobil, dan tingkat suku bunga. *Class diagram* juga dapat digunakan untuk menunjukan kelas implementasi yang merupakan hal yang *programmer* biasanya tangani. Implementasi *class diagram* mungkin menunjukkan beberapa kelas yang sama dengan kelas logis diagram. Implementasi *class diagram* tidak ditarik dengan atribut yang sama karena kemungkinan besar memiliki referensi untuk hal-hal seperti *vector* dan *hashmaps*. (*class diagram*, www.ibm.com)

Berdasarkan seluruh teori tentang *class diagram,* dapat disimpulkan bahwa *class diagram* adalah diagram yang menggambarkan hubungan entitas, kelas-kelas, dan operasi.

2.2.2 Activity Diagram

Menurut Elvis C. Foster pada buku yang berjudul "*Software Engineering: A Methodical Approach*" menjelaskan *activity diagram* didukung UML sebagai alternatif untuk *event diagram*. *Activity diagram* menggambarkan bagaimana terkait kegiatan (perhitungan dan alur kerja). Kegiatan tidak eksistensi independen, tetapi biasanya tunduk pada operasi dan kegiatan lainnya. Diagram aktivitas (grafik) menggambarkan keadaan aktivitas dan transisi di antar kegiatan di alur kerja. *Activity state* merupakan eksekusi pernyataan dalam prosedur, atau kinerja dari suatu kegiatan dalam alur kerja.

Menurut Rod Stephens, dalam buku yang berjudul "*Beginning Software Engineering*", pengertian *activity diagram* adalah presentasi dari sebuah kegiatan. *Activity diagram* menggunakan berbagai simbol yang terhubung dengan panah untuk menggambarkan arah arus kegiatan.

Berdasarkan teori-teori tentang *activity diagram,* dapat disimpulkan bahwa *activity diagram* adalah diagram yang mempresentasikan keadaan dan arus kegiatan dengan menggunakan berbagai simbol yang terhubung satu dengan yang lainnya.

2.3 Face Recognition

Wajah merupakan bagian tubuh manusia yang menjadi fokus perhatian dalam sebuah interaksi sosial. Wajah memainkan peranan cukup vital dalam suatu interaksi sosial karena wajah dapat menunjukan emosi manusia. Kemampuan manusia yang dapat mengenali seorang dari wajah merupakan sesuatu yang luar biasa. Manusia dapat mengenali ribuan wajah tergantung frekuensi interaksi orang tersebut dengan orang lain, baik dalam keadaan hanya sekilas bertemu, sering bertemu, atau dalam rentang waktu yang sangat lama. Manusia juga mampu mengenali seseorang walaupun terjadi perubahan pada orang tersebut karena bertambahnya usia atau pemakaian aksesori pada orang tersebut, misalnya berupa kacamata atau perbedaan gaya rambut. Oleh karena itu wajah digunakan sebagai bagian dari tubuh manusia yang dijadikan indikasi pengenalan terhadap seseorang.

Face recognition adalah suatu metode pengenalan yang berorientasi pada wajah seseorang. Setelah dilakukannya perbandingan dengan pola yang sebelumnya sudah tersimpan di dalam *database*, *face recognition* atau pengenalan wajah hanya dapat dibagi menjadi dua bagian, yaitu wajah dapat dikenali atau wajah tidak dapat dikenali. Metode pengenalan wajah juga harus mampu mengenali objek sekitar yang berbeda dari pembelajaran wajah yang sudah ada di dalam *database*, kesulitan muncul ketika wajah direpresentasikan dalam suatu pola yang berisi informasi unik yang membedakan satu wajah dengan wajah lainnya (*face recognition,* www.academia.edu).

Face recognition merupakan salah satu pendekatan pengenalan pola untuk keperluan identifikasi personal disamping pendekatan *biometric* lainnya seperti pengenalan sidik jari, tanda tangan, pengenalan citra wajah berhubungan dengan objek yang tidak pernah sama, karena adanya bagian-bagian yang dapat berubah. Perubahan ini dapat disebabkan oleh ekspreksi wajah, intensitas cahaya dan sudut pengambilan gambar, atau perubahan aksesori pada wajah. Dalam kaitan tersebut, objek yang sama dengan beberapa perbedaan tersebut harus dikenali sebagai satu objek yang sama.

Face recognition merupakan salah satu cara atau metode yang diguanakan manusia untuk dapat berinteraksi dengan sistem komputer, contohnya sebagai pengganti *password* untuk *login* dalam suatu aplikasi atau membuka kata sandi *handphone* atau sejenisnya. Proses pengenalan wajah adalah mengubah wajah manusia ke dalam citra digital yang kemudian diekstraksi menjadi suatu kumpulan informasi yang istimewa sehingga dapat diimplementasikan ke dalam pola-pola pengklasifikasian dan dilakukannya proses pengenalan wajah.

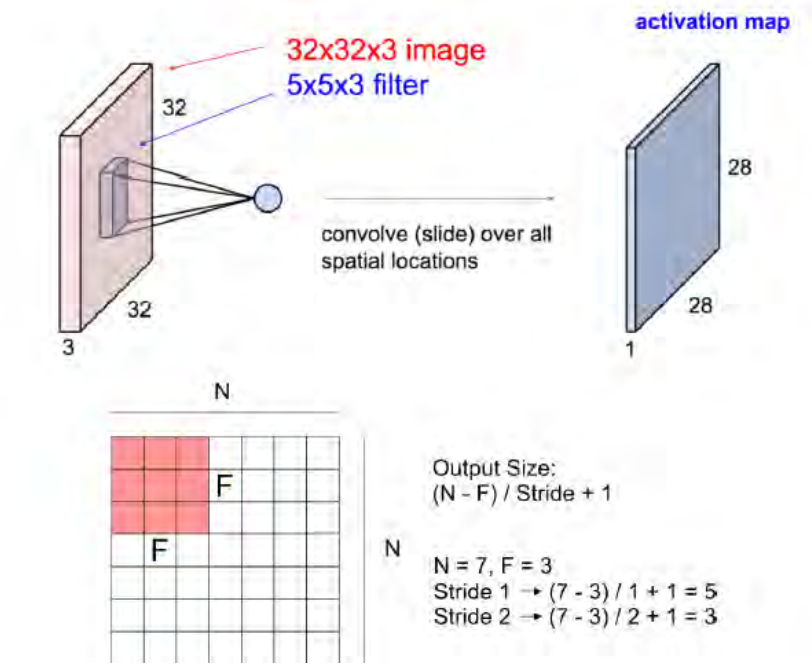
Face recognition adalah semua tentang penggalian fitur-fitur yang berarti dari sebuah gambar, menempatkan mereka dalam representasi berguna dan melakukan beberapa jenis klasifikasi pada mereka. *Face recognition* berdasarkan pada fitur geometris wajah adalah pendekatan yang paling intuitif untuk *face recognition*.

2.4 CNN

Convolutional Neural Network (CNN) adalah merupakan salah satu jenis *neural network* yang berisi kombinasi beberapa layer yaitu *convolutional layer*, *pooling layer*, dan *fully connected layer* (Hu, et al. 2015). *Convolutioinal Layer* memproses data dengan *topologi grid* (Goodfellow, et al. 2016). *Convolutional Neural Network* menggunakan operasi *convolution* pada perkalian matriks di setiap layer. Arsitektur *Convolutional Neural Network* (CNN) dapat dilihat pada gambar dibawah dimana jaringan ini terdiri dari beberapa layer, yakni *convolutional layer*, *pooling layer,* dan *fully connected layer*:

1. *Convolutional Layer*

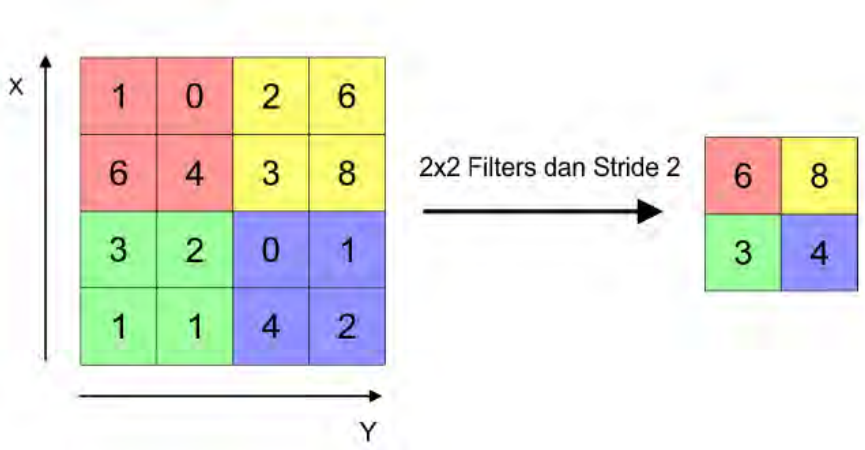
Convolutional Layer adalah yang pertama kali menerima input gambar langsung. *Convolutional layer* memiliki *hyperparameter* dan *parameter* dan terdiri dari neuron yang tersusun sehingga membentuk sebuah filter dengan panjang dan tinggi.



Gambar 2.1
Convolutional Layer
(Sumber:https://leonardoaraujosantos.gitbooks.io/artificial inteligence/content/convolutional_neural_networks.html)

2. *Pooling Layer*

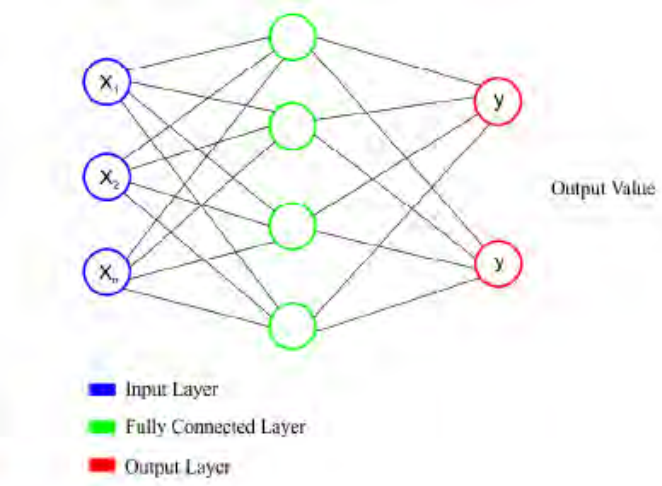
Pooling layer biasanya terletak berada setelah *convolutional layer*. *Pooling layer* juga mereduksi ukuran spasial dan jumlah parameter dalam suatu jaringan serta mempercepat komputasi dan mengontrol apabila terjadinya *overfitting*. *Pooling layer* memiliki beberapa macam tipe antara lain *average pooling*, *max pooling*, dan *Lp Pooling*. Tujuan dari penggunaan *pooling layer* adalah mengurangi dimensi dari *feature map*.



Gambar 2.2
Pooling Layer

3. *Fully Connected Layer*

Sebelum memasuki tahap atau bagian *Fully Connected Layer* ini terlebih dahulu memasuki proses "*flatten*" atau *reshape*. Proses *flatten* ini menghasilkan sebuah vektor untuk digunakan sebagai sebuah input dari *Fully Connected Layer*. *Fully Connected Layer* memiliki *Hidden Layer*, *Action Function*, *Output Layer*, dan *Loss Function*.



Gambar 2.3
Fully Connected Layer

2.5 Tensorflow

Tensorflow adalah *library* yang bersifat *open source end-to-end* yang dibuat dan dikembangkan oleh *Google Brain* untuk mendukung mempercepat pembelajaran *machine*

learning (Rungta, 2018, p.29). *Tensorflow* memiliki pustaka, alat, dan sumber daya yang komprehensif dan fleksibel yang memungkinkan *developer* dengan mudah membangun atau menerapkan aplikasi yang didukung *machine learning*.

Tensorflow juga dibangun untuk berjalan pada banyak CPU dan GPU bahkan sistem operasi selular. *Tensorflow* juga sudah dapat digunakan pada bahasa pemograman seperti *python* dan *C++*.

2.6 OpenCV Library

OpenCV singkatan dari *Open source Computer Vision*. *OpenCV* adalah sebuah *library* fungsi pemrograman yang ditujukan untuk *computer vision* (Gollapudi, 2019:32). Awalnya dikembangkan oleh pusat penelitian Intel di Nizhny Novgorod (Rusia), kemudian didukung oleh Willow Garage dan sekarang dikelola oleh Itseez (*OpenCV,* www.opencv.org./about/). *Library OpenCV* di bawah lisensi BSD *open-source* gratis dan *cross-platform* untuk digunakan. Di dalamnya terdapat ratusan algoritma *computer vision*. *OpenCV* dapat mendukung bahasa pemrograman *Java*, *Python*, *C/C++* yang digunakan untuk membangun aplikasi *desktop* dan selular seperti *Linux, Windows, macOS, iOS* dan *Android*. *OpenCV* diimplementasikan dengan beberapa contoh seperti klasifikasi gerak manusia dalam video, deteksi suatu objek, deteksi wajah, dan pengenalan wajah. Karena *OpenCV* bersifat *open source* sehingga penggunaan yang cukup besar dan dapat digunakan oleh siapapun, perusahaan, lembaga, swasta dan pemerintah. Penggunan *OpenCV* membantu implementasi pengenalan wajah dengan metode yang digunakan.

2.7 Python

Python merupakan salah satu bahasa pemrograman yang dikeluarkan pada tahun 1991 oleh Guido van Rossum di Belanda. Python dikembangan dengan sifat *open source* dengan versi linsensi *GFL-compatible*. Perkembangan Python awal pada tahun 1990, Python dikembangkan di *Scitchting Mathematisch Centrum* Belanda. Lalu pada tahun 1995 diluncurkan Python versi ke 2 di *Corporation for National Research Initiative* Virginia, Amerika Serikat. Lalu pada tahun 2000 Guido dengan tim membentuk tim *BeOpenPythonLabs* untuk focus mengembangkan *Python*. Lalu pada tahun 2001 *Python*

Software Foundation dibuat untuk menangani hak intelektual *Python* dan sampai saat ini masih melakukan riset untuk mengembangkan *Python*.

Python merupakan bahasa pemrograman yang cukup pesat karena memiliki kelebihan pada aspek efisiensi, *readibility*, dan multifungsi interoprabilitas. Aspek efisiensi dari *Python* memiliki *library* yang lengkap, aspek *readibility* dari *Python* mudahnya penggunaan sintak yang sederhana dan tidak memakan ruang untuk *code* dibandingkan dengan bahasa pemograman *Visual Basic, C, Java* yang dapat menghemat waktu serta meningkatkan produktivitas. Aspek multifungsi dari Python dapat membuat aplikasi jaringan, *website*, robotika, *deep learning*, *machine learning*, dan *artificial intelligent* (Lutz, 2013:1).

BAB III
ANALISIS DAN PERANCANGAN PERANGKAT LUNAK

3.1 Gambaran Umum Perangkat Lunak

Perangkat lunak yang dirancang adalah sebuah sistem kehadiran dengan menggunakan metode pengenalan wajah atau *face recognition*. Sistem kehadiran ini dibuat untuk meningkatkan kinerja daripada sistem kehadiran yang sudah ada sebelumnya dan membantu untuk menghindari kontak fisik saat di masa pandemi bahkan sistem ini memudahkan dalam proses perekaman kehadiran. Para *user* yang merekam kehadiran diambil terlebih dahulu sampel wajahnya. Setelah selesai diambil sampel wajah, selanjutnya dilakukan *training* wajah supaya sistem bisa mengenali wajah-wajah para *user* yang melakukan perekaman kehadiran nantinya. Sistem perekaman kehadiran ini menggunakan kamera *webcam* dengan pengambilan wajah dengan cara wajah harus harus menghadap kearah kamera *webcam*. Pada penelitian kali ini wajah diambil dari *website www.pinterest.com* yang berupa gambar wajah orang tapi bisa juga diambil dari contoh wajah orang asli *realtime* setelah itu gambar wajah dimasukkan ke *handphone* lalu layar *handphone* diarahkan kearah *webcam*. Sistem melakukan pengecekan apakah gambar atau citra tersebut bisa kenali bahwa adalah orang setelah itu citra dan gambar itu akan diproses dan di *training* dan akan menjadi model untuk proses sistem kehadiran. Setelah berhasil melakukan perekaman wajah *user* melakukan perekaman kehadiran dan sistem memberikan *output* bahwa *user* sudah berhasil melakukan perekaman kehadiran. Sistem perekaman kehadiran ini dibuat menggunakan *python* sebagai bahasa pemogramannya dengan bantuan *library OpenCV*, dengan metode CNN untuk mengenali wajah.

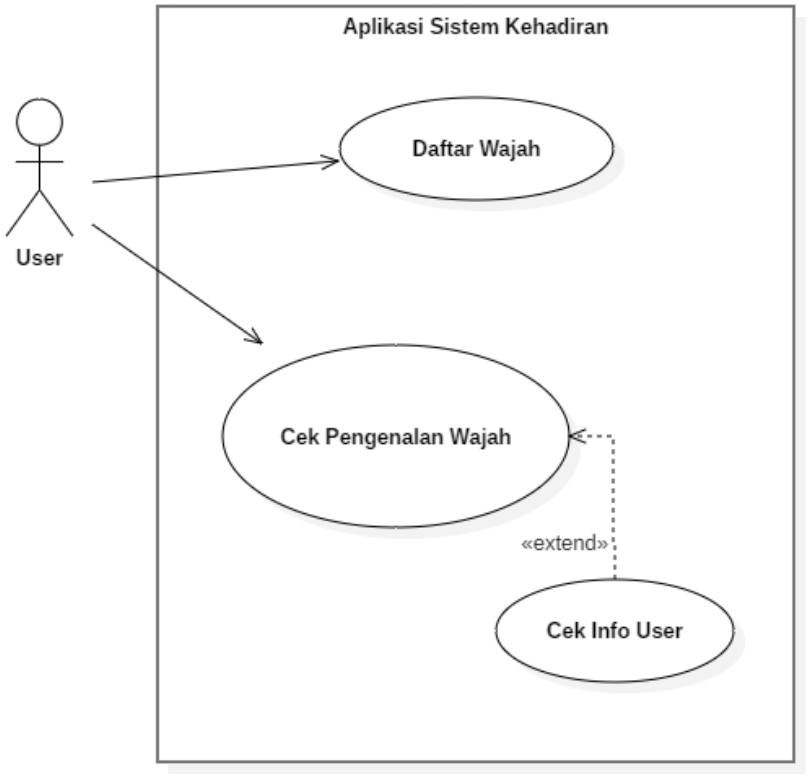
3.2 Spesifikasi Kebutuhan Perangkat Lunak

Perangkat lunak diharapkan untuk memenuhi kebutuhan antara lain sebagai berikut:

1. Sistem dapat terhubung dengan kamera *webcam* dengan baik.
2. Sistem dapat mengenali wajah pada saat melakukan perekaman kehadiran.
3. Sistem diharapkan dapat mengenali wajah dengan tepat.

3.3 Use Case

Use Case ini dirancang antara interaksi pengguna dan sistem. Perangkat lunak yang dirancang untuk para pengguna yang sudah memiliki akses dalam sistem perekaman



Gambar 3.1
Use Case

kehadiran. Dalam *use case* ini sudah dirancang bahwa sistem melakukan *training* wajah, deteksi wajah dan, identifikasi wajah sebagai berikut:

3.4 Spesifikasi Proses

Spesifikasi proses menggambarkan deskripsi dari setiap *use case* yang ada pada *use case diagaram*. Spesifikasi proses menjelaskan bagaimana proses *use case*, peran yang aktor lakukan dan respon dari sistem.

3.4.1 Skenario *Use Case* Daftar Wajah

Use Case: Daftar Wajah

Aktor: *User*

Deskripsi: *Use Case* menggambarkan sedang melakukan *traning* wajah untuk dijadikan *dataset*

Kondisi Awal: *Input* data wajah untuk di-*training*

Kondisi Akhir: Data wajah sudah masuk ke dalam *dataset* dan sudah menjadi model.

Tabel 3.1
Skenario Daftar Wajah

Aksi Aktor	Reaksi Sistem
1. Membuka aplikasi rekam kehadiran	
	2. Menampilkan menu awal dari sistem kehadiran
3. *User* memilih menu ‘daftar wajah’	
	4. Menampilkan layar dengan tampilan *webcam*, dan *text field* (nama, *email*, no hp) yang harus diisi
5. *User* mengisi data diri dan menghadapkan wajah sejajar ke arah kamera sambil menekan tombol ‘daftar’	
	6. Sistem akan memproses data diri dan wajah
	7. Melakukan *preprocessing* wajah
	8. Melakukan *train* wajah
	9. Data wajah disimpan menjadi *dataset* dan model
	10. Proses selesai menampilkan pesan berhasil daftar

Skenario Alternatif: Data wajah tidak dapat masuk ke dalam *database*

Tabel 3.2
Skenario Alternatif Daftar Wajah

Aksi Aktor	Reaksi Sistem
1. Membuka aplikasi rekam kehadiran	
	2. Menampilkan menu awal dari sistem kehadiran
3. *User* memilih menu ‘daftar wajah’	
	4. Menampilkan layar dengan tampilan *webcam*, dan *text field* (nama, *email*, no hp) yang harus diisi
5. *User* mengisi data diri tetapi tidak menghadapkan wajah sejajar sehingga keluar dari *frame* kamera	

Aksi Aktor	Reaksi Sistem
	6. Sistem tidak akan memproses apapun

3.4.2 Skenario *Use* Rekam Kehadiran

Use Case: Rekam Kehadiran Wajah

Aktor: *User*

Deskripsi: *Use Case* menggambarkan *user* sedang melakukan rekam kehadiran

Kondisi Awal: Menampilkan menu untuk rekam kehadiran

Kondisi Akhir: Menampilkan bahwa sudah berhasil melakukan rekam kehadiran

Tabel 3.3
Skenario Rekam Kehadiran

Aksi Aktor	Reaksi Sistem
1. Membuka aplikasi rekam kehadiran	
	2. Menampilkan menu awal dari sistem kehadiran
3. *User* memilih menu ‘rekam kehadiran’	
	4. Menampilkan layar dengan tampilan *webcam* dan *button* ‘rekam kehadiran’
5. *User* menghadapkan wajah sejajar dengan *webcam* dan tidak keluar *frame webcam* dan menekan *button* ‘rekam kehadiran’	
	6. Sistem melakukan validasi wajah
	7. Sistem akan melakukan pengenalan wajah dengan algoritma CNN
	8. Menampilkan pesan bahwa sudah berhasil melakukan rekam kehadiran dan menampilkan data diri *user*

Skenario alternatif 1: Wajah terdeteksi tapi salah pengenalan.

Tabel 3.4
Skenario Alternatif 1 Rekam Kehadiran.

Aksi Aktor	Reaksi Sistem
1. Membuka aplikasi rekam kehadiran	

Aksi Aktor	Reaksi Sistem
	2. Menampilkan menu awal dari sistem kehadiran
3. *User* memilih menu ‘rekam kehadiran’	
	4. Menampilkan layar dengan tampilan *webcam* dan *button* ‘rekam kehadiran’
5. *User* menghadapkan wajah sejajar dengan *webcam* dan tidak keluar *frame webcam* dan menekan *button* ‘rekam kehadiran’	
	6. Sistem melakukan validasi wajah tapi salah mengenali wajah
	7. Menampilkan pesan bahwa sudah berhasil melakukan rekam kehadiran dan menampilkan data diri *user* lain

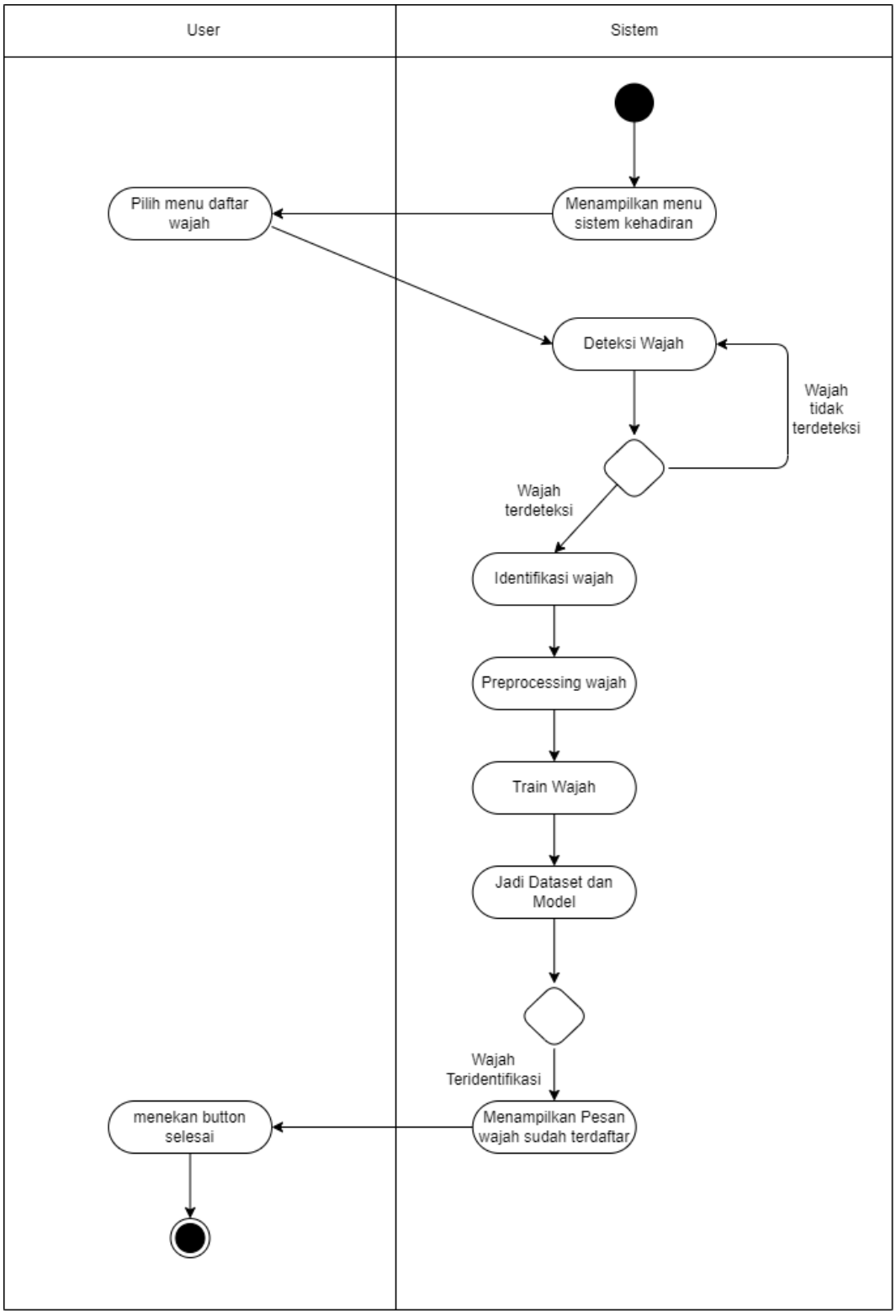
Skenario alternatif 2: Wajah tidak terdeteksi di luar *frame webcam*.

Tabel 3.5
Skenario Alternatif 2 Rekam Kehadiran

Aksi Aktor	Reaksi Sistem
1. Membuka aplikasi rekam kehadiran	
	2. Menampilkan menu awal dari sistem kehadiran
3. *User* memilih menu ‘rekam kehadiran’	
	4. Menampilkan layar dengan tampilan *webcam* dan *button* ‘rekam kehadiran’
5. *User* tidak menghadapkan wajah sejajar dengan *webcam* dan tidak keluar *frame webcam* dan menekan *button* ‘rekam kehadiran’	
	6. Sistem tidak akan melakukan reaksi apapun

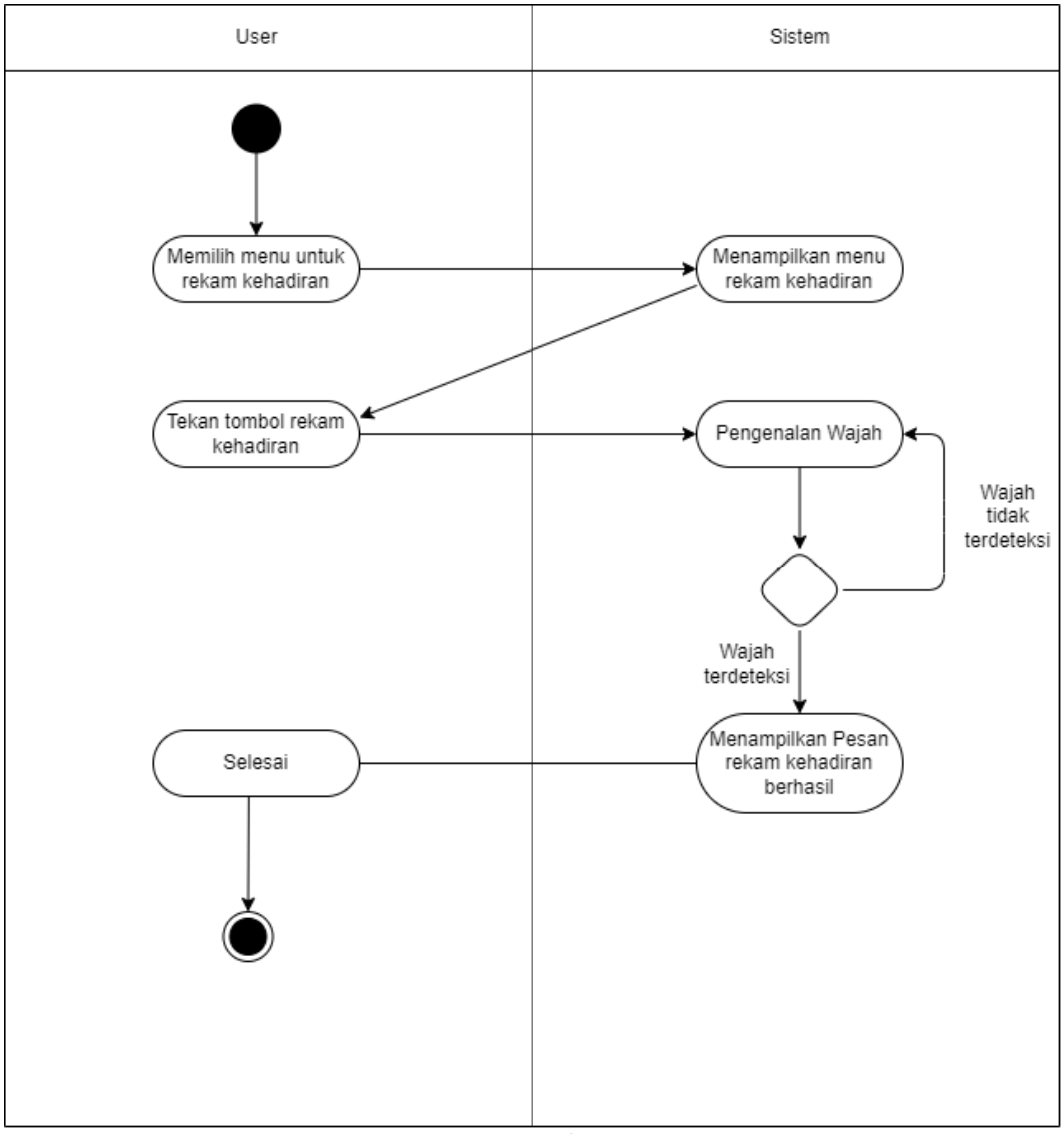
3.5 *Activity Diagram*

Pada gambar 3.2 *user* yang sudah berada di aplikasi sistem kehadiran bisa langsung memilih menu daftar wajah untuk melakukan daftar wajah dengan menggunakan *webcam*. Kemudian sistem melakukan deteksi apakah benar terdapat wajah pada *frame webcam*. Sistem akan melakukan *training* wajah dan dimasukkan menjadi *dataset* dan model wajah.



Gambar 3.2
Activity Diagram Daftar Wajah

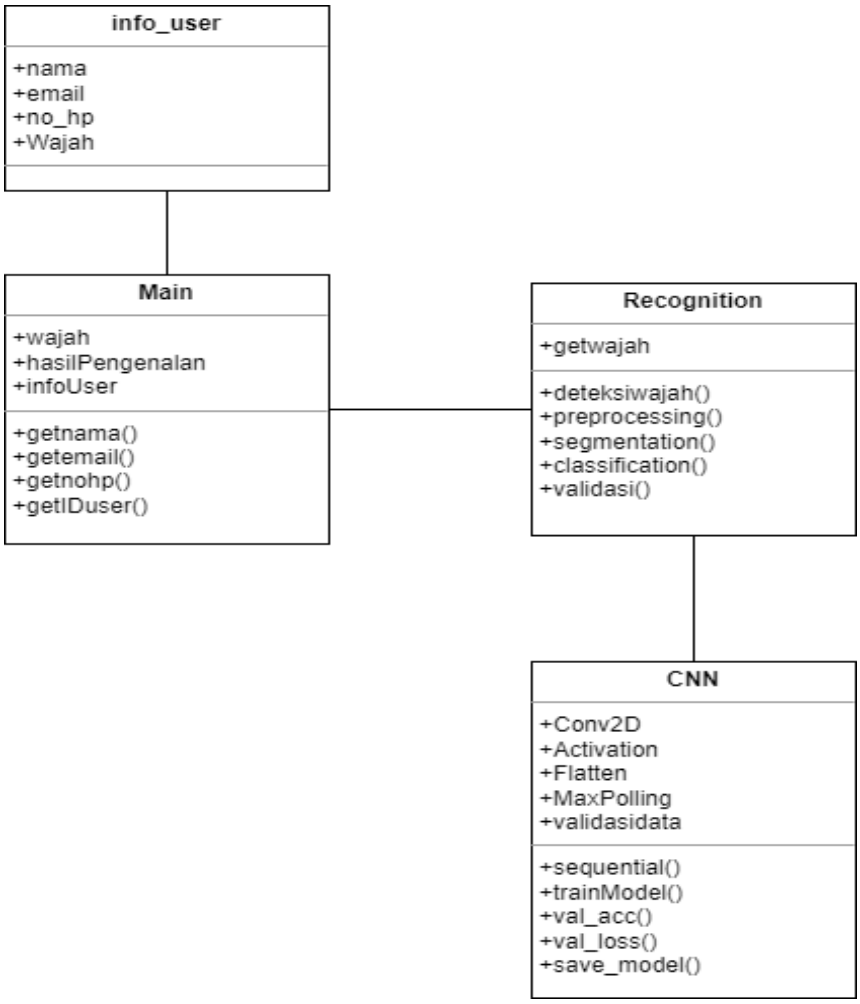
Pada gambar 3.3, *user* akan melakukan rekam kehadiran diamana *user* yang sudah mendaftar sudah memiliki *dataset* dan model wajahnya di dalam aplikasi sistem kehadiran. Sistem akan memproses wajah yang pada *frame webcam*, dan hasil dari itu akan menampilkan hasil bahwa sudah berhasil melakukan rekam kehadiran.



Gambar 3.3
Activity Diagram Rekam Kehadiran

3.6 *Class Diagram*

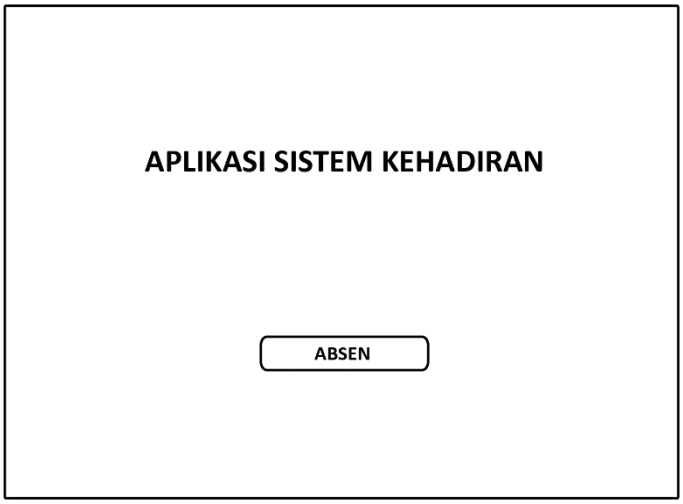
Gambar 3.4 adalah *class diagram* untuk perangcangan perangkat lunak yang ditujukan untuk pengenalan wajah dalam sistem kehadiran.



Gambar 3.4
Class Diagram

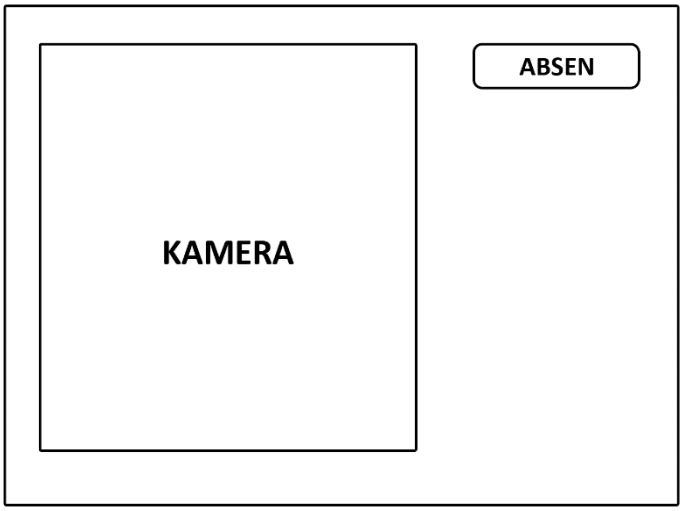
3.7 Rancangan Antar Muka

Rancangan antar muka pada perangkat lunak digambarkan pada gambar 3.5 merupakan tampilan utama.



Gambar 3.5
Tampilan Menu Awal

Pada menu utama dalam perancangan antar muka ini terdapat sebuah *icon* yang ditujukan ke dalam proses pengenalan wajah pada gambar 3.6.



Gambar 3.6
Tampilan Kamera

Setelah masuk menu perekaman kehadiran dan *user* sudah memilih *icon* absen muncul layar kamera untuk melakukan foto wajah yang dilakukan oleh *user* gambar 3.6. Pada tahap ini *user* menekan tombol foto seperti layaknya ketika kita sedang foto, setelah itu dideteksi oleh sistem apakah benar ada wajah atau tidak.

Gambar 3.7
Tampilan *User* Berhasil Absen

Pada rancangan antar muka ketika *user* sudah melakukan foto wajah sistem melakukan deteksi wajah dan ketika sudah berhasil melakukan absen aplikasi menampilkan seperti pada gambar 3.7.

BAB IV
IMPLEMENTASI DAN PENGUJIAN PERANGKAT LUNAK

4.1 Spesifikasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan untuk melakukan pengembangan aplikasi adalah sebagai berikut:

1. Laptop dengan *processor Intel Core I5-8250u 1.6 GHz - 3.4 GHz*
2. *RAM 20 GB*
3. *Intel UHD Graphics 620* dan *NVIDIA MX130 2GB*
4. *HDD 1TB* dan *SSD 256GB*
5. *Webcam* Laptop

4.2 Spesifikasi Perangkat Lunak

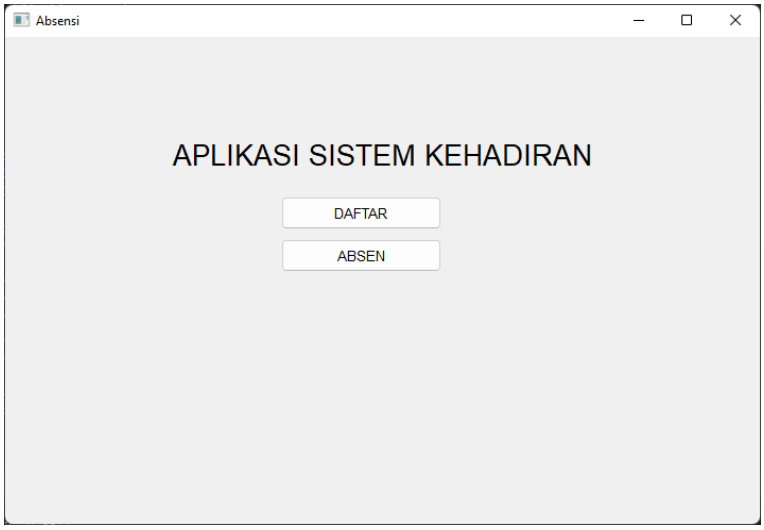
Spesifikasi perangkat lunak yang digunakan untuk melakukan pengembangan aplikasi adalah sebagai berikut:

1. Sistem Operasi *Windows 11 pro 64 bit*
2. *OpenCV*
3. *PyCharm*

4.3 Pengujian Antarmuka

Pengujian antarmuka pada perangkat lunak yang dibuat terdiri dari tampilan awal untuk absensi sistem kehadiran.

Pada gambar 4.1 terdapat antarmuka awal yang terdiri dari 2 *button*, ‘daftar’ dan ‘absen’.

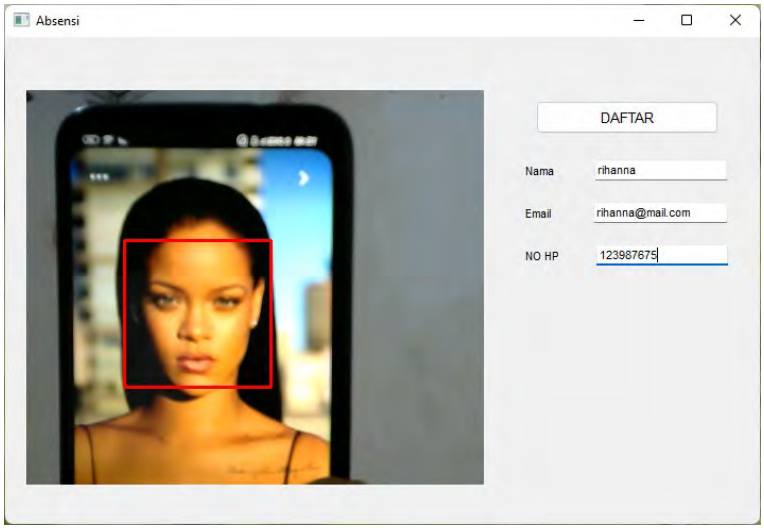


Gambar 4.1
Antarmuka Halaman Awal

Pada halaman awal pada gambar 4.1 terdapat beberapa komponen yaitu:

1. *Button* ‘Daftar’ untuk mendaftarkan wajah.
2. *Button* ‘Absen’ untuk absen.

Pada Gambar 4.2 merupakan antarmuka halaman daftar wajah



Gambar 4.2
Tampilan Daftar Wajah

Pada halaman daftar pada gambar 4.2 terdapat beberapa komponen yaitu:

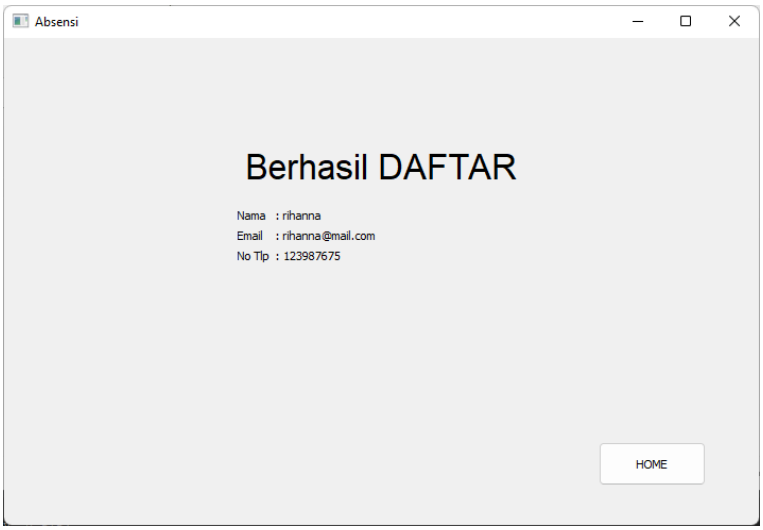
1. *Button* Daftar untuk mendaftarkan wajah beserta data-data yang di-*input*.
2. *Text input field* nama untuk mendaftarkan nama *user*.

3. *Text input field email* untuk mendaftarkan *email user*.

4. *Text input field* no. hp untuk mendaftarkan nomor *handphone user*.

5. *Webcam Realtime* untuk melihat pendeteksian wajah.

Pada gambar 4.3 merupakan antar muka setelah mendaftarkan wajah.



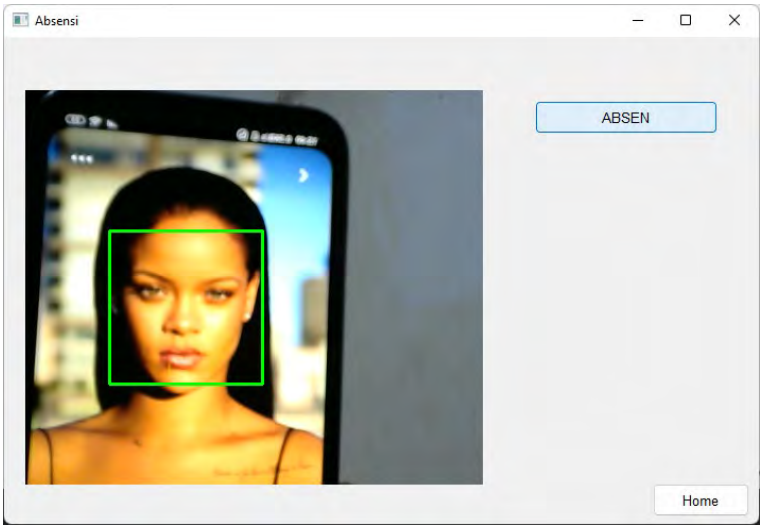
Gambar 4.3
Tampilan Halaman setelah Daftar Wajah

Pada halaman hasil dari daftar wajah pada gambar 4.3 terdapat beberapa komponen yaitu:

1. *Text Preview* menampilkan data-data yang berhasil didaftarkan.

2. *Button Home* untuk kembali ke halaman awal aplikasi.

Pada gambar 4.4 merupakan antarmuka halaman absen.

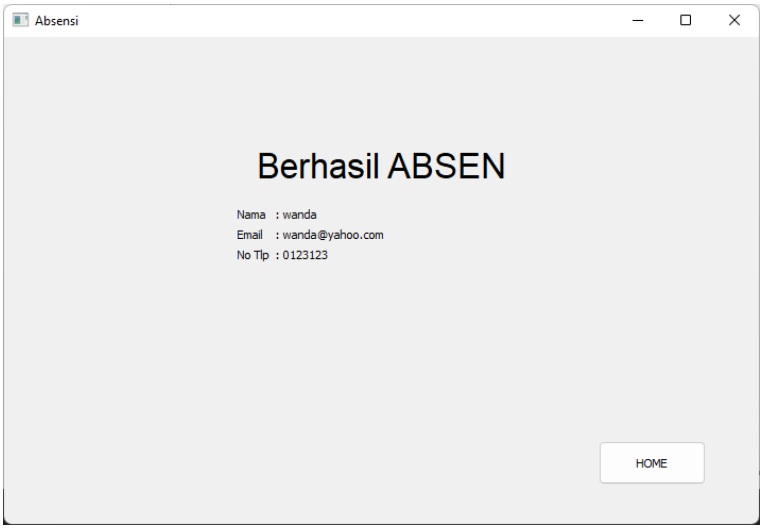


Gambar 4.4
Tampilan Halaman Absen

Pada halaman absen di gambar 4.4 terdapat beberapa komponen yaitu:

1. *Webcam Realtime* untuk menampilkan wajah yang diabsen.
2. *Button* 'Absen' untuk absen.
3. *Button 'Home'* untuk kembali ke halaman awal.

Gambar 4.5 menunjukkan halaman hasil setelah *user* absen.



Gambar 4.5
Tampilan Hasil setelah Absen

Pada halaman hasil absen pada gambar 4.5 terdapat beberapa komponen yaitu:

1. *Text field preview* untuk menampilkan data *user* yang melakukan perekaman kehadiran.
2. *Button 'Home'* untuk kembali ke halaman awal.

4.4 Pengujian Fungsi

Untuk menganalisa program yang diuji menggunakan pengujian *black box* untuk mendapatkan hasil dan memenuhi kebutuhan perangkat lunak.

Tabel 4.1
Pengujian *Black Box*

No	Data yang diuji	Alternatif Pengujian	Hasil yang diharapkan	Hasil Pengujian	Status
1	*Button* 'Daftar'	-	Menampilkan halaman Daftar untuk absen	Menampilkan halaman Daftar untuk absen	Valid
2	*Button* 'Absen'	-	Menampilkan halaman Absen untuk absen	Menampilkan halaman Absen untuk absen	Valid
3	*Button* 'Daftar' (pada halaman daftar)	Berhasil mendaftarkan wajah dan data *user*	Menampilkan bahwa wajah dan data sudah tersimpan	Menampilkan bahwa wajah dan data sudah tersimpan	Valid
		Tidak berhasil mendaftarkan wajah dan data *user*	Keluar Aplikasi	Keluar Aplikasi	Valid
4	*Text field input* data *user*	-	Menampilkan teks yang di tulis oleh *user*	Menampilkan teks yang di tulis oleh *user*	Valid
5	*Button* 'Absen' (pada halaman absen)	Berhasil Absen	Menampilkan bahwa *user* berhasil melakukan absen	Menampilkan bahwa *user* berhasil melakukan absen	Valid
		Tidak Berhasil Absen	Keluar Aplikasi	Keluar Aplikasi	Valid
6	*Button Home*	-	Menampilkan kembali ke halaman awal aplikasi	Menampilkan kembali ke halaman awal aplikasi	Valid

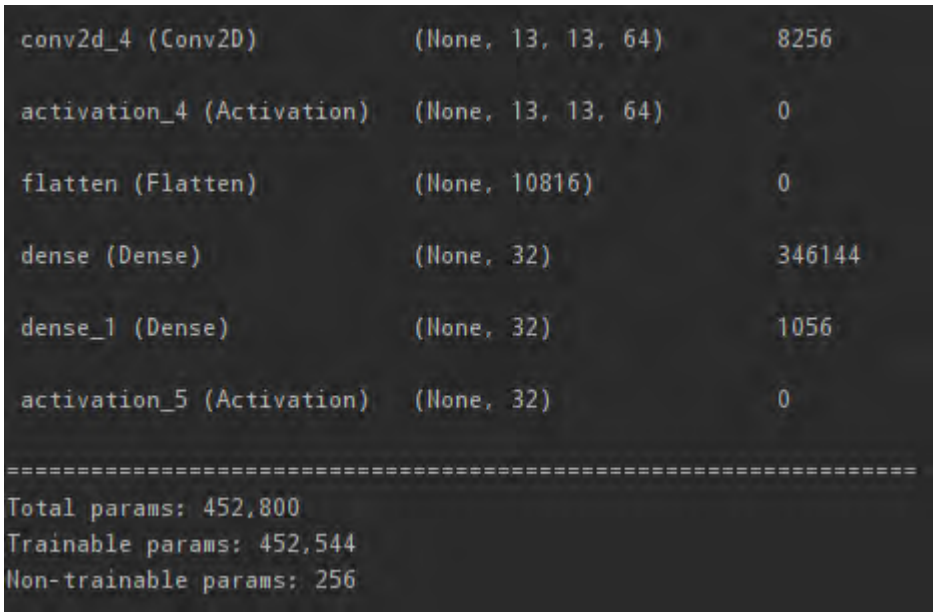
4.5 Pengujian Sistem

Pengujian yang dibuat terdiri dari proses *training*, *testing*, dan pengenalan terutama pengujian aplikasi dapat berjalan sesuai yang tujuan awal pembuatan. Proses *data train*

juga di-*training*-kan dengan parameter iterasi sebesar 5 *epoch* dengan setiap *step*-nya 32 *batch size*.

Tahapan proses dimulai dari proses pendaftaran wajah yang akan disimpan sebagai *user* yang akan melakukan perekaman kehadiran atau melakukan absensi kehadiran menggunakan *webcam* dan wajah perekaman wajah ini disertai perekaman data *user* tersebut sebagai contoh data yang dimasukkan berupa nama, *email*, dan no hp. Dalam proses pendaftaran ini citra wajah sudah di berikan identifikasi berupa border kotak yang menandakan bahwa terdapat citra yang dianggap wajah.

Pada citra wajah dilakukan augmentasi data juga melakukan *data train* yang di-*training*-kan dengan parameter iterasi sebesar 5 *epoch* dengan setiap *step* 32 *batch size* dan menggunakan lapisan *Conv2D, MaxPooling2d, Flatten*, dan *Dense*. Model *training* ini juga merupakan model *training sequential*. Jumlah *dataset* yang digunakan dalam pengujian kali ini sebanyak 30 orang dan dibuat sendiri.



Gambar 4.6
Model Latih CNN

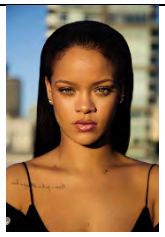
Pada table 4.2 terdapat hasil dari perhitungan *Confusion Matrix* dan *overall* akurasi mendapatkan nilai sebesar 93.904%. Berikut hasil dari masing-masing tabel:

Tabel 4.2
Hasil perhitungan *Confusion Matrix*

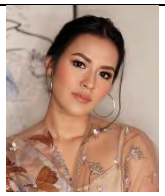
User ID	Precision	Recall	F1-score	Jumlah Data
0	1.00	1.00	1.00	14
1	1.00	1.00	1.00	7
2	1.00	1.00	1.00	25
3	1.00	0.95	0.98	21
4	1.00	1.00	1.00	26
5	1.00	1.00	1.00	25
6	1.00	0.78	0.88	27
7	1.00	1.00	1.00	30
8	1.00	0.96	0.98	26
9	1.00	0.96	0.98	26
10	0.00	0.00	0.00	25
11	1.00	1.00	1.00	24
12	1.00	1.00	1.00	23
13	1.00	1.00	1.00	41
14	1.00	1.00	1.00	26
15	0.28	1.00	0.44	18
16	0.96	0.96	0.96	25
17	1.00	1.00	1.00	28
18	1.00	1.00	1.00	27
19	1.00	1.00	1.00	25
20	1.00	1.00	1.00	25
21	1.00	0.94	0.97	16
22	1.00	1.00	1.00	24
23	1.00	1.00	1.00	20
24	1.00	1.00	1.00	21
25	1.00	0.70	0.82	23
26	1.00	1.00	1.00	16
27	1.00	1.00	1.00	17
28	1.00	1.00	1.00	25
29	1.00	1.00	1.00	16
30	1.00	0.84	0.91	25
31	1.00	1.00	1.00	26
32	1.00	1.00	1.00	28
Akurasi			0.94	771
Macro avg	0.95	0.94	0.94	771

Berikut ini merupakan hasil dari pengujian dari aplikasi:

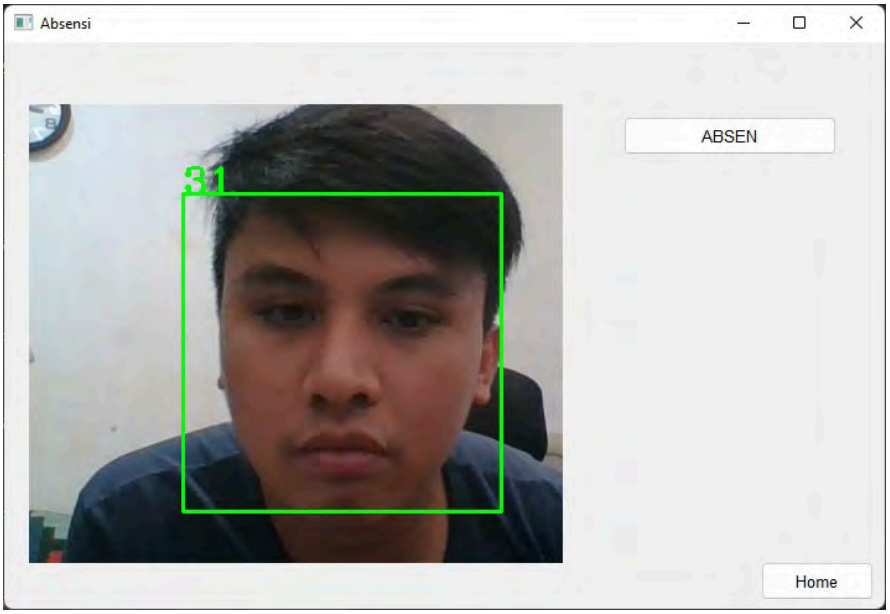
Tabel 4.3
Hasil Pengujian Sistem Kehadiran

No.	Wajah *User* Terdaftar	Nama *User* Terdaftar	Hasil Sistem Kehadiran
1		Scarlet	Benar
2		Wendy	Benar
3		Pevita	Benar
4		Hawk	Benar
5		Rihanna	Benar
6		Bruce	Benar

No.	Wajah *User* Terdaftar	Nama *User* Terdaftar	Hasil Sistem Kehadiran
7		Kim	Benar
8		Dawyne	Benar
9		Rose	Benar
10		Tesa	Benar
11		Sungkang	Benar
12		Antman	Benar
13		Michelle	Benar

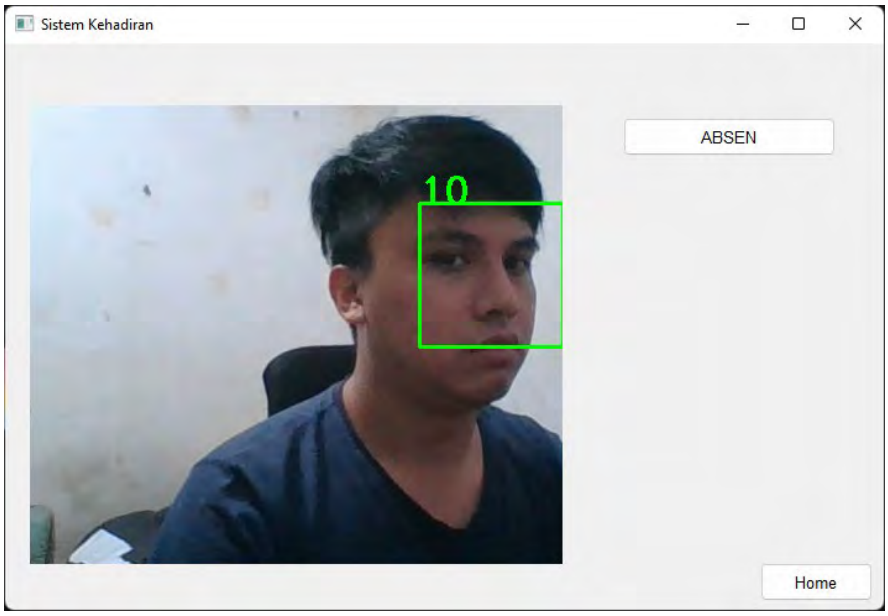
No.	Wajah *User* Terdaftar	Nama *User* Terdaftar	Hasil Sistem Kehadiran
14		Raisa	Benar
15		Mark	Benar

Pada gambar 4.7 adalah tampilan yang sedang melakukan perekaman kehadiran.

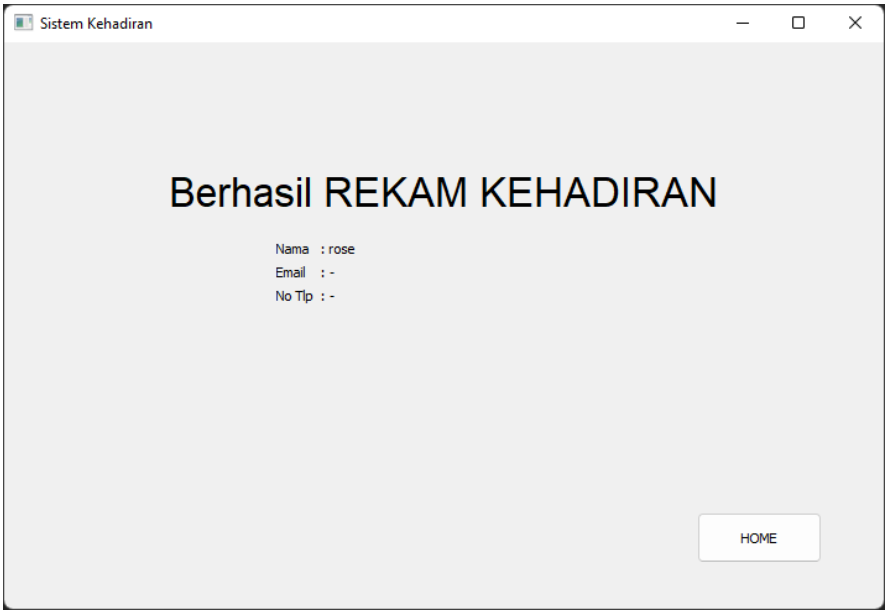


Gambar 4.7
Tampilan user sedang merekam kehadiran

Pada gambar 4.8 jika *user* tidak tepat pada posisi pengambilan *webcam* yang tepat makan hasil dari absen akan salah ditunjukan pada gambar 4.9.



Gambar 4.8
Tampilan user pada posisi Kamera yang tidak tepat



Gambar 4.9
Hasil Rekaman Kehadiran yang salah

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil dari penelitian, berikut kesimpulan yang dapat di berikan oleh penulis:

1. Hasil yang diperoleh pengenalan wajah menggunakan *Convolutional Neural Network* mendapatkan akurasi dengan nilai 93.904% dari hasil pengujian sistem kehadiran yang menggunakan 5 *epoch*. Nilai yang kurang karena penilitian kali ini menggunakan data wajah yang diambil dari gambar wajah yang terdapat pada layar *handphone*, tapi dengan hasil yang mencapai 93% perangkat lunak sudah bisa beroperasi dengan baik.
2. Penyebab terjadi kesalahan hasil perekaman kehadiran adalah jika wajah *user* berada tidak sejajar dengan *webcam*, ketika *user* sedang menghadap ke kiri atau ke kanan akan menyebabkan kesalahan perekaman kehadiran. Termasuk jumlah *data train* yang minimal akan membuat kesalahan pengenalan wajah *user*.

5.2 Saran

Untuk mengembangkan dari penelitian tugas akhir ini terdapat saran-saran berikut:

1. Penambahan *hardware* yang lebih menghasilkan kualitas gambar lebih baik akan membantu mendekatkan pada hasil yang lebih sempurna, sebagai contoh *external webcam* dari pada *webcam* laptop.
2. Menambahkan *dataset* wajah dari sisi lain selain wajah sejajar dengan *webcam*.
3. Menambahkan identifikasi lain ketika mau melakukan perekaman wajah tidak perlu *button* ‘rekam kehadiran’, tetapi menggunakan *motion user* menghadap kiri, menghadap kanan, menghadap depan sebagai verfikasi.

DAFTAR PUSTAKA

Academia, "*Face Recognition.",* https://www.academia.edu/ diakes tanggal 21 November 2021

Wibowo, A, et.al, 2019. *"Pendeteksian dan Pengenalan Wajah Pada Foto Secara Real Time Dengan Haar Cascade dan Local Binary Pattern Histogram"*, *Jurnal Teknik Elektro,* Vol. 9 No.1, ISSN:2503-2941.

Avestro, J., 2007. "*Pengenalan Pemograman 1",* Jeni Jardiknas.

Foster, E. C., 2014. "*Software Engineering: A Methodical Approach",* Springer Science Business Media.

Gollapudi, S., 2019. "*Learn Computer Vision Using OpenCV: With Deep Learning CNNs and RNNs",* Apress.

Goodfellow, Y., 2016. "*Deep Learning".* Massachusetts Institute of Technology.

IBM, http://www.ibm.com/ diakses tanggal 28 November 2020.

Lee, R. Y., 2013. "*Software Engineering: A Hands-On Approach",* Vol. 53, Atlantis Press.

Lutz, M., 2013. "*Python Pocket Reference, Fifth Edition"*, Vol. 53, O'Reilly Media, Inc.

Muhammad Zufar, B. S., 2016. "*Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time*", *Jurnal Sains Dan Seni ITS,* Vol. 5 No. 2(2337-3520).

OpenCV, "*About*", https://opencv.org./about/, diakses tanggal 7 Desember 2021.

Otero, C. E., 2012. "*Software Engineering Desin: Theory and PracticeI",* CRC Press, Taylor & Francis Group.

Pressman, R. S., 2010. "*Rekayasa Perangkat Lunak: Pendekatan Praktisi*", Andi.

Rungta, K., 2018. "*Tensorflow Learn in 1 Day",* Krishna Rungta.

Schach, S. R., 2010. "*Object-oriented and Classical Software Engineering",* Mc Graw Hill.

Simarmata, J., 2010. "*Rekayasa Perangkat Lunak",* Andi.

Sommerville, I., 2011. "*Software Engineering",* 9th Edition, Pearson Education.

Stephens, R., 2015. "*Beginning Software Engineering",* John Wiley & Sons, Inc.

Suyanto, 2018. "*Machine Learning Tingkat Dasar Dan Tingkat Lanjut",* Informatika.

LAMPIRAN

Camera.py

```
import cv2
import numpy as np
import os
from PIL import Image
from Model import model
from keras.preprocessing.image import ImageDataGenerator
import sqlite3

class Camera:

    def __init__(self):
        self.vc = cv2.VideoCapture(0)
        self.face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
        self.user_id = None
        self.capture = False
        self.captureSuccess = False
        self.count = 0
        self.ids = []
        self.model = None
        self.font = cv2.FONT_HERSHEY_SIMPLEX
        self.currentUserId: -1

    def open(self):
        self.count = 0
        return self.vc.isOpened()


    def read(self):
        rval, frame = self.vc.read()
        frame = cv2.flip(frame, 1)
        if frame is not None:
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            frame1 = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            faces = self.face_detector.detectMultiScale(frame, 1.3, 5)
            for (x, y, w, h) in faces:
                cv2.rectangle(frame, (x, y), (x + w + 50, y + h + 50), (255, 0, 0), 2)
                if(self.capture):
                    self.count +=1
                    gray = frame1[y:y + h, x:x + w]
                    cv2.imwrite("dataset/User." + str(self.user_id) + '.' + str(self.count) + ".jpg",
gray)
                    if(self.count >120):
                        self.captureSuccess = True
                        self.capture = False
            return frame
    def captured(self):
        self.capture = True
        self.captureSuccess = False


    def initAbsen(self):
        from Model import model
        _, self.ids = self.getImagesAndLabels()
```

```
    self.model = model((32, 32, 1), len(set(self.ids)))
    self.model.load_weights('trained_model.h5')
    self.model.summary()
    self.currentUserId = -1
    self.startAbsen = False

def absenStart(self):
    self.startAbsen = True

def absen(self):
    ret, frame = self.vc.read()
    frame = cv2.flip(frame, 1)
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    nframe = frame
    faces = self.face_detector.detectMultiScale(
        frame,
        scaleFactor=1.1,
        minNeighbors=5,
        minSize=(30, 30))
    if(len(faces)<1):
        return nframe
    (x, y, w, h) = faces[0]
    frame = frame[y:y + h, x:x + w]
    frame = cv2.resize(frame, (32, 32))
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    c = cv2.waitKey(1)

    # gray = gray[np.newaxis,:,:,np.newaxis]
    gray = gray.reshape(-1, 32, 32, 1).astype('float32') / 255.
    prediction = self.model.predict(gray)
    # print("prediction:" + str(prediction))

    prediction = prediction.tolist()

    listv = prediction[0]
    n = listv.index(max(listv))

    if (self.startAbsen):
        self.currentUserId = n
    print("\n")
    print("-----------------------------------------------")
    print("Highest Probability: " + "User " +
          str(n) + "==>" + str(prediction[0][n]))

    print("-----------------------------------------------")
    print("\n")

    conn = sqlite3.connect('database.db')
    c = conn.cursor()
    # print(c)
    query = f"""SELECT * FROM user WHERE id={str(n)}"""
    c.execute(query)

    namaa = c.fetchone()
    print(namaa)

    c.close()
    conn.close()
```

```
        for (x, y, w, h) in faces:
            cv2.rectangle(nframe, (x, y), (x + w, y + h), (0, 255, 0), 2)
            cv2.putText(nframe, str(n), (x, y), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255,
0), 2)
        return nframe



    def getImagesAndLabels(self):
        path = 'dataset'
        imagePaths = [os.path.join(path, f) for f in os.listdir(path)]
        faceSamples = []
        self.ids = []

        for imagePath in imagePaths:

            # if there is an error saving any jpegs
            try:
                PIL_img = Image.open(imagePath).convert(
                    'L')  # convert it to grayscale
            except:
                continue
            img_numpy = np.array(PIL_img, 'uint8')

            id = int(os.path.split(imagePath)[-1].split(".")[1])
            faceSamples.append(img_numpy)
            self.ids.append(id)
        return faceSamples, self.ids
```

Popoup.py

```
from PyQt5 import QtCore, QtGui, QtWidgets

class Popup(QtWidgets.QMainWindow):
    def __init__(self, parent=None, type = None, nama = None, email=None, noTlp=None):
        super(Popup, self).__init__(parent)
        self.parent = parent
        self.nama = nama
        self.email = email
        self.noTlp = noTlp
        self.type = type

        self.setupUi()

    def setupUi(self):
        self.setObjectName("MainWindow")
        self.resize(711, 458)
        self.centralwidget = QtWidgets.QWidget(self)
        self.centralwidget.setObjectName("centralwidget")
        self.lblTitle = QtWidgets.QLabel(self.centralwidget)
        self.lblTitle.setGeometry(QtCore.QRect(0, 100, 711, 41))
        font = QtGui.QFont()
        font.setFamily("Arial")
        font.setPointSize(25)
        self.lblTitle.setFont(font)
        self.lblTitle.setAlignment(QtCore.Qt.AlignCenter)

        self.lblTitle.setObjectName("lblTitle")
        self.formLayoutWidget = QtWidgets.QWidget(self.centralwidget)
        self.formLayoutWidget.setGeometry(QtCore.QRect(220, 160, 212, 80))

        self.formLayoutWidget.setObjectName("formLayoutWidget")
        self.formLayout = QtWidgets.QFormLayout(self.formLayoutWidget)
        self.formLayout.setContentsMargins(0, 0, 0, 0)

        self.formLayout.setObjectName("formLayout")
        self.label_2 = QtWidgets.QLabel(self.formLayoutWidget)

        self.label_2.setObjectName("label_2")
        self.formLayout.setWidget(0, QtWidgets.QFormLayout.LabelRole, self.label_2)
        self.lblNama = QtWidgets.QLabel(self.formLayoutWidget)

        self.lblNama.setObjectName("lblNama")
        self.formLayout.setWidget(0, QtWidgets.QFormLayout.FieldRole, self.lblNama)
        self.label_4 = QtWidgets.QLabel(self.formLayoutWidget)

        self.label_4.setObjectName("label_4")
        self.formLayout.setWidget(1, QtWidgets.QFormLayout.LabelRole, self.label_4)
        self.lblEmail = QtWidgets.QLabel(self.formLayoutWidget)

        self.lblEmail.setObjectName("lblEmail")
        self.formLayout.setWidget(1, QtWidgets.QFormLayout.FieldRole, self.lblEmail)
        self.label_6 = QtWidgets.QLabel(self.formLayoutWidget)

        self.label_6.setObjectName("label_6")
```

```
        self.formLayout.setWidget(2, QtWidgets.QFormLayout.LabelRole, self.label_6)
        self.label_7 = QtWidgets.QLabel(self.formLayoutWidget)

        self.label_7.setObjectName("label_7")
        self.formLayout.setWidget(2, QtWidgets.QFormLayout.FieldRole, self.label_7)
        self.btnBack = QtWidgets.QPushButton(self.centralwidget)
        self.btnBack.setGeometry(QtCore.QRect(560, 380, 101, 41))

        self.btnBack.setObjectName("btnBack")
        self.setCentralWidget(self.centralwidget)

        self.btnBack.clicked.connect(self.back)
        self.retranslateUi()
        QtCore.QMetaObject.connectSlotsByName(self)

    def retranslateUi(self):
        _translate = QtCore.QCoreApplication.translate
        self.setWindowTitle(_translate("MainWindow", "Sistem Kehadiran"))
        self.lblTitle.setText(_translate("MainWindow", f"""Berhasil {self.type}"""))
        self.label_2.setText(_translate("MainWindow", "Nama"))
        self.lblNama.setText(_translate("MainWindow", f""": {self.nama if self.nama !="" else
"-"}"""))
        self.label_4.setText(_translate("MainWindow", "Email"))
        self.lblEmail.setText(_translate("MainWindow", f""": {self.email if self.email !="" else
"-"}"""))
        self.label_6.setText(_translate("MainWindow", "No Tlp"))
        self.label_7.setText(_translate("MainWindow", f""": {self.noTlp if self.noTlp !="" else
"-"}"""))
        self.btnBack.setText(_translate("MainWindow", "HOME"))

    def back(self):
        self.hide()
        self.parent.show()
```

Daftar.py

```
import sqlite3
from PyQt5 import QtCore, QtGui, QtWidgets
import sys

from PyQt5.QtCore import QThread, QTimer
from PyQt5.QtGui import QImage, QPixmap
from PyQt5.QtWidgets import QMessageBox

from Popup import Popup
from TrainingData import train


class DaftarWindow(QtWidgets.QMainWindow):
    def __init__(self, parent=None,camera=None):
        super(DaftarWindow, self).__init__(parent)
        self.parent = parent
        self.camera = camera
        self.timer = QTimer()
        self.timer.timeout.connect(self.nextFrameSlot)
        self.setupUi()
        self.start()
        self.camera.user_id = self.getLastUserId()


    def getLastUserId(self):
        try:
            sqliteConnection = sqlite3.connect('database.db')
            cursor = sqliteConnection.cursor()
            print("Successfully Connected to SQLite")

            query = "SELECT IIF(max(id) IS NULL,0,max(id) + 1) as id from user"

            result = cursor.execute(query)
            rows = result.fetchall()
            print(rows[0][0])
            return rows[0][0]
        except sqlite3.Error as error:
            print("Failed to insert data into sqlite table")
            print("Exception class is: ", error.__class__)
            print("Exception is", error.args)
            print('Printing detailed SQLite exception traceback: ')
            exc_type, exc_value, exc_tb = sys.exc_info()


    def setupUi(self):
        self.setObjectName("Sistem Kehadiran")
        self.resize(711, 458)
        font = QtGui.QFont()
        font.setFamily("Arial")
        self.setFont(font)
        self.centralwidget = QtWidgets.QWidget(self)
        self.centralwidget.setObjectName("centralwidget")
        self.horizontalLayoutWidget = QtWidgets.QWidget(self.centralwidget)
        self.horizontalLayoutWidget.setGeometry(QtCore.QRect(490, 110, 191, 31))
        self.horizontalLayoutWidget.setObjectName("horizontalLayoutWidget")
        self.horizontalLayout = QtWidgets.QHBoxLayout(self.horizontalLayoutWidget)
```

```
        self.horizontalLayout.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout.setObjectName("horizontalLayout")
        self.label = QtWidgets.QLabel(self.horizontalLayoutWidget)
        self.label.setObjectName("label")
        self.horizontalLayout.addWidget(self.label)
        spacerItem = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout.addItem(spacerItem)
        self.txtNama = QtWidgets.QLineEdit(self.horizontalLayoutWidget)
        self.txtNama.setText("")
        self.txtNama.setObjectName("txtNama")
        self.horizontalLayout.addWidget(self.txtNama)
        self.btnDaftar = QtWidgets.QPushButton(self.centralwidget)
        self.btnDaftar.setGeometry(QtCore.QRect(500, 60, 172, 31))
        font = QtGui.QFont()
        font.setPointSize(10)
        self.btnDaftar.setFont(font)
        self.btnDaftar.setObjectName("btnDaftar")
        self.horizontalLayoutWidget_2 = QtWidgets.QWidget(self.centralwidget)
        self.horizontalLayoutWidget_2.setGeometry(QtCore.QRect(490, 150, 191, 31))
        self.horizontalLayoutWidget_2.setObjectName("horizontalLayoutWidget_2")
        self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.horizontalLayoutWidget_2)
        self.horizontalLayout_2.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_2.setObjectName("horizontalLayout_2")
        self.label_2 = QtWidgets.QLabel(self.horizontalLayoutWidget_2)
        self.label_2.setObjectName("label_2")
        self.horizontalLayout_2.addWidget(self.label_2)
        spacerItem1 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_2.addItem(spacerItem1)
        self.txtEmail = QtWidgets.QLineEdit(self.horizontalLayoutWidget_2)
        self.txtEmail.setText("")
        self.txtEmail.setObjectName("txtEmail")
        self.horizontalLayout_2.addWidget(self.txtEmail)
        self.horizontalLayoutWidget_3 = QtWidgets.QWidget(self.centralwidget)
        self.horizontalLayoutWidget_3.setGeometry(QtCore.QRect(490, 190, 191, 31))
        self.horizontalLayoutWidget_3.setObjectName("horizontalLayoutWidget_3")
        self.horizontalLayout_3 = QtWidgets.QHBoxLayout(self.horizontalLayoutWidget_3)
        self.horizontalLayout_3.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_3.setObjectName("horizontalLayout_3")
        self.label_3 = QtWidgets.QLabel(self.horizontalLayoutWidget_3)
        self.label_3.setObjectName("label_3")
        self.horizontalLayout_3.addWidget(self.label_3)
        spacerItem2 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_3.addItem(spacerItem2)
        self.txtNoHp = QtWidgets.QLineEdit(self.horizontalLayoutWidget_3)
        self.txtNoHp.setText("")
        self.txtNoHp.setObjectName("txtNoHp")
        self.horizontalLayout_3.addWidget(self.txtNoHp)

        self.cameraContainer = QtWidgets.QLabel(self.centralwidget)
        self.cameraContainer.setGeometry(QtCore.QRect(20, 50, 431, 371))
        self.cameraContainer.setObjectName("cameraContainer")

        self.btnBack = QtWidgets.QPushButton(self.centralwidget)
        self.btnBack.setGeometry(QtCore.QRect(611, 420, 91, 31))
```

```
        self.setCentralWidget(self.centralwidget)

        self.btnBack.clicked.connect(self.back)

        self.btnDaftar.clicked.connect(self.daftar)

        self.setCentralWidget(self.centralwidget)

        self.retranslateUi()
        QtCore.QMetaObject.connectSlotsByName(self)

    def retranslateUi(self):
        _translate = QtCore.QCoreApplication.translate
        self.setWindowTitle(_translate("Absensi", "Sistem Kehadiran"))
        self.label.setText(_translate("Absensi", "Nama"))
        self.btnDaftar.setText(_translate("Absensi", "DAFTAR"))
        self.label_2.setText(_translate("Absensi", "Email"))
        self.label_3.setText(_translate("Absensi", "NO HP"))
        self.btnBack.setText(_translate("Absensi", "Home"))

    def saveUser(self):
        try:
            sqliteConnection = sqlite3.connect('database.db')
            cursor = sqliteConnection.cursor()
            print("Successfully Connected to SQLite")

            query = f"""INSERT INTO user
                            (id,nama,email,no_hp)
                             VALUES
('{self.getLastUserId()}','{self.txtNama.text()}','{self.txtEmail.text()}','{self.txtNoHp.text()}')"""

            count = cursor.execute(query)
            sqliteConnection.commit()
            print("Record inserted successfully into SqliteDb_developers table ",
cursor.rowcount)
            cursor.close()

        except sqlite3.Error as error:
            print("Failed to insert data into sqlite table")
            print("Exception class is: ", error.__class__)
            print("Exception is", error.args)
            print('Printing detailed SQLite exception traceback: ')
            exc_type, exc_value, exc_tb = sys.exc_info()

    def daftar(self):
        self.camera.captured()

    def back(self):
        self.hide()
        self.parent.show()
        self.timer.stop()

    def start(self):
        if not self.camera.open():
            print('failure')
            msgBox = QMessageBox()
            msgBox.setText("Failed to open camera.")
            msgBox.exec_()
```

```
            return
        self.timer.start(1000. / 24)

    def nextFrameSlot(self):
        if(self.camera.captureSuccess):
            self.camera.captureSuccess = False
            train()
            self.saveUser()
            self.timer.stop()
            popup =
Popup(self.parent,"DAFTAR",self.txtNama.text(),self.txtEmail.text(),self.txtNoHp.text())
            self.hide()
            popup.show()

        frame = self.camera.read()
        if frame is not None:
            image = QImage(
                frame, frame.shape[1], frame.shape[0], QImage.Format_RGB888)
            pixmap = QPixmap.fromImage(image)
            self.cameraContainer.setPixmap(pixmap)
```

TrainingData.py

```
from PIL import Image
import numpy as np
import cv2
import os
from keras.utils.np_utils import to_categorical
from keras import backend as K
from sklearn.model_selection import train_test_split
from keras import callbacks
from keras.preprocessing.image import ImageDataGenerator

path = 'dataset'

recognizer = cv2.face.LBPHFaceRecognizer_create()
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")


def downsample_image(img):
    img = Image.fromarray(img.astype('uint8'), 'L')
    img = img.resize((32, 32), Image.ANTIALIAS)
    return np.array(img)


def getImagesAndLabels(path):
    path = 'dataset'
    imagePaths = [os.path.join(path, f) for f in os.listdir(path)]
    faceSamples = []
    ids = []

    for imagePath in imagePaths:

        try:
            PIL_img = Image.open(imagePath).convert('L')
        except:
            continue
        img_numpy = np.array(PIL_img, 'uint8')

        id = int(os.path.split(imagePath)[-1].split(".")[1])
        faceSamples.append(img_numpy)
        ids.append(id)
    return faceSamples, ids

def train():
    from Model import model
    print("\n [INFO] Training faces now.")
    faces, ids = getImagesAndLabels(path)

    K.clear_session()
    n_faces = len(set(ids))
    model = model((32, 32, 1), n_faces)
    faces = np.asarray(faces)
    faces = np.array([downsample_image(ab) for ab in faces])
    ids = np.asarray(ids)
    faces = faces[:, :, :, np.newaxis]
```

```
ids = to_categorical(ids)

faces = faces.astype('float32')
faces /= 255.

x_train, x_test, y_train, y_test = train_test_split(
   faces, ids, test_size=0.2, random_state=0)

checkpoint = callbacks.ModelCheckpoint('trained_model.h5', monitor='val_accuracy',
                                save_best_only=True, save_weights_only=True, verbose=1)

model.fit(x_train, y_train,
        batch_size=32,
        epochs=5,
        validation_data=(x_test, y_test),
        shuffle=True, callbacks=[checkpoint])


print("\n [INFO] " + str(n_faces) + " faces trained. Exiting Program")
```

Dataset Maker.py

```
import cv2
import os
import numpy as np
from PIL import Image, ImageEnhance
from sklearn.datasets import fetch_lfw_people


def downsample_image(img):
    img = Image.fromarray(img.astype('uint8'), 'L')
    img = img.resize((32,32), Image.ANTIALIAS)
    return np.array(img)

img_list = os.listdir()

x_data = []
y_data = []


def get_face_data():
    people = fetch_lfw_people(color=False, min_faces_per_person=300)
    X_faces = people.images
    X_faces = np.array([downsample_image(ab) for ab in X_faces])
    Y_faces = people.target
    names = people.target_names
    return X_faces,Y_faces,names


X_faces,Y_faces,names = get_face_data()


for img in img_list:
    image = cv2.imread(img,0)
    image = downsample_image(image)
    print(str(image.shape))
    x_data.append(image)
    y_data.append(int(1))


x_data = np.asarray(x_data)
#x_data = x_data[:,:,:,np.newaxis]
y_data = np.asarray(y_data)


np.save('x_data.npy',x_data)
np.save('y_data.npy',y_data)


x_data = np.load('x_data.npy')
y_data = np.load('y_data.npy')


a = np.concatenate([X_faces,x_data])
b = np.concatenate([Y_faces,y_data])


np.save('x_data.npy',a)
```

```
np.save('y_data.npy',b)
```

Model.py

```
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Dropout, BatchNormalization
from keras.layers import Dense, Activation, Flatten


def model(input_shape, num_classes):
    model = Sequential()

    model.add(Conv2D(32, (3, 3), input_shape=input_shape))
    model.add(Activation("relu"))

    model.add(Conv2D(64, (3, 3)))
    model.add(BatchNormalization())
    model.add(Activation("relu"))

    model.add(Conv2D(64, (1, 1)))
    model.add(Dropout(0.5))
    model.add(BatchNormalization())
    model.add(Activation("relu"))

    model.add(Conv2D(128, (3, 3)))
    model.add(Dropout(0.5))
    model.add(Activation("relu"))

    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(64, (1, 1)))
    model.add(Activation("relu"))

    model.add(Flatten())
    model.add(Dense(32))
    model.add(Dense(num_classes))
    model.add(Activation("softmax"))

    model.compile(loss='categorical_crossentropy',
                  optimizer='adam',
                  metrics=['accuracy'])

    model.summary()
    return model
```

Absen.py

```
import sqlite3
from PyQt5 import QtCore, QtGui, QtWidgets
import sys
import calendar, time

from PyQt5.QtCore import QTimer
from PyQt5.QtGui import QImage, QPixmap
from PyQt5.QtWidgets import QMessageBox

from Popup import Popup


class AbsenWindow(QtWidgets.QMainWindow):
    def __init__(self, parent=None,camera=None):
        super(AbsenWindow, self).__init__(parent)
        self.parent = parent
        self.camera = camera
        self.setupUi()

        self.timer = QTimer()
        self.timer.timeout.connect(self.nextFrameSlot)
        self.start()
    def setupUi(self):
        self.setObjectName("Sistem Kehadiran")
        self.resize(711, 458)
        font = QtGui.QFont()
        font.setFamily("Arial")
        self.setFont(font)
        self.centralwidget = QtWidgets.QWidget(self)
        self.centralwidget.setObjectName("centralwidget")
        self.btnAbsen = QtWidgets.QPushButton(self.centralwidget)
        self.btnAbsen.setGeometry(QtCore.QRect(500, 60, 172, 31))
        font = QtGui.QFont()
        font.setPointSize(10)
        self.btnAbsen.setFont(font)
        self.btnAbsen.setObjectName("btnAbsen")

        self.cameraContainer = QtWidgets.QLabel(self.centralwidget)
        self.cameraContainer.setGeometry(QtCore.QRect(20, 50, 431, 371))
        self.cameraContainer.setObjectName("cameraContainer")

        self.btnBack = QtWidgets.QPushButton(self.centralwidget)
        self.btnBack.setGeometry(QtCore.QRect(611, 420, 91, 31))

        font = QtGui.QFont()
        font.setPointSize(10)
        self.btnBack.setFont(font)
        self.btnBack.setObjectName("btnBack")
        self.setCentralWidget(self.centralwidget)

        self.btnBack.clicked.connect(self.back)
        self.btnAbsen.clicked.connect(self.absen)
        self.retranslateUi()
        QtCore.QMetaObject.connectSlotsByName(self)

    def absen(self):
```

```
    self.camera.absenStart()


def start(self):
    if not self.camera.open():
        print('failure')
        msgBox = QMessageBox()
        msgBox.setText("Failed to open camera.")
        msgBox.exec_()
        return
    self.camera.initAbsen()
    self.timer.start(1000. / 24)
def getUser(self,id=None):
    try:
        sqliteConnection = sqlite3.connect('database.db')
        c = sqliteConnection.cursor()
        print("Successfully Connected to SQLite")

        query = f"""SELECT id,nama,email,no_hp FROm user WHERE id={id}"""

        result = c.execute(query)
        rows = result.fetchall()

        return rows[0]
    except sqlite3.Error as error:
        print("Failed to insert data into sqlite table")
        print("Exception class is: ", error.__class__)
        print("Exception is", error.args)
        print('Printing detailed SQLite exception traceback: ')
        exc_type, exc_value, exc_tb = sys.exc_info()

def rekamUser(self,id=None):
    try:
        sqliteConnection = sqlite3.connect('database.db')
        c = sqliteConnection.cursor()
        querytemp = f"""SELECT nama FROM user WHERE id='{id}'"""
        resulttemp = c.execute(querytemp)
        namaa = c.fetchall()
        # print("asdfasdfasdf", namaa)
        # print("asdfasdfasdf", id)
        localtime = time.asctime(time.localtime(time.time()))
        # print("asdfasdfasdf", localtime)

        query = f"""INSERT INTO rekam (id,waktu)  VALUES  ('{id}', '{localtime}')"""
        result2 = c.execute(query)
        sqliteConnection.commit()
        print("Record inserted successfully into SqliteDb_developers table ", c.rowcount)
        c.close()

    except sqlite3.Error as error:
        print("Failed to insert data into sqlite table")
        print("Exception class is: ", error.__class__)
        print("Exception is", error.args)
        print('Printing detailed SQLite exception traceback: ')
        exc_type, exc_value, exc_tb = sys.exc_info()

def nextFrameSlot(self):
```

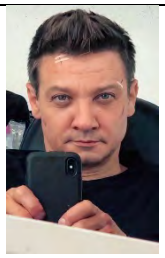
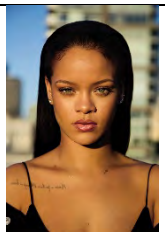
```
        if(self.camera.currentUserId>-1):
            self.timer.stop();
            user = self.getUser(self.camera.currentUserId)
            popup = Popup(self.parent, "REKAM KEHADIRAN", user[1], user[2], user[3])
            self.hide()
            self.rekamUser(self.camera.currentUserId)
            popup.show()

        frame = self.camera.absen()
        if frame is not None:
            image = QImage(
                frame, frame.shape[1], frame.shape[0], QImage.Format_RGB888)
            pixmap = QPixmap.fromImage(image)
            self.cameraContainer.setPixmap(pixmap)

    def retranslateUi(self):
        _translate = QtCore.QCoreApplication.translate
        self.setWindowTitle(_translate("Absensi", "Sistem Kehadiran"))
        self.btnAbsen.setText(_translate("Absensi", "ABSEN"))
        self.btnBack.setText(_translate("Absensi", "Home"))

    def back(self):
        self.hide()
        self.parent.show()
        self.timer.stop()
```

main.py

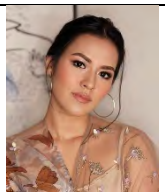
```
from PyQt5 import QtCore, QtGui, QtWidgets
import sys
import Daftar
import Absen
from Camera import Camera



class Ui_Absensi(QtWidgets.QMainWindow):
    def __init__(self):
        super().__init__()

        self.camera = Camera()

    def setupUi(self):
        self.setObjectName("Sistem Kehadiran")
        self.resize(711, 458)
        self.centralwidget = QtWidgets.QWidget(self)
        self.centralwidget.setObjectName("centralwidget")
        self.label = QtWidgets.QLabel(self.centralwidget)
        self.label.setGeometry(QtCore.QRect(0, 90, 711, 41))
        font = QtGui.QFont()
        font.setFamily("Arial")
        font.setPointSize(20)
        font.setBold(False)
        font.setWeight(50)
        self.label.setFont(font)
        self.label.setTextFormat(QtCore.Qt.PlainText)
        self.label.setScaledContents(False)
        self.label.setAlignment(QtCore.Qt.AlignCenter)
        self.label.setObjectName("label")
        self.btnDaftar = QtWidgets.QPushButton(
        self.centralwidget)
        self.btnDaftar.setGeometry(QtCore.QRect(260, 150, 151, 31))
        font = QtGui.QFont()
        font.setFamily("Arial")
        font.setPointSize(10)
        self.btnDaftar.setFont(font)
        self.btnDaftar.setObjectName("btnDaftar")
        self.btnAbsen = QtWidgets.QPushButton(self.centralwidget)
        self.btnAbsen.setGeometry(QtCore.QRect(260, 190, 151, 31))
        font = QtGui.QFont()
        font.setFamily("Arial")
        font.setPointSize(10)
        self.btnAbsen.setFont(font)
        self.btnAbsen.setObjectName("btnAbsen")
        self.setCentralWidget(self.centralwidget)

        self.btnDaftar.clicked.connect(self.daftar)
        self.btnAbsen.clicked.connect(self.absen)

        self.retranslateUi()
        QtCore.QMetaObject.connectSlotsByName(self)

    def retranslateUi(self):
```

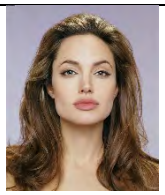
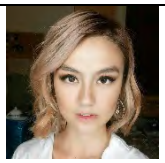
```
        _translate = QtCore.QCoreApplication.translate
        self.setWindowTitle(_translate("Absensi", "Sistem Kehadiran"))
        self.label.setText(_translate("Absensi", "APLIKASI SISTEM KEHADIRAN"))
        self.btnDaftar.setText(_translate("Absensi", "DAFTAR"))
        self.btnAbsen.setText(_translate("Absensi", "KEHADIRAN"))

    def daftar(self):
        self.hide()
        window= Daftar.DaftarWindow(self,self.camera).show()

    def absen(self):
        self.hide()
        window= Absen.AbsenWindow(self,self.camera).show()



if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    ui = Ui_Absensi()
    ui.setupUi()
    ui.show()
    sys.exit(app.exec_())
```

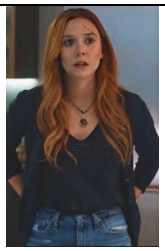
Tabel Hasil Pengujian Sistem Kehadiran

No.	Wajah *User* Terdaftar	Nama *User* Terdaftar	Hasil Sistem Kehadiran
1		Scarlet	Benar
2		Wendy	Benar
3		Pevita	Benar
4		Hawk	Benar
5		Rihanna	Benar
6		Bruce	Benar

No.	Wajah *User* Terdaftar	Nama *User* Terdaftar	Hasil Sistem Kehadiran
7		Kim	Benar
8		Dawyne	Benar
9		Rose	Benar
10		Tesa	Benar
11		Sungkang	Benar
12		Antman	Benar
13		Michelle	Benar

No.	Wajah *User* Terdaftar	Nama *User* Terdaftar	Hasil Sistem Kehadiran
14		Raisa	Benar
15		Mark	Benar
16		Luhut	Benar
17		Jokowi	Benar
18		Iqbal	Benar
19		Maudy	Benar
20		Elon	Benar

No.	Wajah *User* Terdaftar	Nama *User* Terdaftar	Hasil Sistem Kehadiran
21		Bradd	Benar
22		Jolie	Benar
23		Agnes	Benar
24		Erick	Benar
25		Alex	Benar
26		Ria	Benar
27		Gracia	Benar

No.	Wajah *User* Terdaftar	Nama *User* Terdaftar	Hasil Sistem Kehadiran
28		Captain	Benar
29		Wanda	Benar
30		Evan	Benar