DETEKSI PENYAKIT BUAH JERUK MENGGUNAKAN METODE ALGORITMA *CONVOLUTIONAL NEURAL NETWORK* DENGAN ARSITEKTUR *MOBILENET* V2

SKRIPSI

Disusun sebagai bentuk pelaporan penelitian skripsi dan salah satu syarat menempuh Sarjana Strata 1 (S1)

Disusun oleh:

Eka Muspita Dewi

NPM. 3332190026

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS SULTAN AGENG TIRTAYASA

2025

LEMBAR PERNYATAAN KEASLIAN SKRIPSI

Dengan ini saya sebagai penulis Skripsi berikut:

Judul	:	Deteksi Penyakit Buah Jeruk Menggunakan Metode Algoritma *Convolutional Neural Network* Dengan Arsitektur *Mobilenet* V2
Nama	:	Eka Muspita Dewi
NPM	:	3332190026
Fakultas/Jurusan	:	Teknik/Teknik Elektro

Menyatakan dengan sesungguhnya bahwa Skripsi tersebut di atas adalah benar-benar hasil karya asli saya dan tidak memuat hasil karya orang lain, kecuali dinyatakan melalui rujukan yang benar dan dapat dipertanggungjawabkan. Apabila dikemudian hari ditemukan hal-hal yang menunjukkan bahwa sebagian atau seluruh karya ini bukan karya saya, maka saya bersedia dituntut melalui hukum yang berlaku. Saya juga bersedia menanggung segala akibat hukum yang timbul dari pernyataan yang secara sadar dan sengaja saya nyatakan melalui lembar ini.

Serang, 30 Desember 2024

Eka Muspita Dewi

3332190026

LEMBAR PENGESAHAN

Dengan ini ditetapkan bahwa Skripsi berikut.

Judul	:	Deteksi Penyakit Buah Jeruk Menggunakan Metode Algoritma *Convolutional Neural Network* Dengan Arsitektur *Mobilenet* V2
Nama	:	Eka Muspita Dewi
NPM	:	3332190026
Fakultas/Jurusan	:	Teknik/Teknik Elektro

Telah diuji dan dipertahankan pada tanggal 9 Januari 2025 Melalui Sidang Skripsi di Fakultas Teknik, Universitas Sultan Ageng Tirtayasa, Cilegon, Banten dan dinyatakan LULUS/TIDAK LULUS.

		Dewan Penguji	Tanda Tangan
Pembimbing I	:	Masjudin, ST., M.Eng	.....
Pembimbing II	:	Fadil Muhammad, ST., MT	.....
Penguji I	:	Dr. Irma Saraswati, S.Si., M.T.	.....
Penguji I	:	H. Alief Maulana, S.T., M.T.	.....

Mengetahui
Ketua Jurusan Teknik Elektro

(Dr. Eng. Rocky Alfanz, S.T., M.Sc.)
NIP. 198103282010121001

PRAKATA

Puji dan syukur saya panjatkan kepada Allah Swt. atas rahmat dan hidayahnya sehingga saya dapat menyelesaikan skripsi ini. skripsi ini bertujuan untuk memenuhi syarat dalam menyelesaikan program Pendidikan starta-1 (S1) pada program studi Teknik Elektro Fakultas Teknik Universitas Sultang Ageng Tirtayasa. Saya menyadari banyak pihak yang memberikan dukungan dan bantuan selama proses magang hingga penyusunan laporan akhir ini. Oleh karena itu, dengan penuh hormat penulis mengucapkan terima kasih sebanyak-banyaknya kepada:

1. Dr. Eng Rocky Alfanz, S.T., M.Sc selaku Ketua Jurusan Teknik Elektro Fakultas Teknik Universitas Sultan Ageng Tirtayasa.
2. Fadhil Muhammad, S.T., M.T selaku Dosen Pembimbing Akademik dan selaku pembimbing 2 skripsi saya
3. Masjudin, S.T., M.Eng., selaku Dosen Pembimbing 2 Skripsi saya yang telah menyediakan waktu, tenaga dan pikiran sehingga penulis dapat menyelesaikan proyek akhir dengan baik.
4. Kedua orang tua serta seluruh keluarga yang telah memberikan dukungan, nasihat, doa dan materi yang tak terhingga nilainya.

Penulis berharap agar skripsi ini dapat memberikan manfaat baik bagi pembaca maupun bagi penulis itu sendiri. Penulis menyadari bahwa dalam penyusunan skripsi ini masih terdapat berbagai kekurangan. Oleh karena itu, penulis sangat mengharapkan adanya kritik dan saran yang membangun untuk perbaikan laporan ini.

Serang, 30 November 2023

Eka Muspita Dewi

3332190026

ABSTRAK

Eka Muspita Dewi 3332190026

Teknik Elektro

Deteksi Penyakit Buah Jeruk Menggunakan Metode Algoritma *Convolutional Neural Network* Dengan Arsitektur *MobileNetV2*

Revolusi Industri Keempat, atau *Industrial Revolution* 4.0, membawa perubahan signifikan dalam sektor pertanian melalui *Agriculture* 4.0. Teknologi informasi dan otomasi meningkatkan produktivitas dan keberlanjutan, terutama pada tanaman jeruk yang penting di Indonesia. Namun, petani jeruk menghadapi tantangan besar dari hama dan penyakit akibat kurangnya pengetahuan pengendalian yang tepat, serta metode diagnosa manual yang kurang akurat. Penelitian ini memanfaatkan kecerdasan buatan (AI) dengan metode *Deep Learning*, khususnya *Convolutional Neural Network* (CNN) menggunakan arsitektur *MobileNetV2*, untuk mendeteksi tiga buah penyakit buah jeruk dan membandingkan dengan buah jeruk yang tidak terkena penyakit. Hasilnya menunjukkan akurasi 0.9612 dan performa yang baik dalam metrik presisi, *recall*, dan *F1-score*, membuktikan bahwa teknologi AI dapat efektif mendukung petani jeruk dalam meningkatkan kualitas dan kuantitas produksi.

Kata kunci: Kecerdasan Buatan, *Convolutional Neural Network*, *MobileNetV2*

ABSTRACT

Eka Muspita Dewi 3332190026

Electrical Engineering

Detection of citrus fruit diseases using the convolutional neural network algorithm with the MobileNetV2 architecture.

The Fourth Industrial Revolution, or Industry 4.0, is bringing significant changes to the agricultural sector through Agriculture 4.0. Information technology and automation are enhancing productivity and sustainability, especially in orange cultivation, which is crucial in Indonesia. However, orange farmers face major challenges from pests and diseases due to a lack of proper control knowledge and inaccurate manual diagnostic methods. This research utilizes artificial intelligence (AI) with *Deep learning* methods, specifically Convolutional Neural Network (CNN) using the MobileNetV2 architecture, to detect four types of orange diseases. The results show an accuracy of 0.9612 and strong performance in precision, recall, and F1-score metrics, demonstrating that AI technology can effectively support orange farmers in improving the quality and quantity of their production.

Keywords: Artificial Intelligence, Convolutional Neural Network, MobileNetV2

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Revolusi Industri Keempat, dikenal juga sebagai Industrial Revolution 4.0, menjadi perhatian banyak pihak saat ini. Baik pemerintah, industri, maupun perusahaan berlomba-lomba mempersiapkan diri untuk menghadapinya. Singkatnya, revolusi ini menunjukkan bagaimana teknologi seperti artificial intelligence, kendaraan tanpa pengemudi, dan internet saling berintegrasi serta memengaruhi kehidupan manusia [1].

Industri yang terdampak oleh revolusi industri ini meliputi berbagai sektor yaitu Teknologi Pertanian *Revolusioner* atau biasa disebut *Agriculture* 4.0. Dimana *Agriculture* 4.0 sendiri merujuk pada revolusi industri keempat di sektor pertanian. Konsep ini menggabungkan teknologi informasi, komunikasi, dan otomasi untuk meningkatkan produktivitas, efisiensi, dan keberlanjutan dalam pertanian [2]. Salah satu industri pertaniannya atau agriculture 4.0 yaitu pertanian tanaman jeruk. Jeruk adalah tanaman tahunan yang tergolong dalam famili Rutaceae. Buahnya, yang populer sebagai salah satu tanaman hortikultura, sangat digemari dan dikonsumsi oleh masyarakat Indonesia. Dengan kandungan gizi yang kaya, terutama vitamin C, jeruk berfungsi sebagai antioksidan yang dapat membantu mencegah berbagai penyakit, seperti kanker, gangguan jantung, dan penuaan dini [3]. Jeruk memiliki banyak keunggulan termasuk sebagai sumber makanan, bahan baku untuk agroindustri, penghasil pendapatan, dan sumber lapangan kerja terutama bagi kaum miskin di pedesaan [4]. Petani tanaman Jeruk tidak lepas dari ancaman penyakit yang biasa mengganggu bahkan menghentikan produksi Jeruk sehingga mengharuskan para pengusaha agribisnis maupun para petani kebun mencari cara untuk mengantisipasi menyebarnya penyakit pada tanaman jeruk [5].

Kurangnya pengetahuan petani mengenai hama dan penyakit yang menyerang tanaman jeruk, serta metode pengendalian yang tepat, berdampak pada kerugian bagi petani karena menghasilkan jeruk dengan kualitas rendah saat panen [6]. Menurut buku panduan abad 21 tentang jamur, karena penyakit pada jeruk, produksi dan budidaya jeruk menimbulkan kerugian

ekonomi yang signifikan di seluruh wilayah pertumbuhan jeruk di dunia dan terus terancam oleh banyak faktor, seperti patogen dan hama [7]. Salah satu masalah yang dihadapi adalah bahwa proses diagnosis penyakit pada buah jeruk masih dilakukan secara manual, yang menyebabkan akurasi yang tidak stabil dan cenderung subjektif. Perbedaan pandangan ini dapat mengakibatkan hasil diagnosis yang bervariasi. [8]. Dengan demikian, dibutuhkan teknologi yang dapat membantu petani jeruk Menganalisis gambar penyakit pada buah jeruk dengan cepat dan akurat, menggunakan teknologi informasi berbasis komputer serta data seperti Kecerdasan Buatan (AI). Penggunaan AI dalam berbagai sektor kehidupan menjadi solusi untuk memenuhi tuntutan masyarakat saat ini. Kehadiran AI, dengan berbagai inovasi canggih dan kreatif, Memberikan pengaruh besar pada setiap aspek kehidupan manusia [9]. Saat ini AI telah banyak digunakan di berbagai bidang seperti pertanian [10].

Salah satu teknologi AI yang memiliki pengaruh besar adalah *Deep Learning.* Salah satu metode Deep Learning yang sering digunakan adalah Convolutional Neural Network (CNN), yang mampu mengekstraksi fitur secara otomatis dan efisien, sehingga menghasilkan klasifikasi yang lebih tepat. [11]. *Deep Learning* saat ini berkembang pesat dalam berbagai aspek, seperti ekstraksi fitur, interaksi antar fitur, dan representasi data. Teknologi ini adalah kecerdasan buatan yang meniru cara kerja otak manusia. Metode ini sangat membantu aktivitas sehari-hari, karena dapat memproses data untuk menghasilkan pola yang berguna dalam pengambilan keputusan atau pelaksanaan tugas tertentu [12]. Untuk mencapai hasil yang tepat, algoritma CNN digunakan. CNN adalah jenis jaringan saraf yang umum diterapkan dalam proses klasifikasi citra [13].

Penelitian yang mengaplikasikan CNN untuk menganalisis penyakit pada buah jeruk dilakukan oleh Dwiretno Istiyadi Swasono, Mohammad Abuemas Rizq Wijaya, dan Muhamad Arief Hidayat pada tahun 2023 [14]. Penelitian ini menggunakan dataset yang terdiri dari 1790 gambar penyakit buah jeruk yang dikelompokkan dalam 4 kategori, yaitu bercak hitam, kanker, hijau, dan buah sehat. Hasil terbaik diperoleh dengan skenario 90% data latih dan 10% data validasi, dengan akurasi 94,34%, presisi 93,0%, recall 94,0%, dan F1-score 95,0%. Penelitian ini membuktikan bahwa metode Convolutional Neural Network (CNN)

efektif untuk mengklasifikasikan penyakit pada buah jeruk. Model CNN yang diusulkan dimaksudkan untuk membedakan buah yang sehat dengan buah yang terkena penyakit seperti bercak hitam, kanker, dan penyakit jeruk lainnya. Model CNN yang diusulkan bertujuan untuk mengekstrak fitur diskriminatif yang komplementer dengan mengintegrasikan beberapa lapisan[15]

Berdasarkan penjelasan sebelumnya, peneliti akan menggunakan *Deep learning* dengan metode CNN dengan arsitektur *mobilenet*V2. MobileNetV2 adalah arsitektur CNN yang dirancang untuk mengurangi kebutuhan sumber daya komputasi yang besar. Dikembangkan oleh peneliti Google, arsitektur ini memungkinkan penggunaan pada perangkat mobile atau ponsel [16]. Penelitian ini bertujuan untuk mengklasifikasikan penyakit pada buah jeruk, yaitu *blackspot*, *canker*, *grenning*, dan buah sehat, dengan menerapkan beberapa skenario pada arsitektur MobileNetV2 untuk memperoleh hasil yang optimal.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah di atas, dirumuskan pokok permasalahan yang menjadi fokus kajian dalam penelitian ini :

1. Bagaimana merancang sistem deteksi penyakit pada buah jeruk menggunakan metode algoritma CNN dengan arsitektur *MobilenetV2*?
2. Bagaimana memahami tingkat keakurasian dan nilai efektifitas sistem pada deteksi penyakit buah jeruk dengan menerapkan algoritma CNN dengan arsitektur *MobilenetV2*?

1.3 Tujuan Penelitian

penelitian ini memiliki beberapa tujuan yang ingin dicapai, yaitu :

1. Membuat sistem yang dapat mendeteksi penyakit buah jeruk dan membandingkan dengan buah jeruk yang tidak terkena penyakit dengan menerapkan Algoritma CNN dengan arsitektur *MobilenetV2*.
2. Memahami tingkat keakurasian dan nilai efektifitas sistem pada deteksi penyakit buah jeruk dan membandingkan buah jeruk yang tidak terkena penyakit dengan menerapkan algoritma CNN dengan arsitektur *MobilenetV2.*

1.4 Manfaat Penelitian

Sebuah penelitian dilakukan untuk mencapai suatu manfaat, adapun manfaat dari penelitian ini agar memenuhi mata kuliah skripsi:

1. Bagi akademisi, penelitian ini diharap bisa menambahkan wawasan ilmu serta penangkapan mahasiswa saat menerapkan algoritma CNN untuk pendeteksian penyakit buah jeruk, serta memahami cara mengukur kinerja sistem yang dirancang menggunakan metrik seperti akurasi, presisi, *recall*, dan *F1-score*.
2. Bagi petani dan masyarakat, melalui penelitian ini diharapkan dapat memberikan kontribusi dalam peningkatan pertanian berkelanjutan dengan memberikan sistem yang efektif dalam mendeteksi penyakit pada buah jeruk dan dapat membantu petani dan masyarakat luas untuk mengambil tindakan pencegahan yang tepat dan mengurangi kerugian hasil panen.

1.5 Batasan Masalah

Penelitian ini memiliki beberapa batasan masalah yang dapat dijelaskan sebagai berikut :

1. Hanya mendeteksi buah jeruk dengan penyakit seperti *black spot, canker, greening* dan membandungkan dengan buah jeruk yang tidak terkena penyakit.
2. Metode yang digunakan adalah Algoritma CNN dengan arsitektur *Mobilenet*V2.
3. Data yang dipergunakan berupa data citra yang diperoleh dari kaggle dengan 2 sumber *Dataset* yang berbeda.
4. Citra yang digunakan dengan format *Joint Photographic Group* (JPG).
5. Menggunakan bahasa pemrograman phyton.
6. Sistem dijalankan dengan bantuan *command prompt*.

1.6 Sistematika Penulisan

Pada sistematika penulisan yaitu urutan garis besar dari penulisan skripsi dengan rincian seperti:

BAB I PENDAHULUAN

Bab ini berisi penjelasan mengenai latar belakang masalah dalam penelitian ini, tujuan penelitian, manfaat penelitian, rumusan masalah, batasan masalah, dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini memuat teori-teori yang diperlukan dalam penelitian ini, di antaranya mengenai, Buah Jeruk, CNN, *mobilenetV2, command prpomt* dan pengolahan citra digital.

BAB III METODOLOGI PENELITIAN

Bab ini menjelaskan metodologi penelitian, yang meliputi diagram alir penelitian, komponen-komponen penelitian, metode pengumpulan data, waktu, serta objek penelitian.

BAB IV PEMBAHASAN

Bab ini berisi tentang pengujian serta analisis pada hasil yang diperoleh dari simulasi sistem deteksi penyakit buah jeruk menggunakan metode algoritma CNN dengan arsitektur *mobilenetV2.*

BAB V PENUTUP

Bab ini memberikan kesimpulan serta saran demi mengembangkan penelitian lebih lanjut.

BAB II
TINJAUAN PUSTAKA

2.1 Buah Jeruk

Jeruk adalah tanaman buah tahunan yang berasal dari Asia, dengan Cina sebagai tempat asalnya. Di Indonesia, jeruk telah lama dibudidayakan, termasuk jenis jeruk manis dan keprok yang dibawa oleh Belanda dari Amerika dan Italia. Buah ini kaya vitamin C dan A, sehingga populer di masyarakat [17][18]. Sebagai komoditas hortikultura prioritas, jeruk memberikan keuntungan tinggi bagi petani, baik sebagai buah segar maupun bahan olahan, menjadikannya sumber pendapatan yang menjanjikan [19].

Jeruk menjadi pilihan favorit karena rasanya yang segar dan berbagai manfaat kesehatan. Selain itu, jeruk tidak tergantung musim, karena dapat berbuah dua kali dalam setahun dengan hasil panen yang melimpah, asalkan dirawat dengan baik untuk menghasilkan buah berkualitas. Jeruk memiliki kemampuan untuk tumbuh dan berbuah di berbagai ketinggian, baik di dataran rendah maupun tinggi, serta dapat ditanam di lahan sawah maupun tegalan. Prospek agribisnis jeruk sangat menjanjikan, didukung oleh potensi lahan yang meliputi jutaan hektar untuk perkebunan skala besar dengan memperhatikan kesesuaian agroklimat. Dengan pengelolaan intensif, produksi jeruk dapat meningkat signifikan. Selain itu, potensi pasar yang besar dipengaruhi oleh kenaikan pendapatan, pertumbuhan populasi, dan elastisitas pendapatan terhadap permintaan [20].

Citrus sp atau jeruk, termasuk tanaman angiospermae yang dibudidayakan secara konvensional atau melalui pemuliaan. Teknik fusi protoplas dikembangkan untuk mengatasi ketidakcocokan genetik, menyilangkan tanaman lewat sel somatik. Jeruk hasilnya perlu diuji ketahanan terhadap hama dan penyakit [21].

2.2 Penyakit Buah Jeruk

Salah satu tantangan utama dalam budidaya jeruk adalah serangan hama dan penyakit. Hama tanaman adalah organisme atau hewan yang, dalam jumlah tertentu, dapat menghambat pertumbuhan tanaman dan menyebabkan kerugian secara ekonomi. Penyakit tanaman sebagai ketidakmampuan tanaman untuk

menjalankan fungsi fisiologisnya akibat gangguan terus-menerus yang disebabkan oleh patogen atau faktor lain, sehingga memunculkan gejala penyakit. Sebagian besar patogen yang menyerang tanaman berasal dari kelompok jamur, bakteri, dan virus, yang berdampak pada penurunan hasil produksi [22].

Tanaman jeruk mudah terpapar berbagai penyakit yang dapat mengurangi kualitas dan kuantitas hasil produksi. bahkan menghancurkan perkebunan. Penyakit utama yang menyerang budidaya jeruk termasuk CVPD (*citrus vein phloem degeneration*), disebabkan oleh bakteri *Liberibacter asiatus* virus Tristeza, penyakit blendok akibat jamur *Botryodiplodia theobromae*, serta busuk pangkal batang akibat *Phytophthora citrophthora* [23]. Gambar 2.1 berikut ini merupakan beberapa gambar penyakit buah jeruk yang sering terjadi selama proses budidaya.

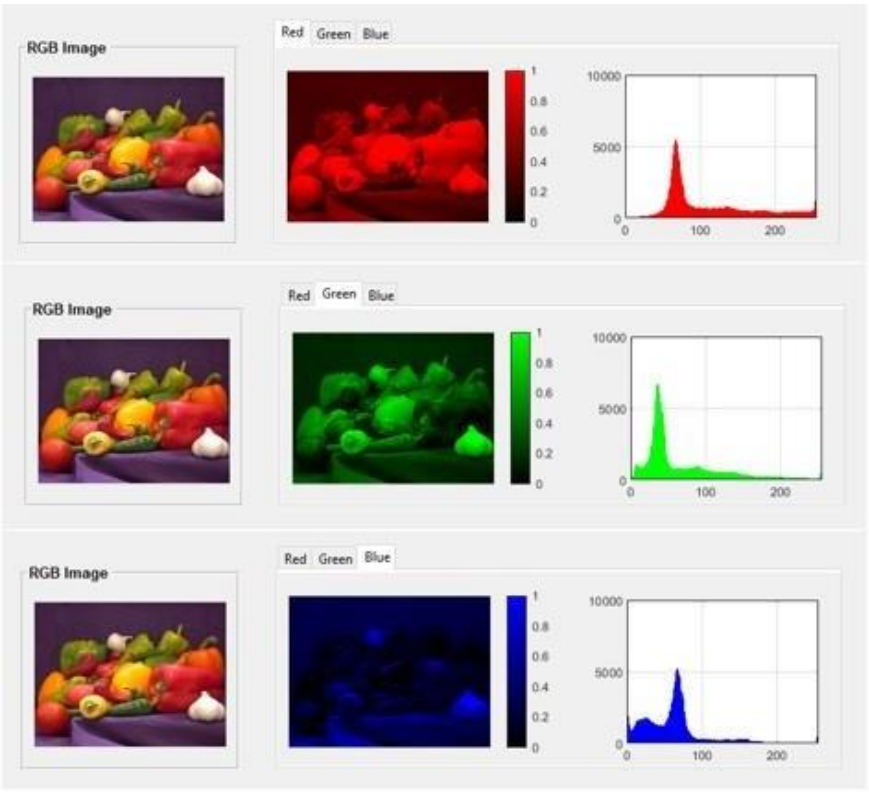
Gambar 2.1 Tanaman Jeruk Yang Terinfeksi Oleh L. Asiaticus [24]

Gambar 2.1 diatas merupakan gambar buah jeruk yang terinfeksi oleh L.Asiaticus. Kualitas buah dari tanaman jeruk yang terinfeksi oleh L. asiaticus menurun, menghasilkan buah yang lebih kecil dengan kandungan vitamin C yang berkurang [24].

2.3 Pengolahan Citra Digital

Pemrosesan citra atau *image processing* adalah teknik untuk mengolah citra menggunakan komputer agar kualitasnya meningkat. Tujuan dari pemrosesan ini adalah untuk mengurangi gangguan atau noise pada citra Sehingga lebih mudah dipahami dan dianalisis [26]. Citra digital adalah representasi dari objek dalam bentuk gambar yang diambil atau direkam, baik dalam bentuk foto, sinyal video analog seperti pada televisi, maupun dalam format digital yang dapat disimpan pada media penyimpanan [25]. Pada pengolahan citra digital kumpulan titik kecil yang dibentuk menjadi citra digital dan titik kecil tersebut merupakan sebuah piksel [27].

Suatu citra atau gambar bisa diarti fungsikan dengan berkuran *M* baris dan *N* kolom, serta juga ada kordinat spasial serta amplitudo pada titik kordinat dan dinamakan sebagai intensitas tingkat keabuan. Koordinat spasial dan *value* amplitudo *f* seluruhnya diskrit dan berhingga maka citra tersebut merupakan citra *digial* [26]. Citra warna mempunyai 3 jenis bagian warna disetiap pikselnya yakni *red*, *green* dan *blue*. Warna yang berada pada setiap piksel hasil dari gabungan ketiga warna utama yakni merah hijau serta biru yang dikumpulkan pada ruang atau bidang warna. Untuk format *file* grafis itu sendiri *24 bit* dalam menyimpan citra warna dan yang bersumber dari komponen utama yaitu merah hijau biru dengan masing-masingnya bernilai 8 *bit* [28] . Gambar 2.2 merupakan contoh RGB pengolahan citra digital.



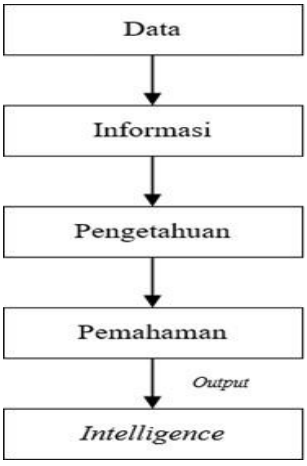
Gambar 2.2 RGB Pengolahan Citra Digital [28]

Gambar 2.2 merupakan contoh gambar dari pengolahan citra digital. Proses citra ini memiliki peran untuk dikenalinya suatu bentuk khusus yang diketahui lewat mesin [26]. Pemrosesan citra diguunakan untuk mentrasformasikan citra

menjadi citra lain, seperti *image compression* menjadi proses *preprocessing* dari computer visi [28].

2.4 Kecerdasan Buatan

Kecerdasan Buatan dapat dijelaskan sebagai sistem atau perangkat yang dirancang untuk meniru kemampuan berpikir manusia dan melaksanakan tugas-tugas secara otomatis, yang sebelumnya memerlukan kecerdasan manusia [29]. Tujuan utama AI adalah untuk mengotomatisasi berbagai proses yang melibatkan keputusan atau analisis, yang sering dilakukan oleh manusia. Penggunaan AI kini telah meluas di berbagai sektor, termasuk pendidikan, kesehatan, ekonomi, dan pertanian [30]. Dengan kemajuan teknologi yang pesat, AI telah memberikan dampak signifikan pada kehidupan sehari-hari, menjadikannya bagian penting dalam berbagai aktivitas manusia. Cara kerja AI melibatkan pengumpulan data dengan cepat, pemrosesan berulang, serta penerapan algoritma yang cerdas untuk menghasilkan keputusan atau solusi yang lebih efisien dan akurat. Keberadaan AI semakin meningkatkan efisiensi dan produktivitas di berbagai bidang, mengubah cara kerja dan berinteraksi dalam kehidupan sehari-hari. [31]. Berikut Gambar 2.3 merupakan pola kecerdasan buatan.



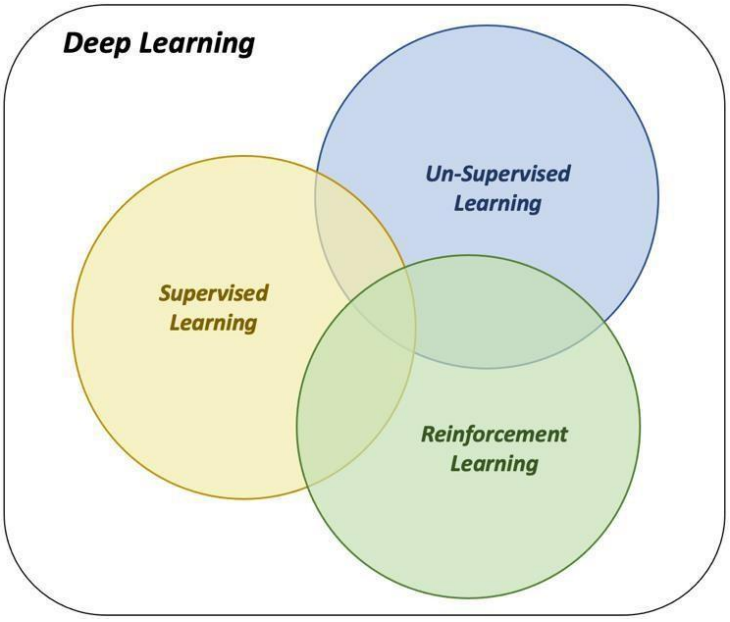
Gambar 2.3 Pola Kecerdasan Buatan [29]

Kecerdasan Buatan (AI) adalah bidang ilmu dan teknik yang bertujuan untuk mengembangkan mesin yang memiliki kemampuan cerdas, terutama dalam membuat program atau aplikasi komputer yang pintar. AI merupakan upaya untuk

menciptakan komputer, robot, atau aplikasi yang dapat beroperasi secara cerdas, mirip seperti cara kerja manusia [32].

2.5 *Deep Learning*

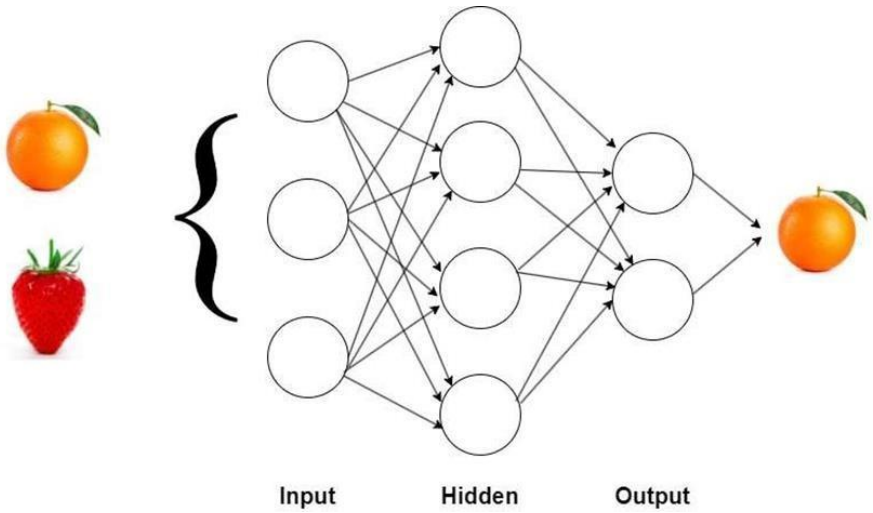
Deep learning (DL) merupakan bagian dari Machine Learning yang terinspirasi oleh struktur korteks otak manusia dengan memanfaatkan jaringan syaraf buatan yang memiliki banyak lapisan tersembunyi. Metode ini merupakan pengembangan dari *Convolutional Neural Networks* (CNN) yang memperbaiki teknik sebelumnya. Dengan model Deep Learning, sejumlah parameter yang tidak perlu dapat dikurangi, Selain itu deformasi gambar input seperti translasi, rotasi, dan skala dapat diatasi [29]. DL memanfaatkan beberapa lapisan di antara input layer dan output layer, yang memungkinkan pemrosesan non-linear melalui beberapa tahap.Hasil dari proses ini dapat digunakan untuk pembelajaran fitur (feature learning) dan klasifikasi pola (pattern classification) [30]. Gambar 2.4 Merupakan Kategori tiga kategori yang ada pada *Deep Learning.*



Gambar 2.4 Kategori Dalam *Deep learning* [30]

Deep learning (DL) merupakan cabang dari Machine Learning yang menggunakan beberapa lapisan pemrosesan non-linear untuk ekstraksi fitur dan transformasi data. Setiap lapisan menggunakan keluaran dari lapisan sebelumnya sebagai input. DL dapat diterapkan pada algoritma supervised atau unsupervised,

yang memungkinkan untuk digunakan dalam analisis pola tanpa pengawasan (unsupervised) dan klasifikasi dengan pengawasan (supervised). Proses ini memanfaatkan struktur bertingkat yang memungkinkan model untuk mengenali pola lebih kompleks dan melakukan prediksi dengan akurasi yang tinggi pada berbagai aplikasi, termasuk pengolahan citra dan analisis data [30]. Berikut Gambar 2.5 merupakan tahapan pada *Deep Learning*.



Gambar 2.5 Tahapan *Deep learning* [28]

Deep learning sangat efektif dalam tugas visi komputer, terutama dalam klasifikasi objek pada citra. Salah satu metode yang populer untuk klasifikasi ini adalah *Convolutional Neural Networks* (CNN) [31]. Perkembangan *Deep learning* saat ini telah didorong oleh banyaknya library dan API yang tersedia. *TensorFlow*, misalnya, adalah sebuah library yang memungkinkan ekspresi algoritma pembelajaran mesin dan eksekusi perintah dengan memanfaatkan informasi tentang objek yang ada, sehingga dapat membedakan dan mengklasifikasikan objek-objek tersebut. TensorFlow juga mendukung pelatihan model menggunakan *Central Processing Unit* (CPU) dan *Graphic Processing Unit* (GPU) [32].

2.6 *Convolutional Neural Network* (CNN)

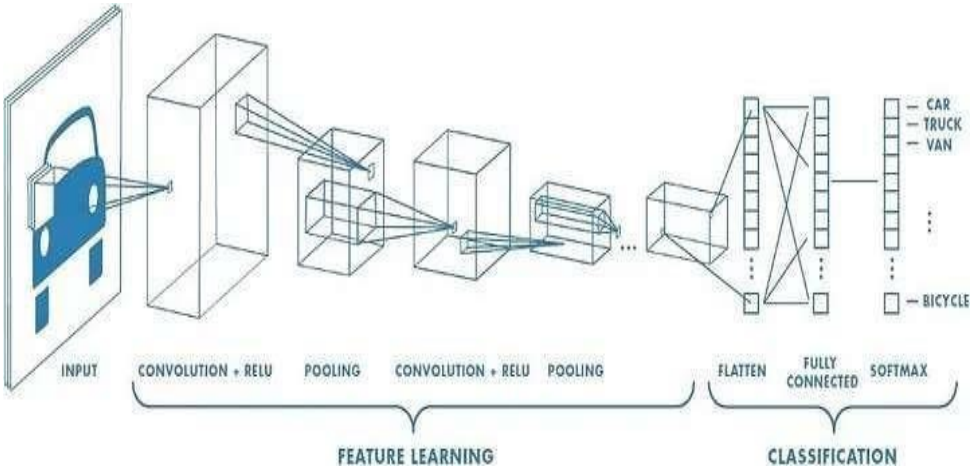
Deep learning telah memperkenalkan berbagai arsitektur yang dapat diterapkan dalam berbagai kasus, salah satunya adalah *Convolutional Neural Networks* (CNN) [33]. CNN pertama kali dikembangkan oleh Kunihiko Fukushima,

seorang peneliti dari NHK Broadcasting Science Research Laboratories di Tokyo, Jepang, dengan nama *Neo Cognitron*. CNN adalah pengembangan dari *Multilayer Perceptron* (MLP) yang dirancang khusus untuk memproses data dua dimensi, seperti gambar, dan lebih efisien dalam mengenali pola-pola kompleks [34]. CNN adalah salah satu jenis Deep Learning yang paling populer dan telah mencapai terobosan besar dalam dekade terakhir di berbagai bidang, mulai dari pengenalan pola dalam pemrosesan gambar, deteksi objek, hingga pengenalan wajah [33]. CNN diimplementasikan sebagai jaringan multi-layer yang dilatih untuk melakukan keputusan klasifikasi. Pendekatan-pendekatan dibangun untuk menjalankan tugas-tugas berbeda secara simultan seperti ekstraksi fitur, reduksi dimensi data, dan klasifikasi. Selain itu, *Convolutional* berfokus pada memisahkan dan mendeteksi pola-pola dalam gambar [35].

Convolutional Neural Networks (CNN) adalah jenis algoritma *deep learning* yang banyak digunakan dalam pemrosesan citra, penglihatan komputer, dan pengenalan pola. CNN memiliki struktur hierarkis yang terdiri dari beberapa lapisan, masing-masing melakukan transformasi tertentu. Lapisan pertama, yaitu lapisan konvolusional, berfungsi untuk mengekstraksi fitur dari gambar input dengan menerapkan filter kecil atau kernel. Kernel ini membantu mendeteksi fitur dasar seperti tepi, tekstur, dan bentuk. Melalui operasi konvolusi, CNN dapat mempelajari hubungan spasial antar piksel yang penting untuk memahami konten gambar. Setelah lapisan konvolusional, lapisan pooling digunakan untuk mengurangi dimensi spasial gambar, sehingga meningkatkan efisiensi komputasi tanpa kehilangan fitur penting. Selanjutnya, ada lapisan fully connected yang meratakan fitur yang telah dipooled dan menghubungkannya ke lapisan output akhir. Lapisan ini membantu model untuk membuat keputusan berdasarkan fitur yang diekstraksi, sering kali digunakan dalam tugas klasifikasi.

Keunggulan CNN terletak pada kemampuannya untuk secara otomatis mempelajari fitur hierarkis tanpa memerlukan ekstraksi fitur manual. Lapisan-lapisan pada CNN dilatih menggunakan backpropagation, di mana jaringan menyesuaikan bobotnya untuk meminimalkan kesalahan prediksi. CNN sangat efektif untuk tugas seperti pengenalan gambar, deteksi objek, dan segmentasi. Selain itu, CNN dapat menangani *Dataset* besar dan struktur gambar yang

kompleks, menjadikannya sangat penting di berbagai bidang seperti pencitraan medis, mobil otonom, dan pengenalan wajah. Dengan adanya framework *deep learning* seperti TensorFlow dan PyTorch, implementasi dan penerapan CNN menjadi lebih mudah dan praktis dalam aplikasi dunia nyata [39]. Gambar 2.6 merupakan Tahapan pada CNN.



Gambar 2.6 Tahapan CNN [39]

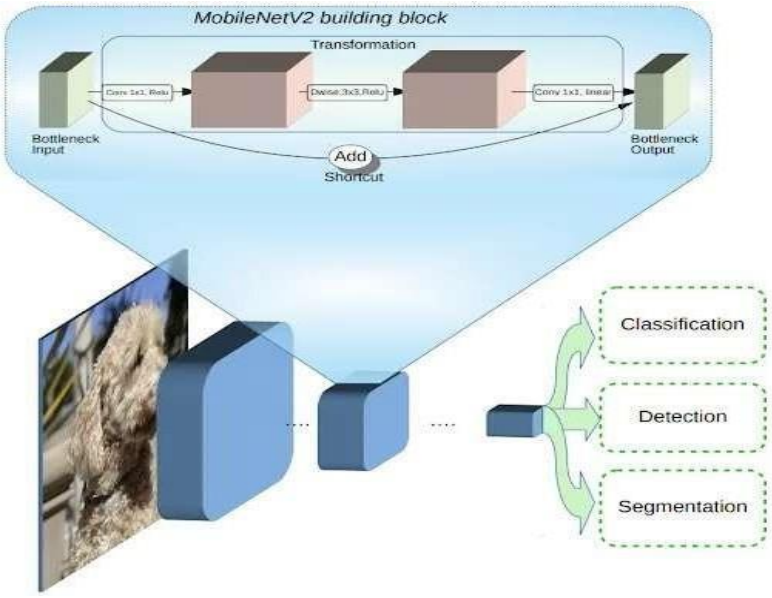
CNN berfungsi untuk membuat *feature map* dari input gambar sehingga gambar lebih dapat dikenali oleh computer [40]. CNN terdiri dari beberapa lapisan yang berbeda, termasuk lapisan input data, lapisan konvolusi, lapisan aktivasi *ReLU*, lapisan penyatuan, dan lapisan yang terhubung sepenuhnya. Lapisan-lapisan yang disebutkan di atas digunakan untuk mencapai ekstraksi fitur dan klasifikasi secara efektif [41] .

2.7 MobileNet V2

MobileNetV2 adalah salah satu model *deep learning* terbaru yang sangat efisien, dirancang untuk mengatasi masalah penggunaan sumber daya komputasi yang besar tanpa mengorbankan akurasi. Berbeda dengan arsitektur CNN tradisional, MobileNetV2 menggunakan dua jenis konvolusi, yaitu depthwise convolution dan pointwise convolution. Penggunaan keduanya membuat MobileNetV2 lebih ringan dan lebih cepat dalam proses komputasi, memungkinkan model ini bekerja lebih baik pada perangkat dengan keterbatasan sumber daya, seperti perangkat mobile atau edge devices. Beberapa penelitian menunjukkan

bahwa MobileNetV2 dapat memberikan hasil klasifikasi gambar dengan tingkat akurasi yang sangat baik meskipun menggunakan lebih sedikit sumber daya dibandingkan dengan model CNN lainnya [42][43][44]. Keunggulan ini menjadikan MobileNetV2 sebagai pilihan utama dalam berbagai aplikasi pengolahan citra, terutama yang memerlukan efisiensi tinggi [45]. *Mobilenet*V2 memiliki modul modul lapisan baru, yaitu *inverted residual* dengan *linear bottleneck*, yang secara signifikan mengurangi memori yang dibutuhkan untuk pemrosesan [21].

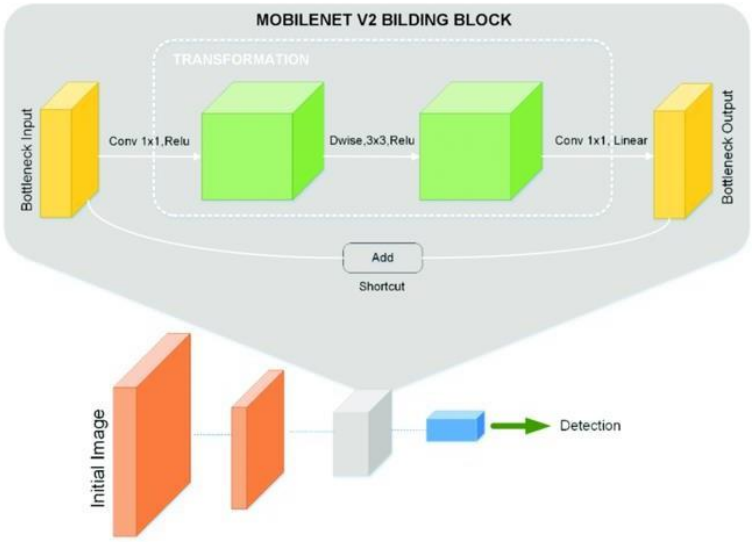
MobileNet adalah arsitektur CNN yang dirancang untuk mengatasi tantangan penggunaan sumber daya komputasi yang besar. Model ini telah dilatih menggunakan *Dataset* ImageNet untuk tugas deteksi dan klasifikasi gambar, dan diimplementasikan dengan menggunakan library TensorFlow [42]. Perbedaan utama antara MobileNet dan arsitektur CNN lainnya adalah penerapan lapisan konvolusi yang menggunakan filter yang lebih tipis, disesuaikan dengan ketebalan gambar input. MobileNet membagi konvolusi menjadi dua jenis, yaitu depthwise convolution dan pointwise convolution, yang memungkinkan model ini untuk bekerja lebih efisien tanpa mengorbankan kinerja [42]. Gambar 2.7 merupakan Tahapan pada *MobileNetV2.*



Gambar 2.7 Tahapan MobileNetV2 [44]

MobileNet adalah salah satu model yang diuji di TensorFlow untuk mempercepat penelitian di berbagai bidang. Model ini terus berkembang sejak

Inception v1 hingga Inception v3, dengan perbaikan pada struktur jaringan. Dalam penelitian ini, MobileNet terbukti efektif dalam klasifikasi gambar [44]. MobileNetV2, khususnya, memiliki kemampuan untuk mempelajari fitur yang lebih kompleks karena menggunakan lebih banyak lapisan konvolusi dengan ukuran filter yang lebih kecil, dibandingkan dengan model sebelumnya seperti AlexNet atau GoogleNet, yang memanfaatkan filter yang lebih besar untuk proses pembelajaran. Model ini lebih efisien dalam penggunaan sumber daya komputasi. *Mobilenet* dioptimalkan untuk menjadi kecil dan efisien [41]. Gambar 2.8 merupakan arsitektur *MobileNetV2*.



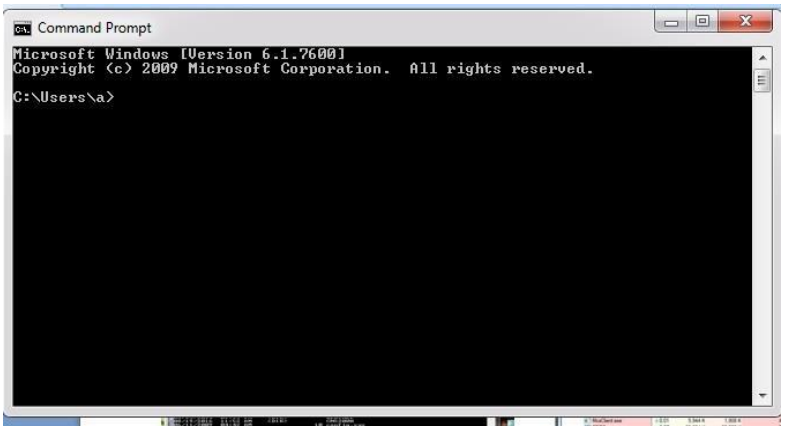
Gambar 2.8 Arsitektur MobileNetV2 [41]

Struktur utama *MobilenetV2* didasarkan pada versi sebelumnya, yaitu *MobilenetV1* dengan menambahkan *inverted residual* dengan modul *bottleneck linier* [41]. Arsitektur MobileNet didasarkan pada konvolusi depthwise yang dapat dipisahkan. Konvolusi 2D standar memproses semua saluran input secara langsung untuk menghasilkan satu saluran output dengan melakukan konvolusi dalam dimensi kedalaman saluran. Sedangkan pada konvolusi kedalaman, gambar input dan filter dipisahkan ke dalam saluran yang berbeda, kemudian setiap saluran input

dikonvolusi dengan saluran filter yang sesuai. Setelah saluran output yang telah difilter dihasilkan, saluran-saluran tersebut kemudian digabungkan kembali [21].

2.8 *Command Prompt*

Command Prompt, atau yang dikenal dengan nama cmd.exe, adalah aplikasi baris perintah yang digunakan pada sistem operasi Windows untuk menjalankan perintah secara langsung. Fungsinya sangat penting untuk mengotomatiskan berbagai tugas menggunakan skrip atau file batch, serta untuk menjalankan fungsi administratif dan pemecahan masalah. Di Windows, Command Prompt sering kali disebut sebagai Windows Command Processor dan memfasilitasi pengguna untuk mengakses berbagai perintah yang tidak tersedia dalam antarmuka grafis. Selain itu, *Command Prompt* juga membantu dalam memecahkan masalah teknis yang sering terjadi di sistem. Pada sistem operasi Linux, peran ini dipegang oleh pengguna dengan hak akses root yang memberikan kontrol penuh. Tampilan awal *Command Prompt* dapat diakses melalui aplikasi di menu Start, dan pengguna dapat memasukkan perintah untuk melakukan berbagai operasi sistem yang diperlukan.



Gambar 2.9 Tampilan *Command Prompt* [46]

Command prompt dapat membaca sekumpulan instruksi dari sebuah file. Dalam hal ini shell cmd bertindak sebagai penerjemah bahasa dan isi file dapat dianggap sebagai program yang sebenarnya. Ketika mengeksekusi program batch ini, tidak ada fase kompilasi perantara. Program-program ini biasanya dibaca, ditafsirkan dan dieksekusi baris demi baris. Karena tidak ada kompilasi, tidak ada produksi file yang dapat dieksekusi terpisah. Karena alasan ini, program-program

ini disebut *skrip batch* atau *skrip shell* [42]. *Command prompt* merupakan salah satu cara untuk mengakses dan mengontrol berbagai aspek dari sistem operasi *Windows* tanpa menggunakan antarmuka grafis pengguna (GUI) [44]. *Command prompt* adalah antarmuka baris perintah yang dapat digunakan untuk menjalankan berbagai perintah dan skrip, termasuk yang terkait dengan *machine learning*.

2.9 Efektivitas Sistem

Performansi subset fitur didapati lewat penilaian *accuracy*, *precision*, *r e c a l l* serta *F1-Score*. Ketiga metrik evaluasi ini dihitung dengan *confusion matrix* pada Tabel 2.1.

Tabel 2.1 *Conffusion Matrix*

Kelas Sebenarnya	**Hasil klasifikasi**	
Positif	TP	FN
Negatif	FP	TN

Tabel 2.1 merupakan tampilan dari *coffusion matrix* yang dimana memilki keterangan *True Possitive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN) didapati lewat tabel *confusion matrix.* TN ialah banyaknya lulus dengan tepat serta berhasilnya diprediksi dengan tepat waktu. TP ialah total lulus tidak tepat yang berhasil diperkirakan lulus tidak tepat. FP yaitu banyaknya lulus tepat waktu yang diperkirakan sebaliknya. Sedangkan FN adalah lulus tidak tepat waktu yang sudah dideteksi sebaliknya, Tabel 2.1 menunjukkan bagian yang ada pada *coffusion matrix* dengan pembagian pada hasil klasifikasi dan kelas sebenarnya, *Specificity* atau *presisi* ialah ketrampilan jenis saat mengenali semua informasi lulus tidak tepat waktu dengan benar. *Accuracy* menyebutkan bahwa ketrampilan model saat menjalankan deteksi lulusan tepat serta tidak tepat waktu dengan baik. dan persamaan yang digunakan untuk menentukan nilai efektivitas sistem yaitu Persamaan (2.1) yang merupakan persamaan untuk mencari nilai *accuracy*.

$$\text{Accuracy} = \frac{total\ TP}{total\ semua\ data} \tag{2.1}$$

Berdasarkan Persamaan (2.1) *Accuracy* mengatakan ketrampilan model saat deteksi lulusan tepat serta tidak tepat waktu, dan untuk mencari nilai *precision* memakai Persamaan (2.2).

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

Berdasarkan Persamaan (2.2) *Specificity* atau *presisi* yaitu ketrampilan mengetahui semua informasi lulus tidak tepat waktu dengan benar, untuk mencari nilai sensitivity atau recall memakai persamaan (2.3).

$$Recall = \frac{TP}{TP + FN} \quad (2.3)$$

Berdasarkan Persamaan (2.3) *sensitivity* atau *recall* dirincikan dengan tingkat kemampuan model untuk mengetahui semua informasi lulus tepat waktu dengan tepat, untuk mencari nilai *F1-score* menggunakan Persamaan (2.4).

$$F1\text{-}score = 2X\frac{Precision \cdot Recall}{Precission + Recall} \quad (2.4)$$

Berdasarkan Persamaan (2.4) *F1-score* yaitu metrik evaluasi yang digunakan dalam klasifikasi untuk mengukur keseimbangan antara *precision* dan *recall*. Nilai *F1-score* berkisar antara 0 hingga 1, hasil 1 menunjukkan hasil yang sempurna dan 0 menunjukkan hasil yang sangat buruk.

2.10 Kajian Pustaka

Penelitian ini mengajukan metode berbasis Convolutional Neural Network (CNN) untuk mendeteksi penyakit mata melalui citra fundus. Pendekatan yang digunakan mengadopsi transfer learning dengan dua komponen utama: base model dan head model. MobilenetV2 dipilih sebagai base model berdasarkan hasil eksperimen karena menunjukkan efisiensi tinggi dan akurasi yang unggul dibandingkan model CNN lain seperti ResNet50V2, InceptionV3, InceptionResNetV2, VGG16, dan VGG19. Uji coba dilakukan pada *Dataset* citra fundus yang berisi 601 gambar dengan klasifikasi ke dalam dua kategori, yaitu "Normal" dan "Abnormal". Hasil menunjukkan bahwa metode ini memberikan kinerja optimal meskipun *Dataset* relatif kecil, dengan akurasi mencapai 72% [26]. Penelitian ini menguji arsitektur CNN seperti ResNet 101, AlexNet, dan VGG-16-19 untuk mendeteksi parasit penyebab malaria. Hasilnya menunjukkan bahwa ResNet 101 unggul dengan akurasi 95,92%, membuktikan efektivitasnya dalam klasifikasi penyakit berbasis citra medis [21].

Penelitian ini bertujuan untuk mendeteksi malaria menggunakan algoritma pre-trained models seperti EfficientNet B0 dan MobileNetV2. Berdasarkan hasil pengujian, kedua algoritma ini dapat diandalkan untuk prediksi malaria. Namun, MobileNetV2 menunjukkan performa yang unggul dengan rata-rata akurasi mencapai 97,27%. Algoritma ini memiliki keunggulan dalam efisiensi pemrosesan

dan akurasi prediksi yang tinggi, menjadikannya pilihan yang lebih baik dibandingkan metode lainnya. Penelitian ini menegaskan potensi penggunaan MobileNetV2 dalam diagnosis malaria secara akurat dan efisien [41].

Penelitian ini bertujuan mendeteksi penyakit pada otak dengan memanfaatkan arsitektur CNN, khususnya MobileNetV2. klasifikasi menggunakan pemrosesan gambar tradisional dilakukan secara bertahap hingga akhirnya mengadopsi arsitektur MobileNetV2 yang dikenal memiliki akurasi tinggi. Arsitektur ini unggul dalam efisiensi dengan ukuran model dan jumlah parameter pelatihan yang kecil dan memberikan performa yang baik. Tahapan penelitian meliputi pengumpulan data, pemrosesan awal data (data preprocessing), klasifikasi, evaluasi, hingga implementasi. MobileNetV2 menunjukkan kemampuannya yang optimal dalam klasifikasi jenis tumor otak dengan menghasilkan akurasi sebesar 88,64%. Hal ini membuktikan keunggulan utama MobileNetV2 mencakup efisiensi komputasi dan akurasi tinggi meskipun menggunakan model yang relatif ringan. Dengan pendekatan ini, penelitian memberikan kontribusi signifikan terhadap penerapan teknologi berbasis CNN dalam bidang kesehatan, khususnya untuk mendeteksi penyakit otak dengan tingkat akurasi yang sangat memadai [50].

Penelitian ini melakukan klasifikasi penyakit pada daun teh dengan menggunakan gabungan tiga arsitektur CNN, yaitu GoogleNet, Xception, dan Inception ResNet V2. Ketiga arsitektur tersebut memanfaatkan *Dataset* yang sama untuk pengujian dan validasi, dengan menerapkan fungsi aktivasi ReLU serta Batch Size sebesar 10. *Dataset* yang digunakan mencakup 4.272 citra, yang dibagi dalam proporsi data training sebesar 80%, testing 10%, dan validasi 10%. Pendekatan ini berhasil menghasilkan akurasi akhir sebesar 89,64%. Penelitian ini menunjukkan efektivitas kombinasi arsitektur CNN dalam meningkatkan performa klasifikasi, khususnya pada penyakit daun teh, dengan memanfaatkan metode pengolahan data dan algoritma berbasis *deep learning* [43].

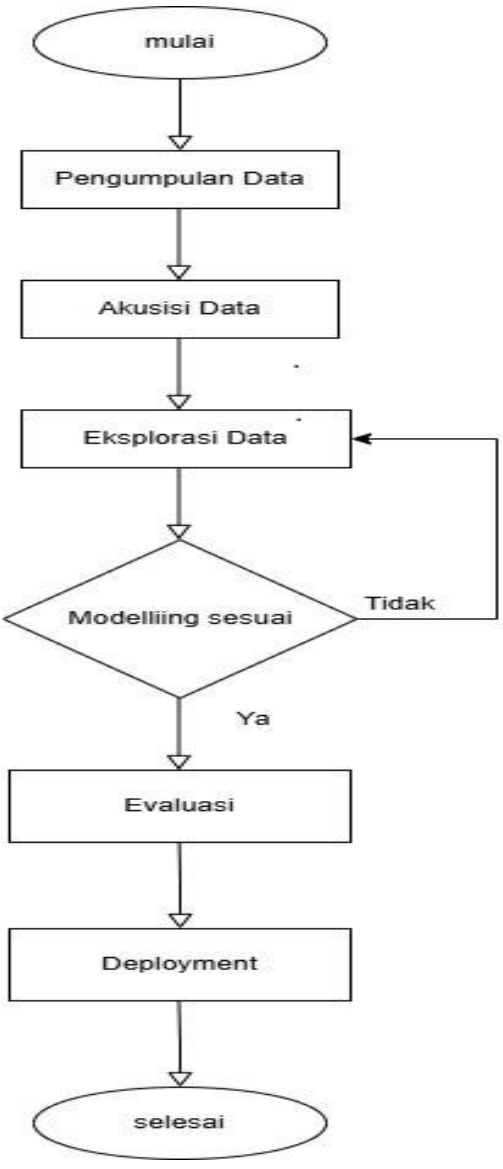
BAB III
METODOLOGI PENELITIAN

Penelitian ini mengaplikasikan metode pengembangan Deep Learning dengan menggunakan CNN dan arsitektur MobileNetV2. Terdapat beberapa langkah yang harus dilakukan untuk mencapai hasil yang diinginkan, di antaranya sebagai berikut. Studi literatur dilakukan bertujuan untuk mencari informasi mengenai *Deep learning* dengan metode CNN dengan menggunakan algoritma *MobileNetV2.* Pada tahapan ini dimulai dari pengenalan dan analisis permasalahan yang terjadi pada kehidupan sehari-hari atau biasa disebut dengan *problem scoping.* Setelah itu tahap selanjutnya yaitu pengambilan *Dataset* atau *data acquisition*. Jika *Dataset*nya sudah terkumpul, maka langkah selanjutnya yaitu tahap *pre-processing* data untuk pengolahan data yang akan digunakan.

Data processing adalah operasi pada data untuk mengelompokkan, mengambil, dan mengubahnya, sedangkan data *exploration* merupakan proses investigasi untuk menentukan pola atau tren, dan mengidentifikasi hubungan yang mungkin ada diantara variabel. Pada proses ini, data akan diolah secara keseluruhan dengan berbagai macam model AI. Hasil dari proses ini merupakan informasi-informasi yang digunakan dalam merepresentasikan hubungan antar data. Tahap selanjutnya adalah data training dan data testing, skala yang digunakan adalah 80:20 dimana 80% untuk *data train* dan 20% untuk data *test. Training* data adalah kumpulan data awal yang digunakan untuk melatih suatu model dalam mengimplementasikan teknologi dan menghasilkan keluaran yang baik. Setelah melalui tahapan training dan testing, model akan dievaluasi dengan berbagai metrik kinerja, seperti akurasi, presisi, recall, dan F1-score menggunakan *Confussion matrix. Confussion matrix* memberikan gambaran yang lebih jelas tentang bagaimana model bekerja, terutama ketika ada ketidakseimbangan kelas dalam *Dataset*. Hal ini bertujuan untuk mencari model mana yang memberikan hasil evaluasi terbaik.

3.1 Alur Penelitian

Sebagai bagian penting dari penelitian ini, penulis telah merancang metodologi yang sistematis untuk memastikan bahwa setiap tahap dari proses penelitian dapat dilaksanakan dengan baik. Diagram alir berikut ini akan memberikan gambaran yang jelas tentang langkah-langkah yang akan diambil dalam penelitian ini. Dengan visualisasi ini, penulis berharap dapat memperjelas alur kerja dan memudahkan pemahaman mengenai bagaimana penulis akan mencapai tujuan penelitian yang telah ditetapkan. berikut diagram alir pada Gambar 3.1 dari sistematika projek ini:



Gambar 3.1 Diagram Alir Penelitian

Gambar 3.1 merupakan proses pembuatan model *Deep Learning* untuk *Citrus Disease Detector* secara umum. Implementasi pemrograman pada *Deep Learning* yang menggunakan metode *Convolutional Neural Network* (CNN) dengan arsitektur Mo,bileNetV2 dapat menjadi jauh lebih kompleks dan bervariasi tergantung pada kebutuhan dan tujuan spesifik model yang akan dibangun. Langkah-langkah yang diuraikan sebelumnya mencerminkan tahapan umum dalam pengembangan model *Deep Learning* berbasis CNN, meskipun penyesuaian tertentu sering kali diperlukan berdasarkan tugas yang dihadapi.

Metode CNN pada *Deep Learning* dirancang untuk menangkap pola lokal dan ketergantungan spasial dalam data visual. Hal ini membuat CNN sangat efektif dalam analisis citra, terutama untuk klasifikasi dan deteksi objek. CNN bekerja dengan memanfaatkan lapisan konvolusi untuk mengekstraksi fitur dari data input, yang memungkinkan jaringan mengenali pola-pola penting dalam citra seperti tepi, tekstur, dan bentuk. MobileNetV2, sebagai salah satu arsitektur CNN, dirancang untuk meningkatkan efisiensi komputasi tanpa mengorbankan akurasi, menjadikannya pilihan populer untuk aplikasi pada perangkat dengan sumber daya terbatas.

CNN terinspirasi oleh cara kerja korteks visual di otak manusia, yang bertugas memproses informasi visual. Korteks visual mampu mengenali pola visual melalui hierarki yang dimulai dari pola sederhana seperti garis hingga pola kompleks seperti objek. Pendekatan ini diadaptasi dalam CNN melalui susunan lapisan konvolusi, *pooling*, dan lapisan *fully connected*, yang semuanya berkontribusi pada kemampuan model untuk memahami dan mengklasifikasikan data visual. Dalam konteks *Citrus Disease Detector*, CNN bertujuan untuk mengenali pola visual spesifik yang menunjukkan keberadaan penyakit pada tanaman jeruk. MobileNetV2 digunakan karena arsitekturnya yang ringan dan efisien, memungkinkan implementasi yang optimal meskipun pada perangkat dengan keterbatasan daya komputasi.

3.2 Komponen Penelitian

Penelitian deteksi penyakit pada buah jeruk menggunakan metode algoritma *Convolutional Neural Network* (CNN) dengan arsitektur MobileNetV2

memerlukan sejumlah komponen yang mendukung kelancaran dan efektivitas pelaksanaannya. Pemilihan komponen yang tepat menjadi sangat penting untuk memastikan hasil yang optimal. Dalam penelitian ini, komponen utama yang digunakan terdiri dari perangkat keras dan perangkat lunak. Pada aspek perangkat keras, penelitian ini menggunakan komputer atau server dengan spesifikasi tinggi yang dilengkapi dengan unit pemrosesan grafis (*GPU*). Komponen ini dipilih karena kemampuan GPU dalam menangani komputasi intensif yang dibutuhkan dalam pelatihan model CNN, terutama dengan arsitektur seperti MobileNetV2 yang dirancang untuk efisiensi tinggi. Selain itu, perangkat keras seperti kamera atau alat pengambilan gambar berkualitas tinggi juga digunakan untuk mendapatkan citra buah jeruk yang akurat. Setiap komponen dipilih berdasarkan kriteria spesifik, seperti kompatibilitas, kinerja, dan kemampuan mendukung kebutuhan penelitian. Kombinasi perangkat keras dan perangkat lunak yang sesuai memastikan penelitian dapat berjalan dengan optimal, menghasilkan model yang akurat dan efisien untuk mendeteksi penyakit pada buah jeruk. Tabel 3.1 merupakan Perangkat keras yang digunakan saat penelitian berlangsung.

Tabel 3.1 Spesifikasi Laptop Yang Digunakan

Spesifikasi Laptop	*Thosiba sattelite C840*
Processor	*Intel*(R) *Core*(TM) i3-238M CPU @ 2.20GHz
RAM	2.00 GB
ROM	HDD 1 TB
System	64-*Bit Operating System*

Tabel 3.1 spesifikasi laptop yang digunakan sebagai perangkat utama dalam melakukan penelitian ini, dimana perangkat laptop yang digunakan yaitu laptop *Thosiba sattelite C840* yang bertujun untuk menjalankan program yang akan dirancang menggunakan *google colab.* Tabel 3.2 merupakan perangkat lunak yang digunakan pada saat penelitian berlangsung.

Tabel 3.2 Perangkat Lunak

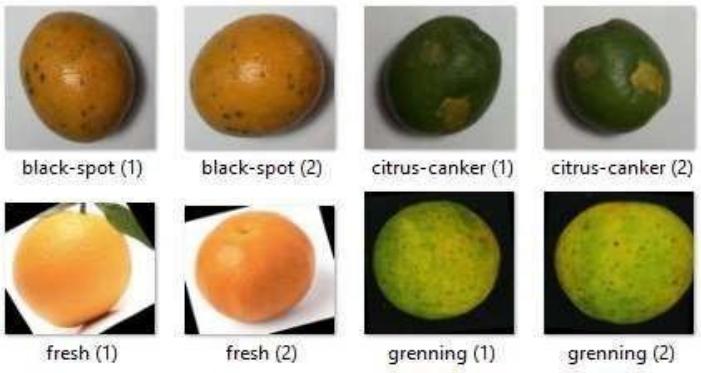
Google Colab	layanan berbasis *cloud* yang disediakan oleh Google untuk menjalankan kode *Python*. *Colab* mendukung berbagai pustaka *Deep learning* seperti *TensorFlow, Keras, dan PyTorch*, sehingga dapat dengan mudah membangun dan melatih model CNN. Dengan integrasi *Google Drive, google colab* digunakan untuk menyimpan *Dataset* dan model yang telah dilatih.
Command Prompt	*Command Prompt* digunakan untuk menjalankan berbagai perintah atau *input*. Sebagian besar perintah ini dirancang untuk mengotomatisasi tugas melalui skrip dan *file batch*, memungkinkan fungsi administratif seperti konfigurasi sistem, serta membantu dalam memecahkan masalah dan menyelesaikan beberapa jenis *error* yang umum terjadi pada sistem operasi *Windows.*
Pemrograman *phyton*	Bahasa Pemrograman Menjadi Jembatan Untuk Pengaplikasian *Deep Learning. Phyton* menjadi bahasa pemrograman yang memiliki Pendekatan yang fokus pada kesederhanaan dan kejelasan sintaksis Pada Bidang Kecerdasan Buatan. *phyton* sangat popular dilengkapi dengan pustaka-pustaka yang disediakan seperti *Numpy, Pandas, TensorFlow* dll. *TensorFlow* sering digunakan karena pustaka yang paling mudah dikembangkan dan memiliki komputasi tingkat tinggi. Selain itu, *TensorFlow* memberikan dukungan untuk berbagai jenis model pembelajaran termasuk *Convolutional Neural Network*

Tabel 3.2 di atas merupakan perangkat lunak yang digunakan dalam penelitian ini. Setiap perangkat lunak dipilih berdasarkan kemampuannya untuk mendukung proses pengembangan dan analisis data yang efisien. Penggunaan perangkat lunak ini secara sinergis mendukung kelancaran dan keberhasilan penelitian, memungkinkan penulis untuk mencapai hasil yang optimal dalam pengembangan model AI.

3.3 Pengumpulan Data

Proses pengumpulan data merupakan tahap penting dalam penelitian ini untuk memastikan keandalan dan validitas hasil. Data yang digunakan terdiri dari 2.400 sampel citra yang dibagi ke dalam dua kategori utama, yaitu data pelatihan (*training data*) dan data pengujian (*testing data*), dengan proporsi pembagian 80% untuk pelatihan dan 20% untuk pengujian. Proporsi ini dipilih untuk memberikan keseimbangan yang optimal antara pelatihan model dan evaluasi akurasi prediksi.

Dataset yang digunakan dalam penelitian ini diperoleh dari Kaggle, sebuah platform yang terkenal sebagai komunitas global bagi data scientist dan peneliti. Kaggle menyediakan berbagai dataset berkualitas tinggi untuk mendukung analisis data dan pengembangan model berbasis pembelajaran mesin. Dalam konteks penelitian ini, dataset dari Kaggle mengandung informasi yang relevan untuk klasifikasi penyakit pada buah jeruk. Dataset tersebut mencakup beragam citra yang merepresentasikan kondisi buah jeruk, termasuk kategori normal dan yang terkena penyakit. Data ini diolah untuk mempersiapkan model yang mampu mendeteksi dan mengklasifikasi penyakit dengan efisien. Gambar 3.2 menunjukkan dataset yang digunakan dalam penelitian ini, memberikan visualisasi yang memperlihatkan keragaman citra yang diolah dalam proses pelatihan dan pengujian model.



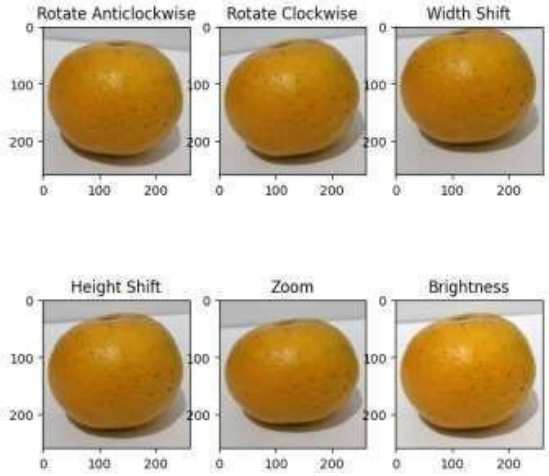
Gambar 3.2 Contoh *Dataset* Citrus

Gambar 3.2 menunjukkan contoh dataset yang digunakan dalam penelitian deteksi penyakit pada buah jeruk. Dataset ini terdiri dari citra yang dikelompokkan ke dalam empat kelas, yaitu *black spot*, *canker*, *greening*, dan *fresh*. Setiap kelas merepresentasikan kondisi spesifik buah jeruk, baik yang sehat (*fresh*) maupun yang terinfeksi penyakit tertentu. Dataset ini disiapkan untuk melatih model dalam

mengenali dan mengklasifikasi jenis penyakit dengan tingkat akurasi yang tinggi. Citra-citra ini menjadi dasar dalam proses pelatihan dan pengujian model berbasis algoritma *Convolutional Neural Networ*k (CNN).

3.4 Pra-pemrosesan data

Tahap pertama dalam pemrosesan data adalah proses augmentasi data. Selain itu, langkah-langkah pra-pemrosesan data juga dilakukan untuk membersihkan dan mempersiapkan data sebelum digunakan dalam analisis atau pemodelan. Gambar 3.3 merupakan contoh augmentasi data.



Gambar 3.3 Augmentasi Data

Gambar 3.3 merupakan proses augmentasi data pada penelitian ini yang bertujuan untuk meningkatkan variasi data, mengurangi overfitting, meningkatkan robustitas model, mengatasi data tidak seimbang, dan mempercepat proses pelatihan. Augmentasi membantu menyediakan lebih banyak contoh untuk pelatihan dan membuat model lebih tahan terhadap gangguan.

Setelah tahap augmentasi data selesai dilakukan, tahap selanjutnya adalah proses load *Dataset*. Pada tahap ini dilakukan proses input *Dataset* serta memisahkan file-filenya, setelah itu dilakukan proses penentuan direktori pelatihan dan pengujian pada dokumen *Dataset* dengan cara labelling dan splitting *Dataset*. Splitting *Dataset* dilakukan dengan proporsi 80:20 dimana 80% untuk data latih dan 20% untuk data uji.. Ukuran dimensi data yang digunakan adalah 224 x 224 dan *batch size* yang digunakan adalah 16.

3.5 *Modeling*

Modelling Citrus yang menggunakan model *MobilenetV2.* Penggunaan model ini memiliki akurasi terbaik dan cocok untuk digunakan dalam *image classification citru*s. *MobileNetV2* adalah model arsitektur CNN yang dikembangkan oleh google. *MobileNetV2* memprioritaskan ukuran dan kebutuhan komputasi yang kecil namun tetap mempertahankan nilai akurasi. *MobileNetV2* sangat tepat digunakan pada perangkat yang memiliki sumber daya yang terbatas.

Penggunaan model *MobilenetV2* Tahap persiapan *Mobilenet*V2 untuk transfer learning Menggunakan base model *Mobilenet*V2 dengan arsitektur sebagai berikut:

1. *Global Average Pooling* 2D untuk mengoperasikan *pooling*
2. *Layers Dense* dengan ukuran file 128 *layers* dan dengan aktivasi relu
3. untuk mencegah adanya overfitting maka ditambahkan Dropout 0.2
4. *Layer Dense* 1 dengan ukuran file 64 layers dan dilengkapi aktivasi relu
5. Kemudian dilakukan *compile* model dengan menggunakan *loss binary crossentropy*, *optimizer adam,* dan *metrics accuracy.*

3.6 Evaluasi

Evaluasi model ini sangat penting untuk dilakukan guna mengukur seberapa baik kinerja yang tercapai. model yang telah dirancang dalam menyelesaikan masalah tertentu. Dalam penelitian ini, evaluasi dilakukan dengan menggunakan beberapa metrics, seperti precision, recall, F1 score, dan accuracy, serta confusion matrix. Precision mengukur seberapa banyak prediksi positif yang benar dari semua prediksi positif yang dilakukan, sementara recall mengukur seberapa banyak prediksi positif yang benar dari semua data positif yang seharusnya terdeteksi. F1 score adalah rata-rata harmonik dari precision dan recall, yang menggambarkan keseimbangan antara keduanya. Accuracy mengukur seberapa banyak prediksi yang benar dibandingkan dengan total prediksi yang dibuat. Confusion matrix digunakan untuk memberikan gambaran lebih mendalam tentang kinerja model dengan menampilkan jumlah prediksi yang benar dan salah dalam kategori positif dan negatif.

Melalui evaluasi ini, kita dapat mendeteksi masalah seperti overfitting, di mana model terlalu baik pada data pelatihan namun gagal pada data baru, dan underfitting, di mana model gagal menangkap pola yang ada dalam data. Evaluasi juga memberikan umpan balik untuk perbaikan model lebih lanjut. Selain itu, hasil evaluasi membantu dalam pengambilan keputusan yang informatif mengenai model mana yang paling sesuai untuk aplikasi tertentu, sehingga dapat digunakan secara optimal dalam konteks dunia nyata.

3.7 *Deployment*

Deployment bertujuan untuk menghubungkan teknologi dengan kebutuhan pengguna dalam mengakses web deteksi penyakit buah jeruk secara efisien. Pada penelitian ini, deployment dilakukan di local host, yang berarti web dijalankan pada komputer lokal. Untuk mengaksesnya, pengguna perlu menggunakan command prompt untuk memulai server dan mengakses aplikasi melalui browser. Metode ini memungkinkan pengujian dan pengembangan web secara lebih mudah sebelum dipublikasikan secara lebih luas. Dengan deployment di local host, pengguna dapat menguji fungsionalitas dan performa web dalam lingkungan yang terkendali dan lebih terfokus pada perbaikan serta pengoptimalan.

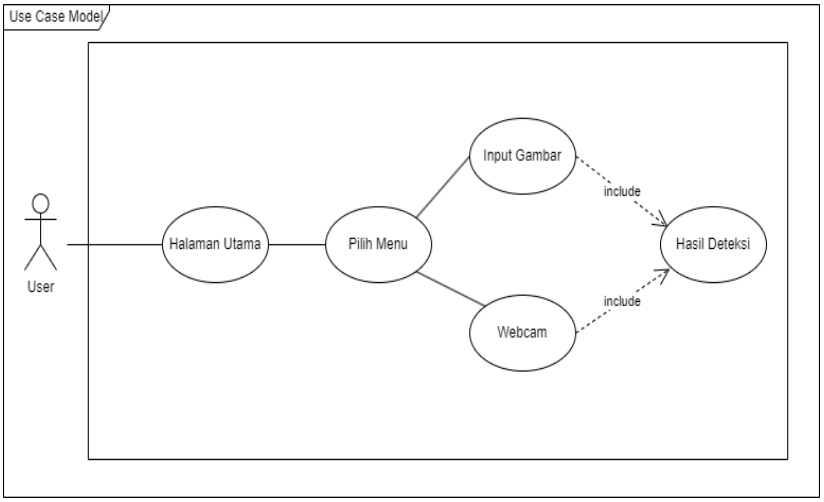
3.8 Perancangan Sistem

Perancangan sistem adalah proses merancang dan mengembangkan struktur, komponen, dan interaksi dalam sebuah sistem untuk memenuhi kebutuhan tertentu. Perancangan sistem yang baik membantu dalam memahami dan mendefinisikan kebutuhan pengguna serta tujuan dari model yang akan dibangun.

Perancangan sistem dilakukan dengan untuk mengembangankan arsitektur yang efisien, memastikan bahwa model dapat diimplementasikan dengan optimal dalam infrastruktur yang tersedia.

3.9.1 *Use Case* Diagram

Use case diagram adalah representasi visual yang menggambarkan interaksi antara aktor atau pengguna dengan sistem yang sedang dikembangkan. Diagram ini merupakan bagian dari Unified Modeling Language (UML) dan digunakan untuk menggambarkan fungsionalitas sistem serta cara aktor berinteraksi dengan berbagai fitur dalam sistem aplikasi web. Gambar 3.4 Merupakan *use case* diagram pada penelitian ini.

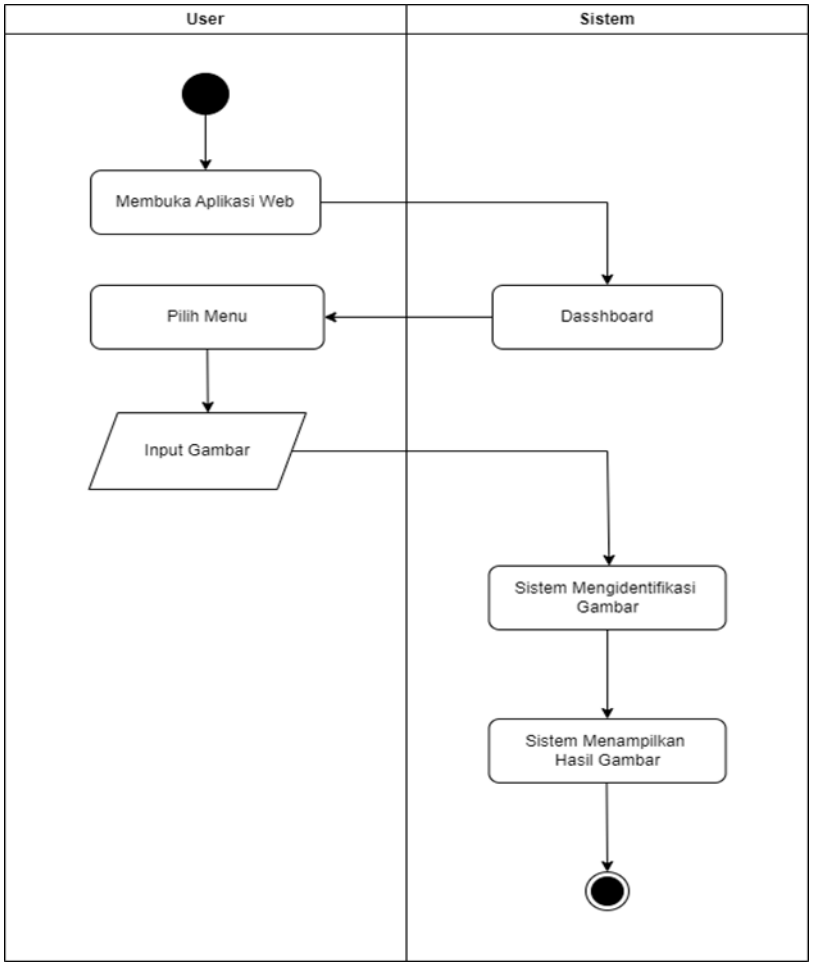


Gambar 3.4 *Use Case Diagram*

Gambar 3.4 diatas merupakan *use case diagram* deteksi penyakit buah jeruk. Diagram diatas menggambarkan interaksi antar user dan sistem aplikasi, dimana *user* dapat menggunakan fitur input gambar atau webcam lalu akan mendapatkan hasil deteksi penyakit pada buah jeruk.

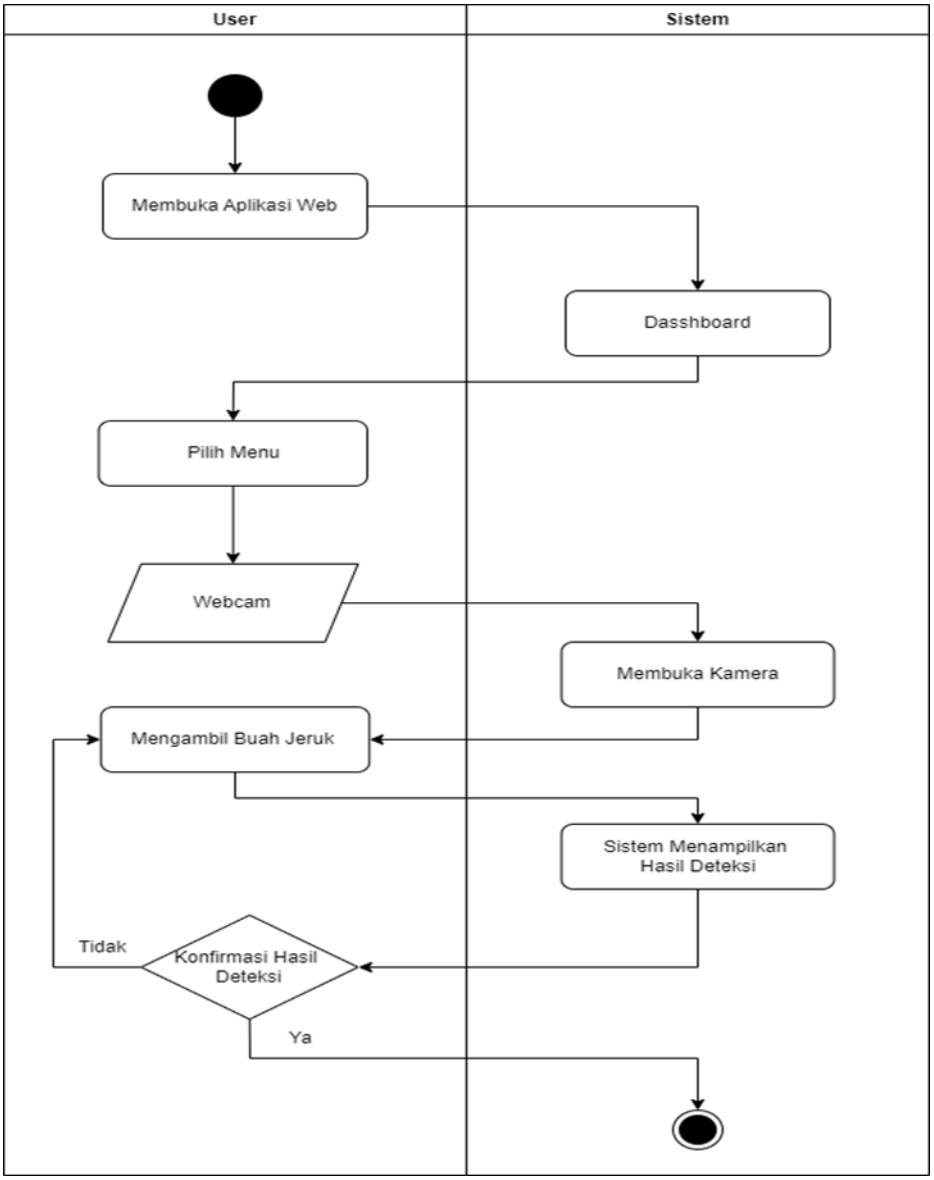
3.9.2 *Activity Diagram*

Activity diagram adalah jenis diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk menggambarkan alur kerja atau proses dalam suatu sistem. Diagram ini menggambarkan urutan langkah-langkah dalam sebuah aktivitas dan bagaimana aktivitas-aktivitas tersebut saling berinteraksi satu sama lain. Gambar 3.5 merupakan *activity diagram* input gambar pada penelitian ini.



Gambar 3.5 *Activity Diagram Input* Gambar

Gambar 3.5 merupakan *activity diagram* input gambar pada deteksi penyakit buah jeruk. Pada diagram tersebut menggambarkan proses *user* yaitu membuka aplikasi web, memilih fitur menu lalu input gambar yang ingin di deteksi. Sedangkan proses pada system yaitu ketika user menginputkan gambar maka sistem akan mengidentifikasi gambar tersebut lalu sistem akan menampilkan hasil deteksi gambar yang telah diinputkan oleh *user.* Pada Gambar 3.6 merupakan *activity diagram* pada fitur *webcam.*



Gambar 3.6 *Activity Diagram Webcam*

Gambar 3.6 diatas menggambarkan bagaimana alur kerja antara *user* dan sistem pada saat menggunakan fitur *webcam* untuk mendeteksi penyakit buah jeruk secara *real time.*

BAB IV
HASIL DAN PEMBAHASAN

4.1 Akusisi Data

Dalam proses pembuatan website "*Citrus disease detector* " jenis data set yang digunakan yaitu kita melakukan pengumpulan data yang diperoleh dari Kaggle, sebuah platform sumber terbuka yang menyediakan berbagai *Dataset* untuk keperluan analisis dan proyek-proyek computer vision Data ini merupakan aset berharga dalam mengembangkan model *Deep learning* untuk mendeteksi penyakit pada tanaman. Pada proyek CDD ini kita menggunakan sebanyak 2400 data. Proses pengambilan data melibatkan pengunduhan *Dataset* dari sumbernya, kemudian data ini diproses, dieksplorasi, dan diolah sebelum digunakan dalam pembuatan model. Data yang berkualitas dan berjumlah besar sangat penting dalam memastikan akurasi dan kinerja model deteksi penyakit tersebut.

Proses augmentasi data ini menggunakan beberapa *function* yaitu sebagai berikut :

1. *clockwise_rotation(img):*

Fungsi ini melakukan rotasi gambar searah jarum jam. Menggunakan `cv2.cvtColor()` untuk mengubah warna gambar, meskipun parameter `0` biasanya digunakan untuk mengonversi gambar ke *grayscale*. Pada fungsi ini Gambar diubah ukurannya menjadi 224x224 piksel kemudian Gambar kemudian diputar berlawanan arah jarum jam (negatif sudut) menggunakan fungsi `rotate()`.

2. *flip_up_down(img):*

Fungsi ini membalik gambar secara vertikal (atas ke bawah) kemudaian gambar diubah ukurannya dan dikonversi warna dan Menggunakan `np.flipud()` untuk membalik gambar.

3. *add_brightness(img):*

Pada fungsi ini menambahkan efek kecerahan pada gambar, Mengonversi dan mengubah ukuran gambar sama seperti sebelumnya, Menggunakan `adjust_gamma()` untuk meningkatkan kecerahan gambar dengan gamma yang lebih kecil dari 1 untuk membuat gambar lebih terang.

4. *blur_image(img):*

Fungsi ini menambahkan efek blur (kabur) pada gambar. Setelah mengonversi dan mengubah ukuran gambar, efek *Gaussian blur* diterapkan menggunakan `cv2.GaussianBlur()` dengan kernel berukuran 9x9.

5. *sheared(img):*

Fungsi ini memberikan efek shear (geser). Pada fungsi ini Mengonversi dan mengubah ukuran gambar menjadi tahap awal, kemudian Menggunakan `AffineTransform()` untuk membuat transformasi shear dan menerapkannya dengan `warp()`.

6. *warp_shift(img):*

Fungsi ini melakukan pergeseran gambar. Setelah konversi dan resizing, fungsi ini menggunakan `AffineTransform()` untuk menerapkan translasi, menggeser gambar ke bawah sebesar 40 piksel, dan menggunakan `warp()` untuk menerapkan transformasi.

Secara keseluruhan, kode yang digunakan pada augementasi data ini mendefinisikan beberapa fungsi untuk melakukan transformasi dan efek pada gambar menggunakan OpenCV dan beberapa pustaka lainnya, sehingga dapat digunakan untuk memproses gambar dalam aplikasi tertentu.

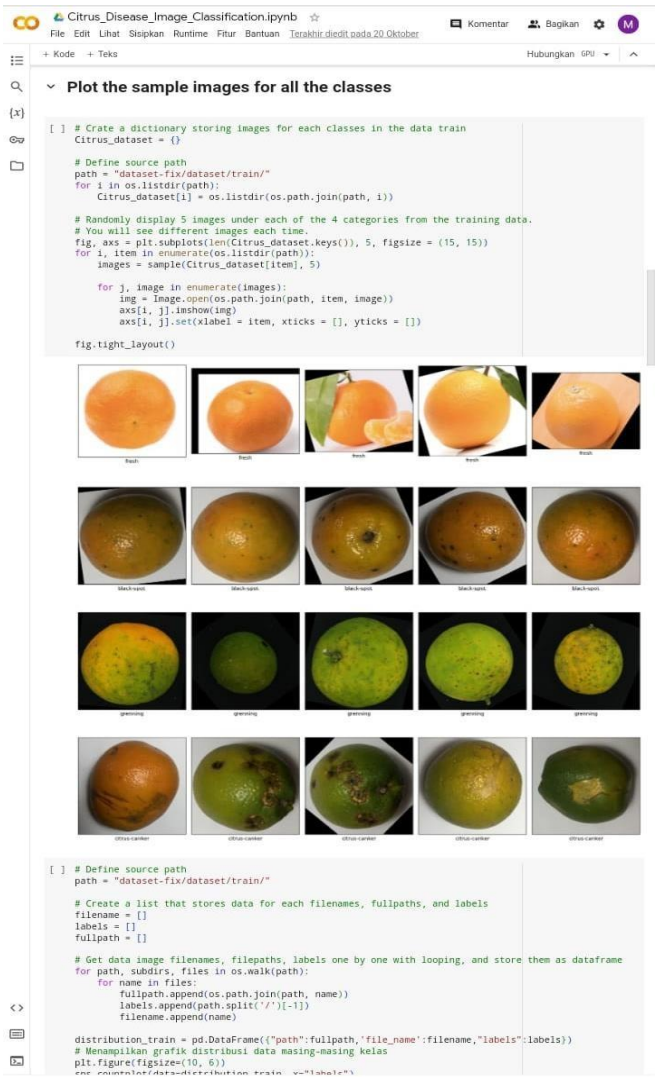
Pertama, terdapat inisialisasi dictionary transformations yang menyimpan nama- nama transformasi sebagai kunci dan fungsi terkait sebagai nilai. Selanjutnya, ditentukan lokasi gambar asli dengan images_path dan lokasi untuk menyimpan gambar yang telah diaugmentasi melalui augmented_path. List images digunakan untuk menyimpan nama-nama gambar yang akan diproses. Kemudian, kode menggunakan os.listdir untuk membaca semua nama file di direktori images_path dan menyimpannya dalam list images. Jumlah gambar yang ingin dihasilkan ditentukan sebanyak 600.

Proses augmentasi dilakukan dalam sebuah loop yang berjalan hingga mencapai jumlah gambar yang diinginkan. Dalam setiap iterasi, gambar dipilih secara acak dari list images, dibaca, dan transformasi diterapkan secara acak berdasarkan *dictionary transformations.* Setelah semua transformasi diterapkan, gambar yang telah diaugmentasi disimpan dengan nama berurutan di path yang ditentukan. Jika terjadi kesalahan saat membaca gambar, kesalahan tersebut

ditangkap dan ditampilkan tanpa menghentikan eksekusi program. Dengan cara ini, program mampu menghasilkan gambar augmentasi secara efisien sambil menangani berbagai kemungkinan kesalahan.

4.2 Eksplorasi Data

Dalam tahap eksplorasi data proyek " *Citrus disease detector*" sejumlah langkah penting telah diambil untuk memahami karakteristik *Dataset* dan mempersiapkannya untuk pengolahan lebih lanjut. Tahapan eksplorasi data melibatkan pemrosesan data hingga siap digunakan, statistik deskriptif, visualisasi data, dan pemahaman tentang nilai yang hilang. Gambar 4.1 Merupakan proses eksplorasi data pada penelitian ini.



Gambar 4.1 Proses Eksplorasi Data

Gambar 4.1 menggambarkan tahap eksplorasi data, di mana dataset diorganisir sesuai kelasnya. Gambar-gambar dalam dataset dikelompokkan dalam folder berdasarkan empat kategori utama: *black spot*, *canker*, *greening*, dan *fresh*. Langkah ini penting untuk memastikan data siap digunakan dalam pelatihan model. Setelah pengorganisasian selesai, proses dilanjutkan dengan pembagian data (*splitting data*), yaitu membagi dataset menjadi dua bagian: data latih dan data uji. Proporsi yang digunakan adalah 80% untuk data pelatihan dan 20% untuk data pengujian. Pembagian ini bertujuan untuk memberikan cukup data bagi model untuk belajar dan menguji performanya secara terpisah. Dengan struktur dan pembagian data yang baik, model dapat dilatih secara optimal untuk mendeteksi penyakit buah jeruk dengan akurasi yang lebih tinggi. Strategi ini menjadi dasar penting untuk keberhasilan penelitian.

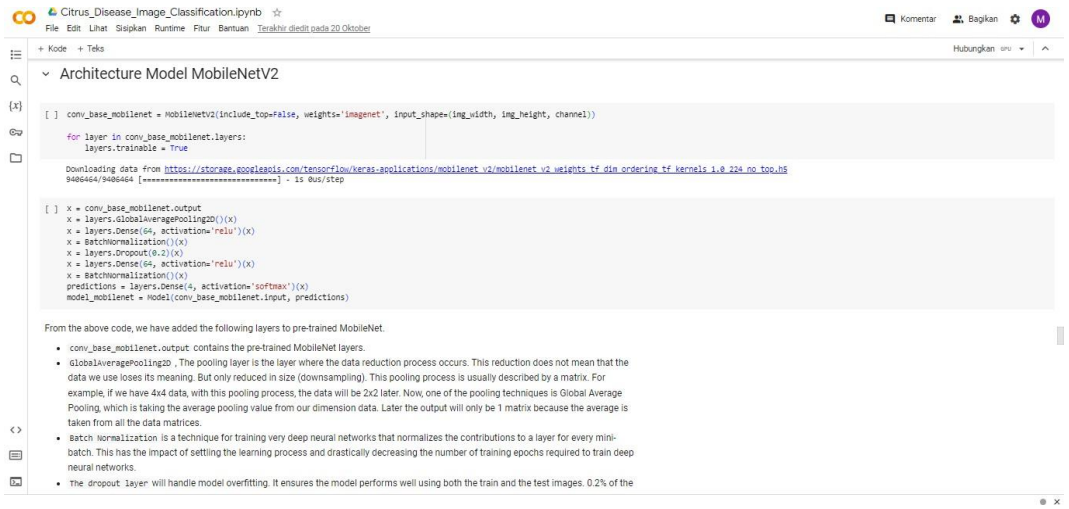
```
      # Variables used in this data separation where
variable x
      = data path and y = data labels
X_Datasetcitrus = df_Datasetcitrus['path']


      # Split the initial Dataset into training and
testing data # Use proportion 80% for training data and
20% for testing data
      X_train, X_test, y_train, y_test = train_test_split(
          X_Datasetc           y_datsetc           test_siz
    itrus,
```

Kode ini membagi dataset citrus menjadi data pelatihan dan pengujian dengan proporsi 80:20, di mana 80% data digunakan untuk melatih model (*X_train* dan *y_train*) dan 20% untuk menguji performa model (*X_test* dan *y_test*). Parameter *random_state=42* memastikan hasil pembagian data konsisten setiap kali kode dijalankan. Pembagian ini memungkinkan model memperoleh data pelatihan yang cukup dan data pengujian yang representatif untuk evaluasi performa. Proses ini penting untuk menjaga validitas hasil prediksi dan meminimalkan potensi *overfitting* selama pelatihan model.

4.3 Modeling

Dalam pembuatan proyek " *Citrus disease detector* " model yang kita gunakan yaitu model *mobileNetV2* . Arsitektur *MobileNetV2* didasarkan pada struktur sisa terbalik dimana masukan dan keluaran dari blok sisa merupakan lapisan hambatan tipis yang berlawanan dengan model sisa tradisional yang menggunakan representasi pada masukan. *MobileNetV2* menggunakan konvolusi mendalam yang ringan untuk memfilter fitur di lapisan ekspansi menengah. Model *mobileNetV2* ini memiliki nilai akurasi yang tinggi. Hal ini dapat dilakukan dengan cara menginput model *mobilenet*v2.predict. Gambar 4.2 merupakan aristektur *MobileNetV2* yang digunakan pada penelitian ini.



Gambar 4.2 Arsitektur MobilenetV2

Gambar 4.2 merupakan arsitektur MobileNetv2 yang digunakan pada penelitian ini. *MobileNetV2* adalah arsitektur jaringan saraf konvolusional yang dirancang untuk efisiensi dan kecepatan dalam pengolahan gambar. Arsitektur ini menggunakan *depthwise separable convolutions*, yang terdiri dari konvolusi kedalaman dan *konvolusi pointwise*, untuk mengurangi jumlah parameter dan komputasi. *mobilenetV2* juga mengintegrasikan blok residual terbalik yang menggabungkan konvolusi biasa dan *pointwise*, menjaga informasi sambil meningkatkan efisiensi. Selain itu, lapisan *bottleneck* dengan aktivasi linier di lapisan terakhir setiap blok membantu mempertahankan representasi data.

Arsitektur ini menggunakan *ReLU6* sebagai fungsi aktivasi, yang membatasi output untuk menjaga stabilitas dan efisiensi. Dengan penerapan *layer*

normalization, MobilenetV2 meningkatkan konvergensi saat pelatihan. Arsitektur ini sangat efisien dan cocok untuk aplikasi di perangkat dengan sumber daya terbatas, seperti perangkat *mobile* dan *IoT*. Gambar 4.3 merupakan hasil parameter *MobileNetV2*.

```
=================================================================================================
Total params: 2,344,900
Trainable params: 2,310,532
Non-trainable params: 34,368
_________________________________________________________________________________________________
```

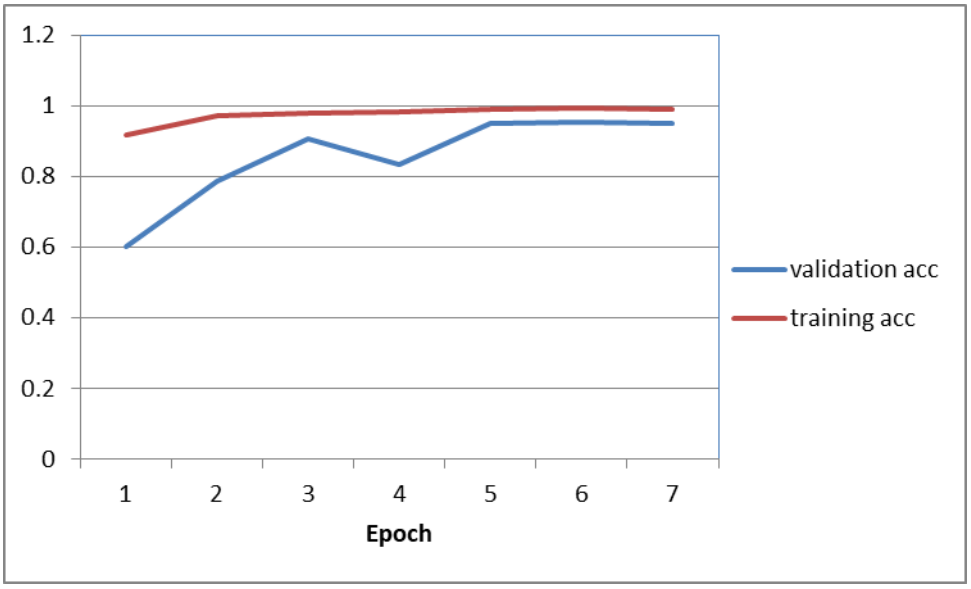
Gambar 4.3 Hasil Parameter MobileNetV2

Gambar 4.3 merupakan hasil parameter *MobileNet* V2. Hasil tersebut menjeaskan dengan menerapkan arsitektur *MobileNet* V2, jumlah total parameter yang dihasilkan adalah 2,344,900, di mana 2,310,532 di antaranya adalah parameter yang dapat dilatih, dan 34,368 adalah parameter yang tidak dapat dilatih.

```
CPU times: user 3 µs, sys: 1 µs, total: 4 µs
Wall time: 6.91 µs
Epoch 1/10
200/200 [==============================] - 63s 207ms/step - loss: 0.2611 - accuracy: 0.9186 - val_loss: 1.0819 - val_accuracy: 0.6005
Epoch 2/10
200/200 [==============================] - 32s 159ms/step - loss: 0.0972 - accuracy: 0.9706 - val_loss: 0.5587 - val_accuracy: 0.7852
Epoch 3/10
200/200 [==============================] - 30s 148ms/step - loss: 0.0766 - accuracy: 0.9775 - val_loss: 0.2256 - val_accuracy: 0.9058
Epoch 4/10
200/200 [==============================] - 31s 156ms/step - loss: 0.0599 - accuracy: 0.9818 - val_loss: 0.5075 - val_accuracy: 0.8329
Epoch 5/10
200/200 [==============================] - 31s 155ms/step - loss: 0.0473 - accuracy: 0.9887 - val_loss: 0.1286 - val_accuracy: 0.9510
Epoch 6/10
200/200 [==============================] - 32s 160ms/step - loss: 0.0294 - accuracy: 0.9919 - val_loss: 0.1463 - val_accuracy: 0.9548
Epoch 7/10
200/200 [==============================] - 33s 165ms/step - loss: 0.0288 - accuracy: 0.9903 - val_loss: 0.1503 - val_accuracy: 0.9485
```

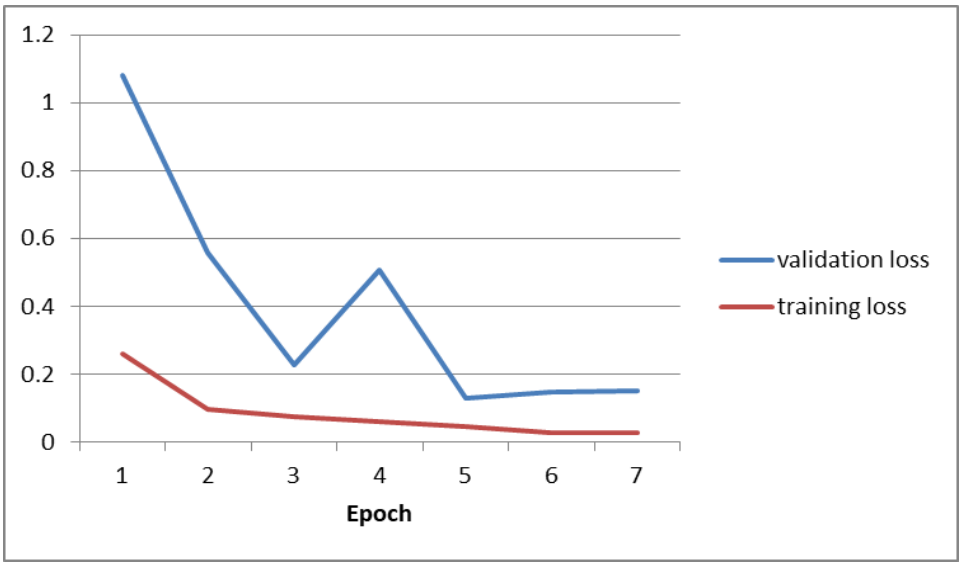
Gambar 4. 4 Hasil *Training Model* MobileNetV2

Gambar 4.4 diatas menjelaskan hasil evaluasi menunjukkan bahwa pada *epoch* pertama hingga *epoch* ketujuh perolehan akurasi mengalami peningkatan, baik itu *accuracy* maupun di val_*accuracy*. Dimana pada epoch ketujuh nilai perolehan *accuracy* sebesar 0.9903 dan nilai val_*accuracy* sebesar 0.9485. Sedangkan untuk nilai loss dari epoch pertama hingga epoch ketujuh perolehan nilai mengalami penurunan, baik itu di loss maupun di val_loss. Dimana pada epoch ketujuh nilai perolehan loss sebesar 0.0288 dan nilai val_loss sebesar 0.1503. Oleh sebab itu, model yang dirancang sudah dapat dikatakan model yang memiliki performa baik dalam melakukan kalsifikasi gambar dan mengidentikasi gambar sehingga sudah dapat digunakan. Pada Gambar 4.5 merupakan grafik hasil *training accuracy* dan *validation accuracy.*



Gambar 4.5 Grafik Hasil *Training Accuracy dan Validation Accuracy*

Gambar 4.5 Sumbu X (*horizontal*) menunjukkan jumlah *epoch* (siklus pelatihan penuh), Sumbu Y (*vertical*) menunjukkan nilai akurasi, garis untuk *training accuracy* menunjukkan perubahan akurasi pada data pelatihan selama pelatihan dan garis untuk *validation accuracy* menunjukkan perubahan akurasi data validasi selama pelatihan. Pada Gambar 4.6 merupakan grafik hasil dari *training loss* dan *validation loss.*

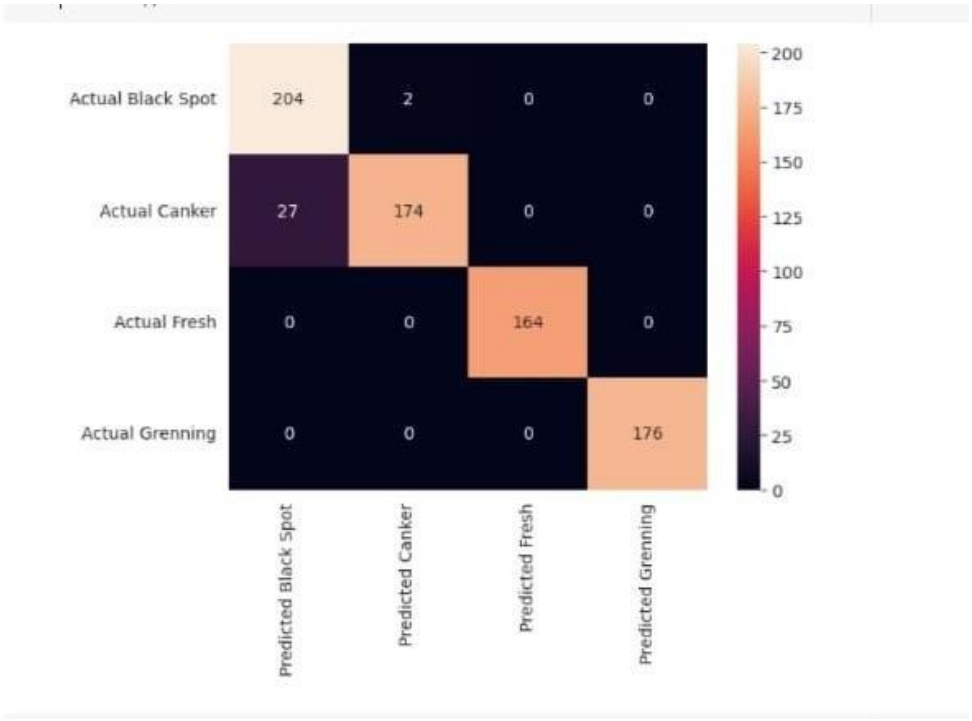


Gambar 4.6 Grafik Hasil *Training Loss* dan *Validation Loss* MobileNetV2

Gambar 4.6, sumbu X (horizontal) mewakili jumlah *epoch* atau siklus pelatihan penuh, sedangkan sumbu Y (vertikal) menunjukkan nilai akurasi. Garis akurasi pelatihan menggambarkan perubahan akurasi pada data pelatihan selama proses pelatihan, sementara garis akurasi validasi menunjukkan perubahan akurasi pada data validasi. Grafik ini digunakan untuk memvisualisasikan kinerja model dan membantu mengidentifikasi pola seperti *overfitting* atau *underfitting*. Jika akurasi validasi stagnan atau menurun meskipun akurasi pelatihan terus meningkat, hal itu mengindikasikan potensi *overfitting* pada model.

4.4 Evaluasi

Dalam evaluasi algoritma dan model yang digunakan untuk mendeteksi penyakit pada aplikasi website " *Citrus disease detector* " beberapa metrik penting digunakan untuk mengukur kinerja masing-masing algoritma. Beberapa metrik yang digunakan Pada perancangan aplikasi berbasis web deteksi penyakit buah jeruk yaitu *metrics precision, recall*, F1 *score,* dan *accuracy* serta menggunakan *confusion matrix* pada model tersebut. Evaluasi pada algoritma yang digunakan pada proyek memperoleh akurasi sebesar 0.9612. Akurasi merupakan metrik yang mengukur sejauh mana model mampu mengklasifikasikan dengan benar, yaitu jumlah prediksi yang benar dibagi dengan total prediksi. Akurasi yang tinggi menunjukkan bahwa algoritma ini memiliki kemampuan yang sangat baik . Dengan diperolehnya akurasi sebesar 0.9612 pada penelitian ini menandakan bahwa algoritma ini mampu memberikan prediksi yang akurat dalam mendeteksi penyakit pada tanaman jeruk. Pada Gambar 4.7 merupakan *confussion matrix* dari penelitian ini.



Gambar 4.7 *Confussion Matrix Model* Mobilenet V2

Gambar 4.7, ditampilkan hasil *confusion matrix* MobileNetV2 pada aplikasi website deteksi penyakit buah jeruk. *Confusion matrix* ini menunjukkan jumlah klasifikasi yang benar dan salah pada masing-masing kelas penyakit jeruk. Dengan menggunakan *confusion matrix*, dapat dianalisis berapa banyak kasus yang diklasifikasikan dengan tepat dan berapa banyak yang salah, serta identifikasi kelas mana yang memiliki tingkat kesalahan tertinggi. *Matrix* ini memberikan gambaran yang lebih jelas mengenai kinerja model dalam mendeteksi penyakit buah jeruk berdasarkan data yang diberikan, yang membantu dalam evaluasi dan peningkatan akurasi model deteksi penyakit.

1. Berikut *Black spot*

Tabel 4.1 merupakan hasil *confusion matrix* pada kelas *black spot.*

Tabel 4.1 *Confussion Matrix Black Spot*

	Positif	Negatif
Positif	204	27
Negatif	2	514

Tabel 4.1 menampilkan hasil *confusion matrix* untuk klasifikasi penyakit *black spot* pada buah jeruk. Dalam *matriks* tersebut, terdapat 204 *True Positif* (TP), yang berarti 204 gambar penyakit black spot terdeteksi dengan benar. *True Negatif* (TN) sebanyak 514 menunjukkan jumlah gambar yang tidak terdeteksi sebagai *black spot* dan memang bukan *black spot*. Sementara itu, terdapat 27 *False Positif* (FP), di mana gambar yang tidak mengandung penyakit malah diklasifikasikan sebagai *black spot*. Terakhir, 2 *False Negatif* (FN) berarti ada 2 gambar yang sebenarnya mengandung penyakit namun tidak terdeteksi.

2. *Citrus canker*

Berikut Tabel 4.2 merupakan *confussion matrix* pada kelas *citrus canker*.Tabel 4.2 Confussion Matrix Citrus Canker

	Positif	Negatif
Positif	174	2
Negatif	27	514

Tabel 4.2, ditampilkan hasil *confusion matrix* untuk klasifikasi penyakit citrus canker pada buah jeruk. Dalam matriks tersebut, terdapat 174 *True Positif* (TP), yang berarti 174 gambar penyakit citrus canker terdeteksi dengan benar. *True Negatif* (TN) sebanyak 514 menunjukkan jumlah gambar yang tidak terdeteksi sebagai citrus canker dan memang bukan citrus canker. Sementara itu, terdapat 2 *False Positif* (FP), di mana gambar yang tidak mengandung penyakit malah diklasifikasikan sebagai *citrus canker*. Terakhir, 27 *False Negatif* (FN) berarti ada 27 gambar yang sebenarnya mengandung penyakit namun tidak terdeteksi.

3. *Fresh*

Berikut Tabel 4.3 merupakan *confussion matrix* pada kelas *fresh.*

Tabel 4.3 *Confussion Matrix Fresh*

	Positif	Negatif
Positif	164	0
Negatif	0	583

Tabel 4.3, ditampilkan hasil confusion matrix untuk klasifikasi penyakit "fresh" pada buah jeruk. Dalam matriks tersebut, terdapat 164 True Positif (TP),

yang berarti 164 gambar yang benar-benar menunjukkan buah jeruk yang segar terdeteksi dengan benar. True Negatif (TN) sebanyak 583 menunjukkan gambar

yang tidak mengandung penyakit dan memang terklasifikasi sebagai buah jeruk segar. Tidak ditemukan False Positif (FP) maupun False Negatif (FN), yang menunjukkan bahwa model berhasil mendeteksi semua gambar segar dengan akurat tanpa kesalahan klasifikasi.

4. *Greening*

Berikut Tabel 4.4 merupakan merupakan *confussion matrix* pada kelas *greening.*

Tabel 4.4 *Confussion Matrix Greening*

	Positif	Negatif
Positif	176	0
Negatif	0	571

Tabel 4.4, disajikan hasil confusion matrix untuk klasifikasi penyakit "greening" pada buah jeruk. Matriks tersebut menunjukkan 176 True Positif (TP), yang berarti 176 gambar buah jeruk yang terinfeksi penyakit greening terdeteksi dengan benar. True Negatif (TN) sebanyak 571 menunjukkan gambar yang tidak mengandung penyakit greening dan terklasifikasi dengan tepat. Tidak ada False Positif (FP) atau False Negatif (FN), yang menandakan bahwa model berhasil mendeteksi semua gambar dengan akurasi tinggi tanpa kesalahan dalam klasifikasi.

Berdasarkan hasil yang diperoleh, beberapa jenis penyakit pada buah jeruk belum terklasifikasi dengan baik. Hal ini disebabkan oleh perbedaan karakteristik visual pada setiap penyakit yang menyebabkan kesulitan dalam membedakan antara satu penyakit dengan yang lainnya. Setiap jenis penyakit memiliki gejala atau tanda khas yang berbeda, yang membuat model kesulitan untuk membedakan dan mengklasifikasikan dengan akurasi tinggi. Misalnya, perbedaan warna, tekstur, dan pola pada kulit buah jeruk dapat mempengaruhi kemampuan model dalam mendeteksi penyakit secara tepat.

Hasil dari confusion matrix menunjukkan adanya kesalahan dalam klasifikasi, yang terdeteksi pada nilai False Positive (FP) atau False Negative (FN). Penyebab kesalahan ini sering kali berkaitan dengan kekurangan data yang

representatif atau variasi dalam kondisi buah jeruk yang diuji. Meskipun model menunjukkan kinerja yang baik secara keseluruhan, adanya kesalahan klasifikasi ini menunjukkan perlunya peningkatan dalam jumlah data pelatihan dan teknik augmentasi data untuk meningkatkan akurasi deteksi.

Sebagai langkah selanjutnya, penyesuaian pada model, seperti tuning parameter dan eksperimen dengan arsitektur yang lebih kompleks atau data yang lebih bervariasi, mungkin diperlukan untuk mencapai klasifikasi yang lebih akurat.

$$Accuracy = \frac{total\ TP}{total\ semua\ data}$$

$$Accuracy = \frac{204+174+164+176}{747}$$

$$Accuracy = 0{,}9612$$

Perhitungan ini menghasilkan nilai akurasi sebesar 0,9612. Selain adanya perhitungan akurasi, perhitungan- perhitungan lainnya dilakukan pada *confussion matrix* seperti *precision, recall dan F1-score*. Berikut merupakan perhitungan-perhitungan matrik pada deteksi penyakit buah jeruk sesuai kelasnya :

1. Perhitungan *black spot*

a) $Precision = \frac{TP}{TP+FP}$

$$precision = \frac{204}{204+27}$$

$$precision = 0{,}8831$$

b) $Recall = \frac{TP}{TP+FN}$

$$Recall = \frac{204}{204+2}$$

$$Recall = 0{,}9903$$

c) $F1\text{-}score = 2X\frac{Precision\,.\,Recall}{Precission+Recall}$

$$F1\text{-}score = 2X\frac{0{,}8831\,.\,0{,}9903}{0{,}8831+0{,}9903}$$

$$F1\text{-}score = 0{,}9336$$

Berdasarkan perhitungan diatas pada kelas *black spot* didapatkan nilai *precision* sebesar 0,8831, *recall* sebesar 0,9903 dan *F1-score* sebesar 0,9336.

2. Perhitungan *citrus canker*

a) $Precision = \frac{TP}{TP+FP}$

$precision = \frac{174}{174+2}$

$precision = 0{,}9986$

b) $Recall = \frac{TP}{TP+FN}$

$Recall = \frac{174}{174+27}$

$Recall = 0{,}8657$

c) $F1\text{-}score = 2X\frac{Precision\,.\,Recall}{Precission+Recall}$

$F1\text{-}score = 2X\frac{0{,}9986\,.\,0{,}8657}{0{,}9986+0{,}8657}$

$F1\text{-}score = 0{,}9231$

Berdasarkan perhitungan diatas pada kelas *citrus canker* didapatkan nilai *precision* sebesar 0,9986, *recall* sebesar 0,8657 dan *F1-score* sebesar 0,9231.

3. Perhitungan *fresh*

a) $Precision = \frac{TP}{TP+FP}$

$precision = \frac{164}{164+0}$

$precision = 1$

b) $Recall = \frac{TP}{TP+FN}$

$Recall = \frac{164}{164+0}$

$Recall = 1$

c) $F1\text{-}score = 2X\frac{Precision\,.\,Recall}{Precission+Recall}$

$F1\text{-}score = 2X\frac{1\,.\,1}{1+1}$

$F1\text{-}score = 1$

Berdasarkan perhitungan diatas pada kelas *fresh* didapatkan nilai *precision* sebesar 1, *recall* sebesar 1 dan *F1-score* sebesar 1.

4. Perhitungan *greening*

a) $Precision = \frac{TP}{TP+ FP}$

$precision = \frac{176}{176+0}$

$precision = 1$

b) $Recall = \frac{TP}{TP+ FN}$

$Recall = \frac{176}{176+0}$

$Recall = 1$

c) $F1\text{-}score = 2X\frac{Precision\ .\ Recall}{Precission+Recall}$

$F1\text{-}score = 2X\frac{1\ .\ 1}{1+1}$

$F1\text{-}score = 1$

Berdasarkan perhitungan diatas pada kelas *greening* didapatkan nilai *precision* sebesar 1, *recall* sebesar 1 dan *F1-score* sebesar 1. Pada Gambar 4.8 merupakan hasil pemodelan *confussion matrix* CNN.

	precision	recall	f1-score	support
Black Spot	0.8831	0.9903	0.9336	206
Canker	0.9886	0.8657	0.9231	201
Fresh	1.0000	1.0000	1.0000	164
Grenning	1.0000	1.0000	1.0000	176
accuracy			0.9612	747
macro avg	0.9679	0.9640	0.9642	747
weighted avg	0.9647	0.9612	0.9610	747

Gambar 4.8 Hasil *Model Confusion Matrix* CNN

Gambar 4.8 merupakan hasil dari model confussion matrix CNN yang dibuat oleh program dimana hasil tersebut sesuai dengan hasil perhitungan manual dengan menggunakan rumus.

4.5 *Deployment*

Tujuan dari *Deployment* pada aplikasi website " *Citrus disease detector* " adalah memungkinkan pengguna, dalam hal ini, pengusaha argibisnis jeruk, untuk

mengakses dan menggunakan alat deteksi pennyakit dengan mudah dan efisien. Proses *Deployment* aplikasi web ini dilakukan pada *local host*, sehingga untuk menjalankan aplikasi perlu menggunakan bantuan *command prompt*. Semua file yang diperlukan dimasukkan dalam satu folder. Pada *command prompt*, lokasi folder dibuka menggunakan command "cd". Lalu, program *flask* app.py dibuka dengan *command "python*". *Local host* akan berjalan dan menghasilkan alamat url untuk dibuka pada web browser. Web browser dan program akan berjalan, lalu data input dapat dimasukkan. Algoritma akan berjalan dan menghasilkan prediksi berdasarkan data input. Pada Gambar 4.9 merupakan proses *deployment* pada penelitian ini.

app.py

Citrus Proyek > app.py

```python
import base64, json
from flask import Flask, render_template, request, Response
import tensorflow as tf
from tensorflow.keras.preprocessing import image as image_utils
from PIL import Image, ImageFile
import numpy as np
import plotly
import plotly.express as px
import plotly.graph_objects as go
import cv2

app = Flask(__name__)

global citrus_fig, img_uploaded, citrus_pred

citrus_model = tf.keras.models.load_model('model\Citrus-Disease-Model.h5')
citrus_classes = ['black-spot', 'citrus-cancker', 'fresh', 'grenning']

camera = cv2.VideoCapture(0)

@app.route('/')
def index():
    return render_template('index.html', show_modal=True)

@app.route("/webcam")
def webcam():
    return render_template("webcam.html")

@app.route('/prediction', methods=['GET','POST'])
def pred():
    global citrus_fig, img_uploaded, citrus_classes, citrus_pred
    if request.method=='POST':
        file = request.files['file']

        img_bytes = file.read()
        img = tf.image.decode_jpeg(img_bytes, channels=3)
        img = tf.image.convert_image_dtype(img, tf.float32)
        img = tf.image.resize(img, [224, 224])
```

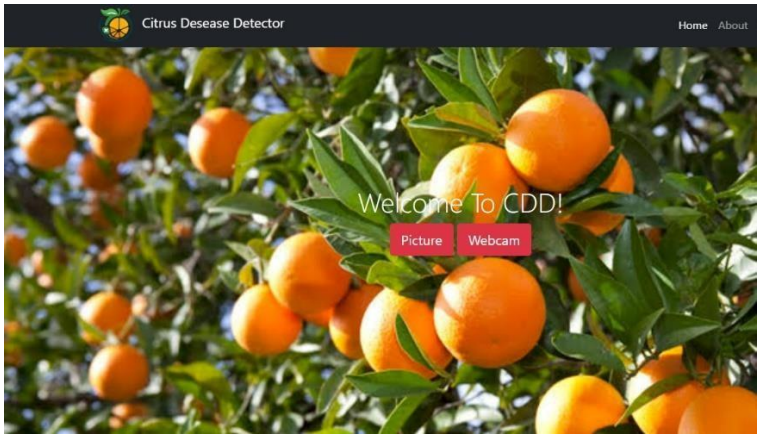
Gambar 4.9 Proses *Deployment*

Gambar 4.9 merupakan poses *deployment* pada aplikasi web deteksi penyakit buah jeruk mengugunakan metode CNN dengan arsitektur *MobileNetV2*. Proses *deployment* merupakan serangkaian langkah untuk merilis aplikasi perangkat lunak ke lingkungan produksi. Proses ini dimulai dengan persiapan kode yang telah selesai dan diuji, dilanjutkan dengan membangun kode menjadi versi yang dapat dijalankan. Setelah itu, dilakukan pengujian tambahan di lingkungan staging untuk memastikan semua fitur berfungsi dengan baik. Kemudian, aplikasi diunggah ke server produksi, dan verifikasi dilakukan untuk memastikan semuanya berjalan

lancar. Setelah *deployment*, pemantauan dilakukan untuk mendeteksi masalah yang mungkin muncul, dan jika diperlukan, langkah *rollback* dapat diambil untuk mengembalikan aplikasi ke versi sebelumnya.

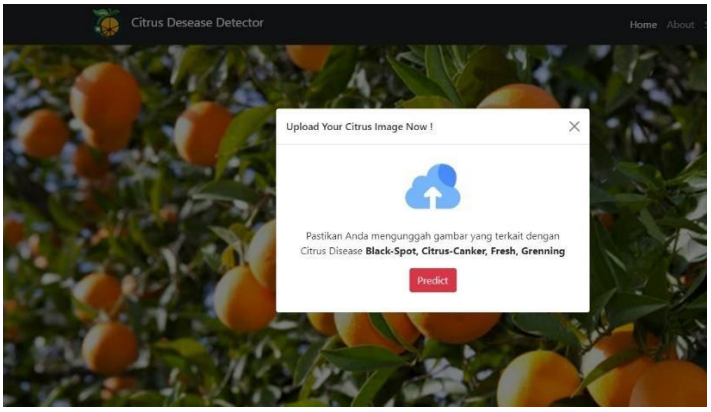
4.6 Tampilan

Antarmuka pengguna adalah tampilan grafis yang berinteraksi langsung dengan pengguna. Gambar 4.10 menunjukkan tampilan dashboard dari aplikasi web *Citrus Disease Detector*. Pada *dashboard* ini, pengguna dapat mengakses berbagai fitur untuk mendeteksi penyakit pada buah jeruk, baik melalui unggahan gambar maupun menggunakan kamera secara langsung. Antarmuka ini dirancang untuk memudahkan pengguna dalam melakukan deteksi penyakit secara efektif dan cepat, memberikan pengalaman pengguna yang interaktif dan mudah dioperasikan untuk memantau kondisi buah jeruk.



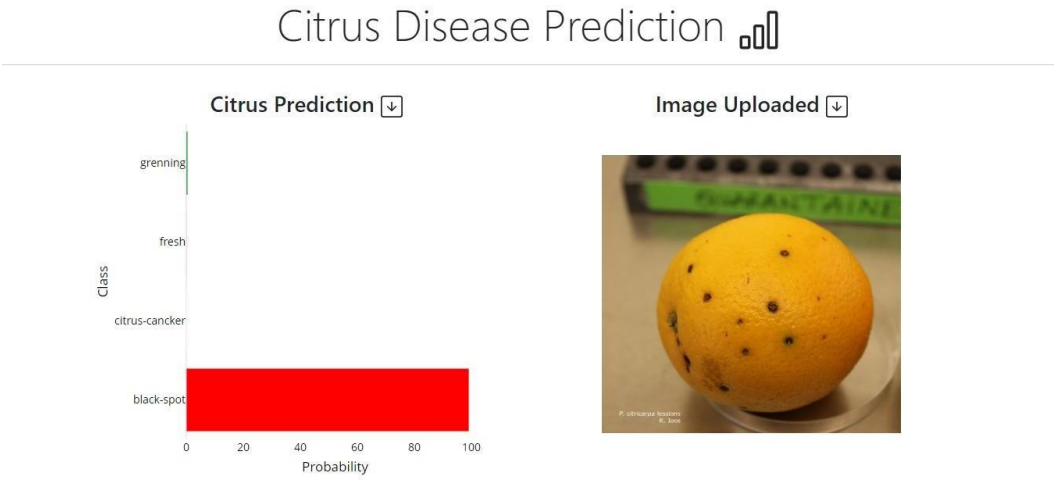
Gambar 4.10 Tampilan *Dashboard*

Gambar 4.13 merupakan tampilan fitur *webcam* yang memungkinkan deteksi penyakit buah jeruk secara real-time. Pada gambar tersebut, hasil deteksi menunjukkan bahwa buah jeruk yang diunggah dalam kondisi segar atau tidak terinfeksi penyakit. Fitur ini memberikan kemudahan untuk melakukan deteksi langsung tanpa perlu mengunggah gambar terlebih dahulu. Selanjutnya, pada Gambar 4.14 merupakan tampilan yang memberikan informasi lebih lanjut mengenai penyakit yang terdeteksi pada buah jeruk melalui aplikasi *Citrus Disease Detector*, memungkinkan pengguna untuk memahami lebih detail kondisi buah yang diidentifikasi.



Gambar 4.11 *Image Detection*

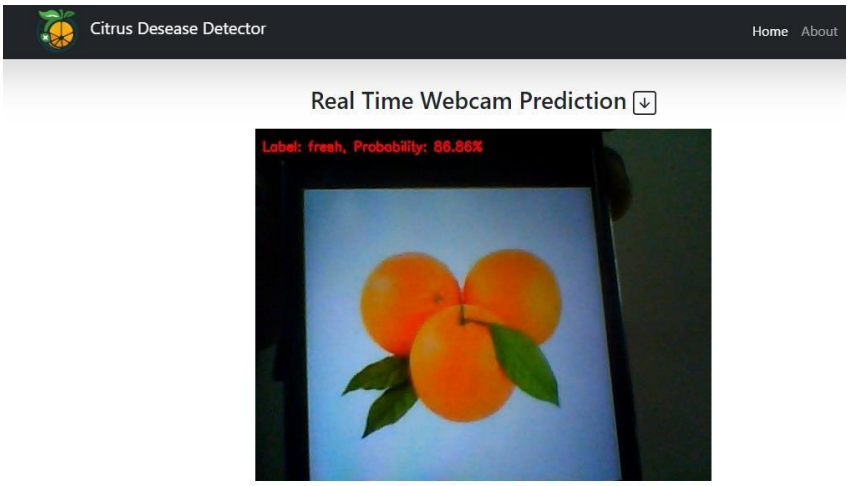
Gambar 4.11 ditampilkan halaman *image detection* pada aplikasi web, di mana pengguna dapat mengunggah gambar penyakit tanaman jeruk yang ingin dideteksi. Halaman ini memudahkan pengguna untuk memasukkan gambar sebagai input deteksi. Gambar 4.12 merupakan hasil deteksi dari gambar yang telah diunggah. Sistem akan memproses gambar dan menampilkan hasil berupa jenis penyakit yang terdeteksi, seperti *black spot*, *canker*, *greening*, atau *fresh*. Tampilan ini mempermudah pengguna untuk memahami hasil analisis secara cepat dan intuitif.



Gambar 4.12 Hasil Deteksi

Gambar 4.12 merupakan tampilan hasil deteksi penyakit pada buah jeruk menggunakan aplikasi *Citrus Disease Detector*. Hasil menunjukkan bahwa buah jeruk tersebut terdeteksi terkena penyakit *black spot*. Proses deteksi dilakukan dengan mengunggah gambar melalui fitur yang tersedia di aplikasi. Gambar 3.13 merupakan tampilan fitur *webcam*, yang memungkinkan pengguna mengambil

gambar langsung dari kamera untuk dideteksi secara real-time. Fitur ini menambah kemudahan dan fleksibilitas dalam proses identifikasi penyakit pada buah jeruk.



Gambar 4.13 Tampilan Fitur *Webcam*

Gambar 4.13 merupakan tampilan fitur *webcam* yang memungkinkan deteksi penyakit buah jeruk secara real-time. Pada gambar tersebut, hasil deteksi menunjukkan bahwa buah jeruk yang diunggah dalam kondisi segar atau tidak terinfeksi penyakit. Fitur ini memberikan kemudahan untuk melakukan deteksi langsung tanpa perlu mengunggah gambar terlebih dahulu. Gambar 4.14 merupakan tampilan yang memberikan informasi lebih lanjut mengenai penyakit yang terdeteksi pada buah jeruk melalui aplikasi *Citrus Disease Detector*, memungkinkan pengguna untuk memahami lebih detail kondisi buah yang diidentifikasi.

Gambar 4.14 Tampilan Informasi Citrus

Gambar 4.14, ditampilkan halaman "*About*" pada aplikasi web *Citrus Disease Detector*. Halaman ini menyediakan informasi tentang berbagai penyakit yang dapat menyerang buah jeruk. Pengguna dapat memperoleh pengetahuan lebih mendalam mengenai setiap jenis penyakit yang terdeteksi oleh aplikasi ini. Dengan informasi yang lengkap, pengguna dapat memahami gejala, penyebab, dan cara pencegahan penyakit-penyakit tersebut, sehingga mereka dapat merawat buah jeruk dengan lebih baik. Fitur ini membantu meningkatkan kesadaran dan memberikan edukasi yang berguna bagi pengguna dalam mengidentifikasi dan menangani penyakit tanaman jeruk.

BAB V PENUTUP

5.1 Kesimpulan

Berdasarkan hasil temuan penelitian, dapat disimpulkan hal-hal berikut:

1. Perancangan untuk deteksi penyakit jeruk menggunakan metode CNN dengan algoritma *mobilenet* V2 berhasil dilakukan dengan baik menggunakan alur sesuai dengan metode penelitian yang telah dirancang.
2. Penelitian ini dilakukan menggunakan Metrik precision, recall, F1 score, dan accuracy serta pemanfaatan confusion matrix pada model tersebut. Hasil dalam evaluasi tersebut mendapatkan *accuracy* sebesar 96,12% yang artinya program ini sangat berjalan dengan baik.

5.2 Saran

Berikut saran dari deteksi tingkat kematangan buah melinjo mnggunakan metode algoritma *self organizing map* yaitu:

1. Saat pengambilan citra buah jeruk diharapkan memiliki kualitas gambar yang lebih bagus lagi, dan juga jarak pengambilan gambar yang statis.
2. Diharapkan dalam penelitian selanjutnya, dapat dilakukan menerapkan metode lainnya selain algoritma *mobilenet V2.*
3. Diharapkan penelitian selanjutnya dapat memanfaatkan *software* yang lain sehingga ada pengetahuan terbaru terkait penelitian penyakit buah jeruk ini.

DAFTAR PUSTAKA

[1] N. J. Harahap, " Mahasiswa dan revolusi Industri 4.0," *Journal of industrial revolution 4.0*, vol. 3, no. 5, pp. 1–2, 2019.

[2] P. Studi Statistika, Retno Dwi Pusptasari, "Pertanian Berkelanjutan Berbasis Revolusi Industi 4.0 ," Jurnal pertanian revolusi 4.0, vol. 1, no. 1, 2020

[3] C. Wariyah, "*Vitamin C Retention And Acceptability Of Orange (Citrus Nobilis Var. Microcarpa) Juice During Storage In Refrigerator*," *Jurnal AgriSains*, vol. 1, no. 1, 2020.

[4] M. Yesuf, "*Pseudocercospora Leaf and Fruit Spot Disease Of Citrus: Achievements And Challenges In The Citrus Industry: A review," Agricultural Sciences*, vol. 04, no. 07, pp. 324–328, 2013, doi: 10.4236/as.2013.47046.

[5] J. A. Ridjal, "Analisis Faktor Determinan Keikusertaan Petani Berkelompok, Pendapatan dan Pemasaran Jeruk Siam di Kabupaten Jember," *Journal of Social and Agricultural Economics*, vol. 2, no. 1, pp. 1–9, 2018.

[6] A. Arif, " Sistem Pakar Hama Dan Penyakit Tanaman Jeruk Gerga Pagar Alam Menggunkan Metode *Euclidean Distance Berbasis Website*," *Jurnal Teknologi Informasi Mura*, vol. 11, no. 02, pp. 68–75, Dec. 2019, doi: 10.32767/jti.v11i02.610.

[7] S. Lee, G. Choi, H. C. Park, and C. Choi, "*Automatic Classification Service System for Citrus Pest Recognition Based on Deep Learning*," *Sensors*, vol. 22, no. 22, Nov. 2022, doi: 10.3390/s22228911.

[8] W. Shen, Y. Wu, Z. Chen, and H. Wei, "*Grading method of leaf spot disease based on image processing," in Proceedings - International Conference on Computer Science and Software Engineering, CSSE 2008*, 2008, pp. 491–494. doi: 10.1109/CSSE.2008.1649.

[9] R. Pakpahan, "Analisa Pengaruh Implementasi *Artifical Inteligence* Dalam Kehidupan Manusia," *Journal of Information System, Informatics and Computing Issue Period*, vol. 5, no. 2, pp. 506–513, 2021, doi: 10.52362/jisicom.v5i2.616.

[10] Z. Unal*, "Smart Farming Becomes Even Smarter With Deep learning - A Bibliographical Analysis," IEEE Access*, vol. 8, pp. 105587–105609, 2020, doi: 10.1109/ACCESS.2020.3000175.

[11] E. Anggiratih, S. Siswanti, S. K. Octaviani, and A. Sari, "Klasifikasi Penyakit Tanaman Padi Menggunakan Model *Deep learning Efficientnet B3* Dengan *Transfer Learning," Jurnal Ilmiah SINUS*, vol. 19, no. 1, p. 75, Jan. 2021, doi: 10.30646/sinus.v19i1.526.

[12] A. Amandri Achyar, A. Muhammud Olow, M. Rizky Perdana, A. Sundawijaya, and A. Dhiyaanisafa Goenawan, "Identifikasi Ras Wajah dengan Menggunakan Metode *Deep learning* Model Keras," 2022.

[13] Y. A. Suwitono and F. J. Kaunang, "Implementasi *Algoritma Convolutional Neural Network* (CNN) Untuk Klasifikasi Daun Dengan Metode Data Mining SEMMA Menggunakan Keras," *Jurnal Komtika (Komputasi dan Informatika)*, vol. 6, no. 2, pp. 109–121, Nov. 2022, doi: 10.31603/komtika.v6i2.8054.

[14] D. I. Swasono, M. Abuemas, R. Wijaya, and A. Hidayat, "Klasifikasi Penyakit pada Citra Buah Jeruk Menggunakan *Convolutional Neural Networks* (CNN) dengan Arsitektur *Alexne*t," 2023. [Online]. Available: https://*www.kaggle.com/Datasets/jonathansilva2020/orange-*

[15] S. Mudholakar, K. G, K. K. K T, and S. G V, "*Automatic Detection of Citrus Fruit and Leaves Diseases Using Deep Neural Network,*" *Int J Res Appl Sci Eng Technol*, vol. 10, no. 7, pp. 4043–4051, Jul. 2022, doi: 10.22214/ijraset.2022.45868.

[16] A. Hadhiwibowo, S. R. Asri, and R. A. Dinata, "Penerapan *Convolutional Neural Network* dengan Arsitektur *Mobilenetv2* Pada Aplikasi Penerjemah dan Pembelajaran Bahasa Isyarat," *TIN: Terapan Informatika Nusantara*, vol. 4, no. 8, pp. 518–523, Jan. 2024, doi: 10.47065/tin.v4i8.4879.

[17] J. A. Ridjal, "Analisis Faktor Determinan Keikutsertaan Petani Berkelompok, Pendapatan dan Pemasaran Jeruk Siam Di Kabupaten Jember," *J-SEP*, vol. 2, no. 1, 2019.

[18] Widyati and S. Hindarti, "Analisis Pendapatan Usaha TaniDan Pemasaran Jeruk Keprok," *JU-ke (JurnalKetahananPangan)*, vol. 5, no. 1, 2021.

[19] W. Datika *et al.*, "Motivasi Membangun Kebon Jeruk Keprok RGL (Rimau Gerga Lembong) Di Kelurahan Agung Lawangan Kecematan Dempo Utara Kota Pagar Alam)," pp. 40–50, 2019.

[20] Y. D. Puspita, L. Sulistyowati, and S. Djauhari, "Eksplorasi Jamur Endofit Pada Tanaman Jeruk (*Citrus sp*) Fusiprotoplas dengan Ketahanan Berbeda Terhadap *Botriodiplodia theobromae Pat*," 2020.

[21] A. Eryana, Sriyanto, and Firmansyah, "Prediksi Malaria Menggunakan Metode *Pre-Trained* Model Algoritma *EfficientNet-B0* dan *MobileNet-V2*," *Jurnal Ilmiah Komputasi*, vol. 22, no. 1, Mar. 2023, doi: 10.32409/jikstik.22.1.3332.

[22] G. N. Agrios, *Plant Pathology 5th Eds*. California, USA: Elsevier Academic Press, 2005.

[23] K. Rajabasa, K. Lampung, and P. Lampung, "Jurnal Pengabdian Fakultas Pertanian Universitas Lampung Menggunakan Terumbu Buatan Di Perairan Desa Jurnal Pengabdian Fakultas Pertanian Universitas Lampung," vol. 02, no. 01, pp. 280–293, 2023.

[24] I. G. P. Wirawan, S. Simanjuntak, M. Sritamin, and N. Wijaya, "Detection of Citrus Vein Phloem Degeneration (CVPD) disease and the quality of healthy fruits in nutrient deficiency of citrus," *Bali Med. J.*, vol. 6, no. 3, p. 117, 2017, doi: 10.15562/bmj.v6i3.757.

[25] S. Tang, S. Yuan, and Y. Zhu, "Convolutional Neural Network in Intelligent Fault Diagnosis Toward Rotatory Machinery," *IEEE Access*, vol. 8, pp. 86510–86519, 2020, doi: 10.1109/ACCESS.2020.2992692.

[26] R. Indraswari, R. Rokhana, and W. Herulambang, "Melanoma image classification based on MobileNetV2 network," *Procedia Comput Sci*, vol. 197, pp. 198–207, 2022, doi: 10.1016/j.procs.2021.12.132.

[27] Y. Miftahuddin and F. Zaelani, "Perbandingan Metode *Efficientnet-B3* dan *Mobilenet-V2* Untuk Identifikasi Jenis Buah-buahan Menggunakan Fitur Daun," 2022.

[28] E. Elfatimi, R. Eryigit, and L. Elfatimi, "*Beans Leaf Diseases Classification Using MobileNet Models," IEEE Access,* vol. 10, pp. 9471–9482, 2022, doi: 10.1109/ACCESS.2022.3142817.

[29] T. Yusnanto and D. Lestiono, "Optimalisasi Penggunaan CMD dan Sysinternalsuits Sebagai Malware Dettection," *Jurnal TRANSFORMASI (Informasi & Pengembangan Iptek)*, vol. 15, no. 1, 2019.

[30] paul kirvan, *Learning CMD*," *IJCIT (Indonesian Journal on Computer and Information Technology*, vol. 3, no. 1, pp. 31–2, . 2021.

[31] B. Wijaya and A. Pratama, "Deteksi Penyusupan Pada Server Menggunakan Metode *Intrusion Detection* System (IDS) Berbasis *Snort*," *Sistem Informasi dan Komputer)*, vol. 09, pp. 97–101, doi: 10.32736/sisfokom.v9.i1.770.

[32] I. H. Al Amin, "*Artficial Intelligence* dalam Proses Industri Manufaktur," *Jurnal Teknologi Informasi DINAMIK*, vol. XIV, no. 2, pp. 98–104, 2019.

[33] A. Tasyah *et al.*, "Pengenalan Kecerdasan Buatan Kepada Para Remaja di Komunitas Perpus Jungle Parung Panjang," 2021.

[34] T. Wahyudi, "Studi Kasus Pengembangan dan Penggunaan *Artificial Intelligence* (AI) Sebagai Penunjang Kegiatan Masyarakat Indonesia," *Indonesian Journal on Software Engineering (IJSE)*, vol. 9, no. 1, pp. 28–32, 2023, [Online]. Availablehttp://ejournal.bsi.ac.id/ejurnal/index.php/ijse28

[35] I. T. Madhini, E. Rahmawati, N. N. Rohmah, Y. Saudi, Ishanan, and Fathurijal, "Kecerdasan Buatan dalam Produksi Konten Penyairan : Peluang dan Tantangan," *Seminar Nasional Paedagoria*, vol. 4, no. 2087–8705, pp. 613–613, 2024.

[36] A. Santoso and G. Ariyanto, "Implementasi *Deep learning* Berbasis Keras Untuk Pengenalan Wajah," *Jurnal Teknik Elektro*, vol. 18, no. 01, [Online]. Available: https://www.mathworks.com/discovery/convol

[37] M. Haris, T. Pustaka, M. H. Diponegoro, S. Kusumawardani, and I. Hidayah,

[38] "Tinjauan Pustaka Sistematis: Implementasi Metode *Deep learning* pada Prediksi Kinerja Murid (*Implementation of Deep learning Methods in Predicting Student Performance: A Systematic Literature Review*)," 2021.

[39] I. W. S. E. P., A. Y. Wijaya, and R. Soelaiman, "Klasifikasi Citra Menggunakan *Convolutional Neural Network* (Cnn) *pada* Caltech 101," *Jurnal Teknik ITS*, vol. 05, no. 1, pp. A65–A65, 2020.

[40] R. Darma Nurfita and G. Ariyanto, "Implementasi *Deep learning* Berbasis Tensorflow Untuk Pengenelan Sidik Jari," *Jurnal Teknik Elektro*, vol. 18, no. 1, pp. 23–23, 2021.

[41] I. W. S. E. P, A. Yudhi Wijaya, and R. Soelaiman, "Klasifikasi Citra Menggunakan *Convolutional Neural Network* (Cnn) pada Caltech 101," *JURNAL TEKNIK ITS*, vol. 5, no. 1, 2019.

[42] B. Dwi Hartomo, "Penerapan Computer Vision Untuk Absensi Wajah Berbasis Algoritma CNN Pada Guru SMK Excellent 1 Tangerang," 2021.

[43] K. Dong, C. Zhou, Y. Ruan, and Y. Li, "*MobileNetV2 Model for Image Classification," in Proceedings - 2020 2nd International Conference on Information Technology and Computer Application, ITCA 2020, Institute of Electrical and Electronics Engineers Inc., Dec.* 2020, pp. 476–480. doi: 10.1109/ITCA52113.2020.00106.

[44] N. A. S. Badrulhisham and N. N. A. Mangshor, *"Emotion Recognition Using Convolutional Neural Network* (CNN)," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Jul. 2021. doi: 10.1088/1742-6596/1962/1/012040.

[45] T. Yusnanto and D. Lestiono, "Optimalisasi Penggunaan CMD dan Sysinternalsuits Sebagai Malware Dettection," *Jurnal TRANSFORMASI (Informasi & Pengembangan Iptek)*, vol. 15, no. 1, 2019.

[46] Paul Kirvan, *Learning CMD*," *IJCIT (Indonesian Journal on Computer and Information Technology*, vol. 3, no. 1, pp. 31–2, . 2021.

[47] B. Wijaya and A. Pratama, "Deteksi Penyusupan Pada Server Menggunakan Metode *Intrusion Detection* System (IDS) Berbasis *Snort*," *Sistem Informasi dan Komputer)*, vol. 09, pp. 97–101, doi: 10.32736/sisfokom.v9.i1.770.

[48] L. Indriyani, W. Susanto, D. Riana, N. Stmik, and J. Mandiri, "Teknik Pengolahan Citra Menggunakan Aplikasi Matlab Pada Pengukuran Diameter Buah Jeruk Keprok," *IJCIT (Indonesian Journal on Computer and Information Technology*, vol. 2, no. 1, pp. 46–52, 2020.

[49] N. Nafi'iyah, "Algoritma *Kohonen* dalam Mengubah Citra *Graylevel* Menjadi Citra Biner," *Jurnal Ilmiah Teknologi dan Informasia ASIA (JITIKA)*, vol. 9, no. 2, 2021.

[50] A. B. Kaswar, A. A. N. Risal, Fatiah, and Nurjannah, "Klasifikasi Tingkat Kematangan Buah Markisa Menggunkan Jaringan Syaraf Tiruan Berbasis Pengolahan Citra Digital," *JESSI*, vol. 1, no. 1, 2020.

[51] M. N. Winnarto, M. Mailasari, and A. Purnamawati, "Klasifikasi Jenis Tumor Otak Menggunakan Arsitektur *MobileNetV2*," *Jurnal SIMETRIS*, vol. 13, no. 2, 2022.

[52] Christovao Jonathan *"Dataset for Classification Of Citrus Diseases". [Online].Available:https://www.kaggle.com/Datasets/jonathansilva2020/Dataset- for-classification-of-citrus-diseases. [Accesed 20 September 2023].*

[53] Christovao Jonathan "*Orange diseases Dataset", [Online]. Available : https://www.kaggle.com/Datasets/jonathansilva2020/orange-diseases-Dataset [Accesed 20 September 2023].*

LAMPIRAN A

Proses Augmentasi

```
      # Created a function to perform anti clockwise
rotation def anticlockwise_rotation(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img,
    (224,224)) angle=
    random.randint(0,180) return
    rotate(img, angle)

      # Create a function to perform clockwise
rotation def clockwise_rotation(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img,
    (224,224)) angle=
    random.randint(0,180) return
    rotate(img, -angle)

      # Create a function to flip images up and
down def flip_up_down(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img,
    (224,224)) return
    np.flipud(img)
      # Create a function to give a bright effect to
the image def add_brightness(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img, (224,224))
          img = adjust_gamma(img,
    gamma=0.5,gain=1) return img

      # Create a function to give the image a blur/blur
effect def blur_image(img):
          img = cv2.cvtColor(img, 0)
```

```
      # Create a function to give the image a sheared
effect def sheared(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img, (224,224))
    transform = AffineTransform(shear=0.2)
          shear_image = warp(img, transform,
    mode="wrap") return shear_image

      # Create a function to perform warp
shifts def warp_shift(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img, (224,224))
          transform =
    AffineTransform(translation=(0,40)) warp_image =
    warp(img, transform, mode="wrap") return
    warp_image
```

```
      # Create a transformation variable that
will hold all the preprocessing processes that
have been made above
Transformations={'rotateanticlockwise':anticlockw
ise_rotat ion,
                      'rotate clockwise':
                       clockwise_rotation, 'warp
                       shift': warp_shift,
                             'blurring image':
                       blur_image,
                             'add
                       brightness' :
                       add_brightness,
                       'flip up down':
                       flip_up_down, 'shear
                       image': sheared
    }

      images_path="Dataset-
fix/Dataset/train/fresh" # Path for the
original image
      augmented_path=" Dataset-
fix/Dataset/train/fresh " # Path to put the
augmented image
      images=[] # To save images that have
been preprocessed from the folder

      # Read image name from folder and add path
into "images" array
for im in os.listdir(images_path):
    images.append(os.path.join(images_path,im))

      # The number of images that will be
added with the results of the augmentation
transformation, the number is adjusted
according to needs
      # Variable to iterate up to a predefined
number of images_to_generate
      ima
ges_to_ge
nerate=60
0 i=1

while
    i<=images_to_ge
    nerate:
    image=random.cho
    ice(images) try:
              original image = io.imread(image)
```

```
      select and call methods

  transformed_image =
            transformations[key](original_image) n = n +
            1

       new_image_path="%s/augmented_image_%s.jpg"
      %(augmented_path, i)

    transformed_image = img_as_ubyte(transformed_image) #
Convert images to unsigned byte format, with values in
[0, 255]
         cv2.imwrite(new_image_path, transformed_image) #
Save the result of the augmentation transformation on the
image into the specified path
              i =i+1
          except ValueError as e:
           print('could not read the',image ,':',e,'hence
skipping it.')
```