# BAB I

# PENDAHULUAN

## A. Latar Belakang

Jagung merupakan salah satu komoditas pertanian yang memiliki peran penting dalam ketahanan pangan dan perekonomian di berbagai negara, termasuk Indonesia. Terdapat beberapa jenis jagung yang umum dibudidayakan dan dikonsumsi, di antaranya jagung manis (Zea mays saccharata), jagung pulut (Zea mays ceratina), dan jagung pakan (Zea mays indentata/dent corn). Masing-masing jenis jagung memiliki karakteristik fisik dan kandungan kimia yang berbeda, yang menentukan penggunaannya dalam konsumsi langsung, industri pangan, maupun pakan ternak.

Permasalahan yang sering terjadi di lapangan adalah sulitnya membedakan jenis-jenis jagung secara visual oleh petani, pedagang, maupun konsumen awam. Proses identifikasi secara manual cenderung subjektif dan memerlukan pengalaman serta waktu yang cukup lama. Oleh karena itu, diperlukan solusi otomatis dan akurat untuk mengklasifikasikan jenis jagung berdasarkan citra visualnya.

Pengolahan citra digital telah menjadi pendekatan yang efektif dalam bidang pertanian presisi (precision agriculture), khususnya dalam identifikasi dan klasifikasi tanaman. Salah satu metode yang populer saat ini adalah **You Only Look Once (YOLO)**, yaitu algoritma deep learning berbasis deteksi objek yang mampu mengidentifikasi dan mengklasifikasikan objek dalam satu tahap (single-

stage detection). YOLO memiliki keunggulan dalam hal kecepatan dan akurasi tinggi, sehingga sangat cocok untuk diaplikasikan dalam skenario real-time.

Dengan menggunakan YOLO, proses identifikasi jenis jagung dapat dilakukan secara otomatis dengan memanfaatkan citra dari kamera atau foto. Metode ini tidak hanya efisien dari segi waktu, tetapi juga meningkatkan objektivitas dalam klasifikasi jenis jagung. Oleh karena itu, pengembangan sistem identifikasi jenis jagung berbasis pengolahan citra digital dengan YOLO memiliki potensi besar untuk membantu petani, pelaku agribisnis, dan industri pangan dalam memastikan kualitas dan jenis produk jagung yang ditangani.

## B. Rumusan Masalah

Bagaimana merancang dan membangun aplikasi identifikasi jenis - jenis jagung berbasis pengolahan citra digital menggunakan algoritma YOLO secara akurat dan efisien?

## C. Tujuan Penelitian

Dari rumusan masalah diatas, maka dapat ditentukan tujuan dari penelitian ini adalah:

1. Mengembangkan model identifikasi otomatis untuk membedakan jagung manis, jagung pulut, dan jagung pakan menggunakan algoritma YOLO.
2. Menganalisis performa model YOLO dalam hal akurasi, presisi, dan waktu inferensi untuk klasifikasi jenis jagung.

## D. Batasan Masalah

Adapun Batasan masalah dalam penelitian ini :

1. Sistem hanya mengidentifikasi tiga jenis jagung**,** yaitu jagung manis**,** jagung pulut**,** dan jagung pakan**.**
2. Sistem ini hanya fokus pada proses **deteksi dan klasifikasi visual**, tanpa mempertimbangkan faktor non-visual seperti rasa, kadar air, atau tekstur biji.

## E. Manfaat Penelitian

Manfaat dari penelitian ini yaitu Dengan adanya sistem identifikasi yang tepat, produk jagung dapat diklasifikasikan sesuai jenisnya sejak awal. Hal ini sangat bermanfaat dalam menjaga kualitas produk yang akan dipasarkan dan mencegah pencampuran antar jenis jagung yang memiliki tujuan penggunaan berbeda, seperti konsumsi langsung atau pakan ternak.

## F. Sistematika Penulisan

Untuk memberikan gambaran singkat mengenai isi tulisan secara keseluruhan, maka akan di uraikan beberapa tahapan dari penulisan secara sistematis, yaitu :

BAB I : PENDAHULUAN

Bab ini menjelaskan dasar-dasar dari penulisan laporan tugas akhir, yang terdiri dari latar belakang, rumusan masalah, batasan masalah, tujuan, serta sistematika penulisan laporan tugas akhir.

BAB II : TINJAUAN PUSTAKA

Bab ini berisi teori-teori tentang hal-hal yang digunakan dalam Perancangan Aplikasi Deteksi Jenis – Jenis Jagung Berbasis

Pengelolahan Citra Digital menggunakan metode YOLO

BAB III : METODE PENELITIAN

Bab ini membahas tentang waktu dan tempat penelitian, metode penelitian serta alat dan bahan yang digunakan dalam penelitian

BAB IV : PERANCANGAN DAN IMPLEMENTASI

Bab ini berisi tentang percancangan sistem yang berjalan, analisis sistem yang di usulkan, rancangan sistem, rancangan input dan rancangan output serta pengujian sistem.

BAB V : PENUTUP

Bab ini berisi tentang kesimpulan yang dihasilkan dari pembahasan tentang Aplikasi Deteksi Jenis – Jenis Jagung Berbasis Pengelolahan Citra Digital menggunakan Metode YOLO untuk memudahkan dalam deteksi jenis jagung, dan beberapa saran sebagai hasil dari penelitian yang telah dilakukan.

# BAB II

# TINJAUAN PUSTAKA

## A. KAJIAN TEORI

### 1. Identifikasi

Identifikasi merupakan proses pengenalan, menempatkan obyek atau individu dalam suatu kelas sesuai dengan karateristik tertentu (Bachtiar, 2012).

Identifikasi sebagai tanda kenal diri, bukti dari penentu atau penetapan identitas seorang, sehingga mengidentifikasi memiliki arti upaya menentukan atau menetapkan identitas seseorang (Sasrawan, 2011).

Hakim, (2010:http://imadiklus.com) menyatakan bahwa kata "identifikasi" berasal dari bahasa asing, yaitu bahasa Inggris. Asal kata to identify sebagai kata kerja, dan identification sebagai benda. To identify secara sederhana artinya adalah mengenali. Hubungannya jika dikaitkan dengan "identifikasi kebutuhan belajar" artinya ialah mengenali kebutuhan belajar seseorang atau masyarakat atau kelompok orang tertentu yang akan menjadi sasaran didik atau peserta didik.

### 2. Jagung

Adapun penjelasan tentang Jagung Manis, Jagung Pulu dan Jagung Pakan :

a. Jagung Manis

Jagung manis (zea mays saccharata ) atau yang lebih dikenal dengan nama Sweet corn mulai dikembangkan di Indonesia pada awal tahun 1980,

diusahakan secara komersil dalam skala kecil untuk memenuhi kebutuhan masyarakat. Jagung manis semakin populer dan dikonsumsi karena memiliki rasa yang lebih manis, aroma lebih harum dan kandungan gizi yang lebih tinggi dibandingkan dengan jagung biasa, serta aman dikonsumsi bagi penderita diabetes karena mengandung gula sukrosa dan rendah lemak. Keistimewaan lain yang dimiliki Jagung manis adalah biji, dari butiran jagung manis lebih khas, tidak lembek dan memiliki serat yang tidak terlalu liat. Hal ini menyebabkan jagung manis banyak digemari kalangan menengah keatas dan masyarakat perkotaan sehingga banyak ditemukan di pasar swalayan (Anonim, 2013).

Adapun Ciri – ciri khas jagung manis :

1) Rasa dan Tekstur :
   - Rasa : Jagung manis dikenal karena kandungan gula yang lebih tinggi dibandingkan dengan varietas jagung lainnya, memberikan rasa manis yang khas saat dimasak.
   - Tekstur : Butirannya lebih lembut dan lebih juicy, yang membuatnya cocok untuk dikonsumsi langsung sebagai makanan atau camilan.
2) Penampilan :
   - Butir : Biasanya memiliki butiran yang berwarna kuning cerah atau variasi warna lain, seperti putih atau merah, tergantung pada varietasnya.

- Cekungan: Tidak memiliki cekungan pada butiran seperti jagung dent; butirannya cenderung lebih bulat dan tidak cekung setelah matang.

b. Jagung Pulut

Jagung Pulut (Zea mays ceratina) termasuk bahan pangan penting dan dapat diolah menjadi beras jagung untuk sumber pangan alternatif. Selain itu dapat dimanfaatkan sebagai pangan, pakan, bahan baku energi dan industri. Tekstur jagung pulut atau jagung ketan yaitu lengket dan lembut karena mengandung amilopektin hingga 90% (Hamzah et al., 2011). Fitur utama Jagung Pulut :

1) Penampilan dan Tesktur :
   - Butiran : Jagung pulu biasanya memiliki butiran yang lebih keras dan kering dibandingkan dengan jagung manis. Setelah matang dan mengering, butirannya menunjukkan cekungan di tengahnya.
   - Warna : Biasanya memiliki warna yang bervariasi, dari kuning hingga putih, tergantung pada varietas dan kondisi tumbuh.
2) Penggunaan :
   - Pakan Ternak : Jagung pulu adalah sumber utama pakan ternak, terutama untuk sapi, babi, dan unggas, karena kandungan pati yang tinggi dan biaya produksinya yang relatif rendah.
   - Industri : Jagung ini digunakan untuk pembuatan produk industri seperti etanol, minyak jagung, dan produk olahan lainnya. Selain itu,

jagung pulu juga digunakan dalam pembuatan tepung jagung dan cornmeal.

- Produk Olahan : Jagung pulu dapat diolah menjadi berbagai produk seperti sirup jagung, malt, dan bahan baku untuk makanan olahan.

c. Jagung Pakan

Jagung pakan adalah varietas jagung yang dirancang khusus untuk pakan ternak dan industri. Dengan kandungan pati yang tinggi dan tekstur yang lebih keras, jagung ini memainkan peran penting dalam produksi pakan ternak dan berbagai produk industri. Budidayanya memerlukan perhatian khusus terhadap kondisi tumbuh, pemupukan, dan pengendalian hama untuk memastikan hasil yang optimal dan efisien.

Fitur utama Jagung Pakan :

1) Penampilan dan Tesktur :
   - Butiran : Memiliki butiran yang lebih keras dan lebih kering dengan kandungan pati yang tinggi. Tekstur butiran jagung pakan umumnya lebih kasar dibandingkan dengan jagung manis.
   - Warna : Biasanya berwarna kuning,
2) Penggunaan
   - Pakan Ternak : Jagung pakan adalah sumber utama pakan ternak karena kandungan patinya yang tinggi, yang merupakan sumber energi yang penting untuk hewan. Ini digunakan dalam pakan untuk berbagai jenis ternak, termasuk sapi, babi, dan unggas.

- Produk industri : Selain ppakan ternak, jagung pakan juga digunakan dalam industri untuk membuat produk seperti etanol, minyak jagung, dan berbagai bahan olahan lainnya.

**3. Pengolahan Citra Digital**

Pengolahan citra digital mengacu pada pemrosesan gambar dua dimensi menggunakan komputer. Dalam konteks yang lebih luas, pengolahan citra digital mengacu pada pemrosesan setiap data 2 dimensi (Putra, 2010). Pengolahan citra digital bertujuan untuk memperbaiki kualitas gambar (peningkatan kontras, translasi, skala, transformasi geometrik), melakukan proses penarikan informasi/ deskripsi objek atau pengenalan objek yang terkandung dalam citra, melakukan kompresi atau reduksi data untuk tujuan penyimpanan data, transmisi data, dan waktu proses data (Hermawati, 2013).

Pengolahan citra digital satelit MODIS dilakukan dengan perangkat lunak pengolahan citra (SeaDAS, ENVI, dan ArcGIS) untuk memperoleh informasi suhu permukaan laut dan klorofil-a yang kemudian digunakan dalam pemetaan daerah potensial penangkapan ikan di Selat Makassar (Selao et al, 2019).

Pengolahan citra digital adalah salah satu bentuk pemrosesan informasi dengan inputan berupa citra (image) dan keluaran yang juga berupa citra atau dapat juga bagian dari citra tersebut. Tujuan dari pemrosesan ini adalah memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin computer.

Operasi dasar Citra Digital :

a) Analis Citra (Image Analysis)

Analisis citra berfungsi untuk menghitung nilai kuantitatif gambar untuk membuat penggambarannya dengan mengeluarkan sorotan tertentu yang digunakan untuk mengidentifikasi objek. Proses segmentasi terkadang diperlukan untuk melokalisasi objek yang diinginkan.

b) Ekstraksi Citra

Ekstraksi adalah siklus untuk mengambil atau melihat nilai unsur-unsur yang terkandung dalam sebuah gambar. Nilai yang muncul atau yang diekstrak akan digunakan untuk proses pelatihan (training). Proses ekstraksi fitur merupakan proses yang mengasumsikan bagian penting dalam mempersiapkan objek info yang akan dikenali.

c) Transfomasi Citra (Image Transformation)

Transformasi citra merupakan proses perubahan suatu bentuk citra dimana dapat berupa perubahan geometri piksel seperti perputaran (rotatio), pergeseran (translation). Penskalaan (scalling), dan lain sebagainya. Transformasi citra berfungsi untuk memperoleh informasi (feature extraction) yang lebih jelas dalam suatu citra. Hasil transformasi dapat dianalisis kembali serta dijadikan acuan untuk melakukan pemrosesan selanjutnya (Putra, 2010).

**4. Deep Learning**

Deep learning (DL) merupakan salah satu cabang dari Machine Learning yang terfokus pada penggunaan algoritma pemodelan abstraksi tingkat tinggi pada data. Pendekatan ini menggunakan serangkaian fungsi transformasi non-linear

yang disusun secara berlapis-lapis dan mendalam. DL efektif dalam penerapan supervised learning, unsupervised learning, semi-supervised learning, dan reinforcement learning dalam berbagai konteks, termasuk pengenalan citra, suara, klasifikasi teks, dan sebagainya.

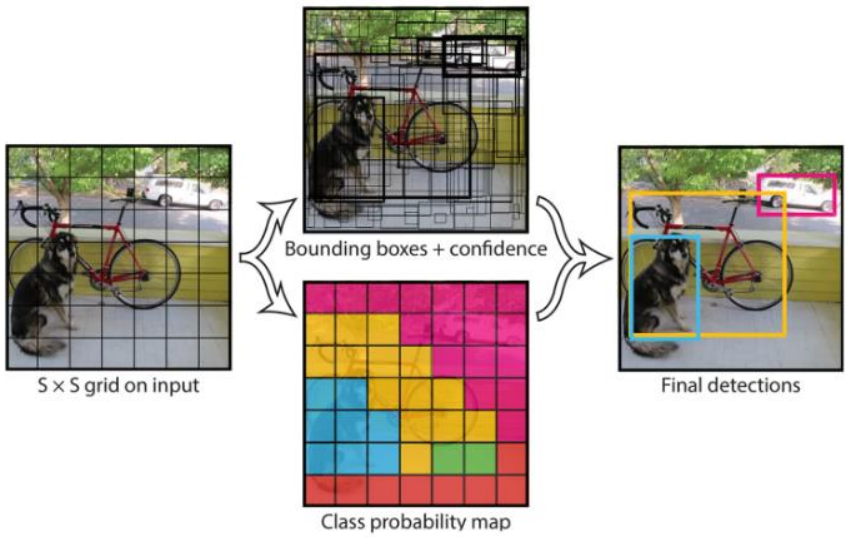
Model dalam deep learning pada dasarnya dibangun berdasarkan jaringan saraf tiruan (Neural Network), yang telah menjadi fokus riset sejak era 80-an. Namun, popularitasnya kembali meningkat, terutama karena perkembangan komputer yang semakin cepat, didukung oleh teknologi Big Data seperti Hadoop dan Spark berbasis multi-node cluster, serta pemrosesan paralel berbasis GPU, dan sebagainya (Cholissodin & Soebroto, 2021).

***5. You Only Look Once (YOLO)***

You Only Look Once (YOLO) adalah metode yang memungkinkan komputer mengenali objek secara real time. Sistem deteksi yang digunakan adalah dengan menggunakan pengklasifikasi atau locator yang dapat digunakan kembali. Model ini diterapkan pada gambar pada berbagai posisi dan skala.Wilayah gambar yang memperoleh skor tertinggi dianggap sebagai deteksi (Azman et al., 2022).

Versi YOLO yang digunakan pada penelitian ini yaitu YOLO v5. YOLO v5 melakukan proses *training* lebih cepat dari *efficientDet*. Model YOLOv5 paling akurat yang dapat memproses gambar berkali-kali lebih cepat dengan akurasi dengan model *EfficientDet* .

Dengan menentukan koordinat tinggi, koordinat lebar, koordinat x , koordinat y dan nilai probabilitas bounding box untuk mendeteksi sebuah objek yang mana dari semua atribut bounding box yang telah melalui proses normalisasi akan menghasilkan antara 0 dan 1 . Penentuan koodinat x dan y yang dinormalisasikan dengan penyesuaian titik kiri atas dan *grid* yang saling bersangkutan sesuai ukuran gambar.

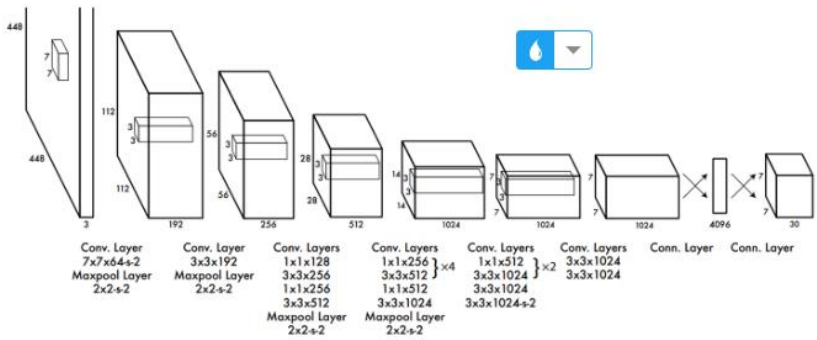



**Gambar 2. 1** Ilustrasi Proses Normalisasi

Arsitektur algoritma YOLO menggunakan jaringan saraf konvolusional dengan 24 lapisan konvolusional diikuti oleh dua lapisan yang terhubung penuh. Lapisan konvolusional digunakan untuk mengekstrak fitur dari gambar masukan, dan lapisan yang terhubung sepenuhnya bertanggung jawab untuk memprediksi probabilitas dan koordinat keluaran.

YOLO mengambil gambar masukan dan membaginya menjadi petak-petak berukuran S × S dalam suatu grid. Jika pusat objek berada di dalam suatu petak, maka tanggung jawab deteksi objek jatuh pada sel petak tersebut. Setiap

sel petak bertugas memprediksi B kotak pembatas dan nilai confidence untuk setiap kotak yang diprediksi. Nilai confidence ini mencerminkan tingkat confidence seberapa besar kemungkinan kotak tersebut mengandung objek, serta seberapa akurat prediksi kotak tersebut. Secara formal, YOLO mendefinisikan nilai confidence sebagai P r(Object) ∗ IoUtruth pred.




**Gambar 2. 2** Arsitektur YOLO

## 6. Python

Python adalah bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif. Python bisa dibilang bahasa pemrograman dengan tujuan umum yang dikembangkan secara khusus untuk membuat source code mudah dibaca. Python juga memiliki library yang lengkap sehingga memungkinkan programmer untuk membuat aplikasi yang mutakhir dengan menggunakan source code yang tampak sederhana (Ljubomir Perkovic, 2012). Adapun fitur – fitur utama python :

a. Sintaks Dinamis

Python adalah bahasa pemrograman yang berorientasi pada sintaks dinamis, yang berarti Anda tidak perlu mendeklarasikan tipe variabel secara eksplisit. Bahasa ini akan secara otomatis menentukan tipe data variabel berdasarkan nilainya.

b. Indentasi

Python menggunakan indentasi untuk mengatur blok kode, bukan tanda kurung kurawal seperti pada banyak bahasa pemrograman lainnya. Ini membuat kode Python lebih mudah dibaca dan terstruktur dengan baik.

c. Pustaka Standar

Python dilengkapi dengan pustaka standar yang kaya, yang mencakup modul untuk berbagai tugas seperti pemrosesan string, berkas, jaringan, dan banyak lagi. Anda dapat mengimpor modul ini untuk memperluas fungsionalitas Python.

d. Dukungan untuk berbagai platform

Python dapat digunakan di berbagai platform, termasuk Windows, macOS, dan Linux. Anda dapat menulis kode Python di satu platform dan menjalankannya di platform lain tanpa perubahan yang signifikan.

e. Memori manajemen otomatis

Python memiliki manajemen memori otomatis, yang berarti Anda tidak perlu secara eksplisit mengalokasikan dan membebaskan memori. Python akan menangani tugas ini untuk Anda.

Momen warna digunakan untuk menggambarkan distribusi statistik warna dalam citra.




**Gambar 2. 3** Logo Python

**7. VSC (Visual Studio Code )**

*Visual Studio Code* adalah sebuah lingkungan pengembangan terintegrasi IDE *(Integrated Development Environmen*t) yang dikembangkan oleh Microsoft. IDE ini menyediakan berbagai alat dan fitur untuk membantu pengembang dalam proses pengembangan perangkat lunak. Aplikasi ini juga dapat digunakan untuk membuat serta mengedit source code memakai berbagai bahasa pemrograman seperti Node.js, TypeScript, dan JavaScript. Bahkan untuk visualnya tergolong kompatibel dengan bahasa serta runtime environment lainnya seperti .NET, Java, bahasa Python, dan PHP. Berikut adalah beberapa fitur utama *Visual Studio Code* :

a. Editor Kode:

- *Visual Studio* menyertakan editor kode yang kuat dengan penyorotan sintaks, penyelesaian otomatis, dan integrasi pengelolaan kode sumber.

- Mendukung berbagai bahasa pemrograman seperti C#, C++, VB.NET, F#, Python, dan banyak lagi.

b. Debugging:
   - Fitur debugging yang canggih dengan kemampuan langkah demi langkah, poin henti, dan penelursuran variabel.
   - Dukungan untuk debug aplikasi lokal atau linkungan yang di host.

c. IntelliSense:
   - Menyediakan penyelesaian kode otomatis dan informasi kontekstual saat menulis kode.
   - Membantu meningkatkan produktivitas dan mengurangi kesalahan.

d. Designer Antarnuka Pengguna (UI):
   - Alat desain UI yang memungkinkan pengembang membuat antarmuka pengguna dengan mudah menggunakan drag-and-drop
   - Mendukung pembuatan antarmuka untuk aplikasi dekstop, web, dan seluler.

e. Pengelolaan Proyek:
   - Alat untuk membuat, mengelola, dan memelihara proyek.
   - Mendukung pengelolaan referensi, paket *NuGet*, dan integrasi dengan sistem kontrol versi seperti *Git.*

f. Pendukung Pengembangan Berkas Besar:
   - Dukungan untuk pengembangan proyek besar dengan ribuan file dan struktur folder.

g. Integrasi dengan *Azure:*

- Integrasi yang kuat dengan *platform cloud Microsoft Azure* untuk pengembangan, pengujian, dan penyebaran aplikasi *cloud*.

h. Pendukung Pemecahan Masalah:

- Alat untuk menganalisis kode dan memberikan saran perbaikan untuk meningkatkan kualitas dan keamanan kode.

i. Pengujian dan Performa:

- Alat pengujian otomatis dan analisis performa untuk membantu mengidentifikasi dan memper
- Baik masalah performa aplikasi.

j. Ekosistem Ekstensi:

- Duukungan untuk ekstensi pihak ketiga yang memperluas fungsionalitas Visual Studio.

k. Pemeliharaan Kode Sumber

- Integrasi dengan sistem kontrol versi seperti Git dan kemampuan untuk melihat histori perubahan, melakukan gabungan (*merge*), dan lainnya.



**Gambar 2. 4** Logo Visual Studio Code

### *8. OpenCV*

*OpenCV* adalah perpustakaan fungsi pemrograman untuk visi komputer real-time. *OpenCV* bersifat open source dan dapat digunakan untuk membantu masalah akademis dan komersial. *OpenCV* memiliki antarmuka untuk C, C++, Python, dan Java dan berjalan di Windows, Mac, Linux, dan Android (budiarjo, 2020). Kata *open* dalam *OpenCV* berarti open source yang artinya gratis dan siapa saja dapat mendownload dan menggunakannya. Di sisi lain, *CV* dalam *OpenCV* adalah singkatan dari *Computer Vision*, yang mengacu pada komputer yang digunakan untuk memproses gambar yang diambil oleh kamera atau webcam yang diubah dari analog ke digital dan kemudian diproses di dalam komputer .berarti Pemrosesan citra memiliki berbagai tujuan, seperti meningkatkan kualitas citra dan mengidentifikasi citra.

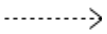
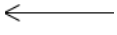
### 9. UML (Unifide Modeling Language)

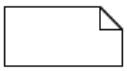
Menurut peneliti (Kroenke et al., 2018), UML adalah seperangkat diagram, struktur, dan teknik untuk memodelkan dan merancang program dan aplikasi berorientasi objek. Sedangkan menurut (Rumpe, 2017), UML digunakan sebagai notasi untuk berbagai kegiatan, seperti memodelkan kasus bisnis, menganalisis bentuk sistem, serta arsitektur dan desain awal. Peneliti (Seidl, Scholz, Huemer, & Kappel, 2015) juga menjelaskan bahwa UML menyajikan berbagai aspek sistem perangkat lunak yang sangat beragam dalam satu kerangka kerja menggunakan konsep object oriented.

Metode *Unified Modelling Language* menggunakan tiga bangunan dasar untuk mendeskripsikan sistem atau perangkat lunak yang akan dikembangkan, yaitu:

1. Sesuatu *(things)*
2. Relasi *(relationship)*
3. Diagram, *Unified Modelling Language* menyediakan sembilan jenis diagram yang dikembangkan berdasarkan sifatnya(statis dan dinamis) yaitu:
   a. *Use Case Diagram*

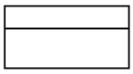
**Tabel 2. 1** Simbol - simbol use case diagram

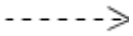
| NO | GAMBAR | NAMA | KETERANGAN |
|---|---|---|---|
| 1 |  | *Actor* | Mengspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*. |
| 2 |  | *Dependency* | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*). |
| 3 |  | *Generalization* | Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*). |
| 4 |  | *Include* | Mengspesifikasikan bahwa *use case* sumber secara *eksplisit*. |

| NO | GAMBAR | NAMA | KETERANGAN |
|---|---|---|---|
| 5 |  | *Extend* | Mengspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan. |
| 6 |  | *Association* | Apa yang menghubungkan antara objek satu dengan objek lainnya. |
| 7 |  | *System* | Menspesifikasikan paket yang menampilkan sistem secara terbatas. |
| 8 |  | *Use Case* | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor |
| 9 |  | *Collaboration* | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemen nya (sinergi). |
| 10 |  | *Note* | Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi |

*b. Class diagram*

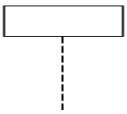
**Tabel 2. 2 Simbol-simbol Class Diagram**

| NO | GAMBAR | NAMA | KETERANGAN |
|---|---|---|---|
| 1 |  | *Generalization* | Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*). |
| 2 |  | *Nary Association* | Upaya untuk menghindari asosiasi dengan lebih dari 2 objek. |
| 3 |  | *Class* | Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama. |
| 4 |  | *Collaboration* | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor |
| 5 |  | *Realization* | Operasi yang benar-benar dilakukan oleh suatu objek. |

| NO | GAMBAR | NAMA | KETERANGAN |
|---|---|---|---|
| 6 |  | *Dependency* | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri |

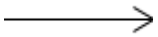
*c. Sequence Diagram*

**Tabel 2. 3** Simbol-simbol Sequence Diagram

| NO | GAMBAR | NAMA | KETERANGAN |
|---|---|---|---|
| 1 |  | *LifeLine* | Objek *entity*, antarmuka yang saling berinteraksi. |
| 2 |  | *Message* | Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi |
| 3 |  | *Message* | Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi |

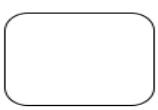
*d. State Chart Diagram*

**Tabel 2. 4** Simbol-simbol Satate Chart Diagram

| NO | GAMBAR | NAMA | KETERANGAN |
|---|---|---|---|
| 1 |  | *State* | Nilai atribut dan nilai link pada suatu waktu tertentu, yang dimiliki oleh suatu objek. |
| 2 |  | *Initial Pseudo State* | Bagaimana objek dibentuk atau diawali |
| 3 |  | *Final State* | Bagaimana objek dibentuk dan dihancurkan |
| 4 |  | *Transition* | Sebuah kejadian yang memicu sebuah state objek dengan cara memperbaharui satu atau lebih nilai atributnya |
| 5 |  | *Association* | Apa yang menghubungkan antara objek satu dengan objek lainnya. |
| 6 |  | *Node* | Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi. |

*e. Activity Diagram*

**Tabel 2. 5** Simbol-simbol Activity Diagram

| NO | GAMBAR | NAMA | KETERANGAN |
|---|---|---|---|
| **1** |  | *Activity* | Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain |

| NO | GAMBAR | NAMA | KETERANGAN |
|---|---|---|---|
| **2** |  | *Action* | State dari sistem yang mencerminkan eksekusi dari suatu aksi |
| **3** |  | *Initial Node* | Bagaimana objek dibentuk atau diawali. |
| **4** |  | *Activity Final Node* | Bagaimana objek dibentuk dan dihancurkan |
| **5** |  | *Fork Node* | Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran |

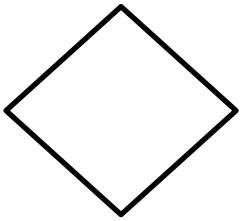
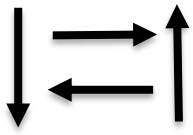
**10. Flowchart**

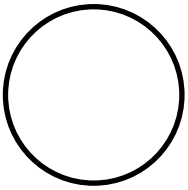
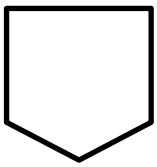
Flowchart dalam Bahasa Indonesia diterjemahkan sebagai Diagram Alir. Dari dua kata ini, maka dapat kita bayangkan bahwa flowchart itu berbentuk diagram yang bentuknya dapat mengalirkan sesuatu. Hal ini memang benar, flowchart memang melukiskan suatu aliran kegiatan dari awal hingga akhir mengenai suatu langkah – langkah dalam penyelasaian suatu masalah. Masalah tersebut bias bermacam – macam, mulai dari masalah yang sederhana sampai yang kompleks. Masalah yang kita pelajari tentu saja masalah pemrograman dengan menggunakan komputer, tetapi secara logika dapat kita awali dengan mengamati permasalahan dalam kehidupan sehari – hari kita (Hasyim, 2023).

Flowchart adalah reprentasi grafik dari langkah – langkah yang harus diikuti dalam menyelesaikan suatu permasalahan yang terdiri atas sekumpulan simbol, dimana masing – masing simbol merepresentasikan suatu kegiatan tertentu.

Flowchart diawali dengan penerimaan input, dan diakhiri dengan penampilan output. Dengan kata lain flowchart merupakan suatu gmbar yang menjelaskan urutan: pembacaan data, pemrosesan data, pengambilan keputusan terhadap data, penyajian hasil pemrosesan data (Hasyim, 2023).

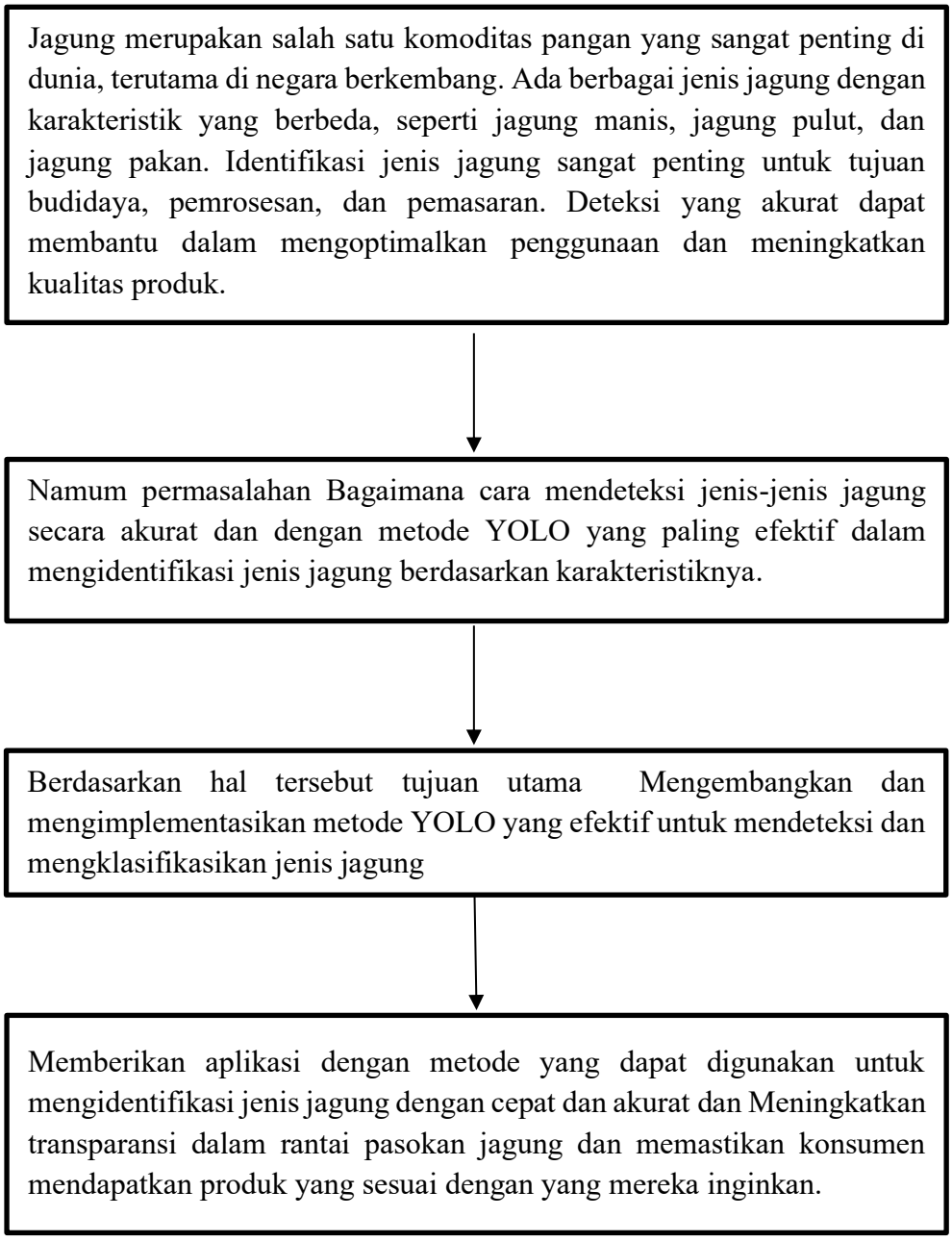
**Tabel 2. 6** Simbol-simbol Flowchart

| No. | Simbol | Keterangan |
|---|---|---|
| 1. |  | Simbol ini digunakan untuk melambangkan kegiataan penerimaan input. Dalam symbol ini, kita dapat menuliskan input yang diperlukan pada suatu waktu secara satu per satu maupun secara keseluruhan. |
| 2. |  | Simbol ini digunakan untuk melambangkan kegiatan pemrosesan input. Dalam simbol ini, kita dapat menuliskan operasi – operasi yang dikenakan pada input, maupun operasi lainnya. |
| 3. |  | Simbol ini digunakan untuk melambangkan kegiatan penampilan output. Dalam simbol ini, kita dapat menuliskan semua output yang harus di tampilkan oleh deprogram. |
| 4. |  | Simbol ini digunakan untuk melambangkan percabangan, yaitu pemeriksaan suatu kondisi. Dalam simbol ini, kita menuliskan keadaan yang harus dipenuhi. Hasil dari pemeriksaan dalam simbol ini adalah “YES” atau “NO” . |
| |  | Simbol ini berperang sebagai blok pembagunan dari suatu program. Prosedur memiliki suatu flowchart yang terdiri sendiri diluar flowchart utama. Jadi dalam simbol ini, kita cukup menuliskan nama prosedurnya saja. |
| 5. |  | Simbol ini digunakan untuk menghubungkan setiap langkah dalam flowchart dan menunjukkan kemana arah aliran diagram anak panah ini harus mempunyai arah dari kiri ke kanan atau dari atas ke bawah |

| | | |
|---|---|---|
| 6. |  | Terminator ini berfungsi untuk menandai awal dan akhir dari suatu flowchart. |
| 7. |  | On Page connector digunakan untuk menghubungkan suatu langkah dengan langka lain dari flowchart dalam satu halaman. |
| 8. |  | Off page connector digunakan untuk menghubungkan suatu langkah dengan langkah lain dari flowchart dalam halaman yang berbeda. |

## Kerangka Pikir

Untuk memahami kerangka pikir penelitian ini dapat dilihat dibawah ini.



## B. KAJIAN PENELITIAN TERDAHULU

**Tabel 2. 7** Kajian Penelitian Terdahulu

| No | Judul Penelitian | Metode | Persamaan | Perbedaan |
|---|---|---|---|---|
| 1. | Ulla Delfana Rosiani, Mustika Mentari, Andi Novan Prastya (2019), Klasifikasi Kualitas Biji Jagung Berdasarkan Deteksi Warna dan Bentuk Menggunakan Metode K-Nearest Neighbor | K-Nearest Neighbor | - Menggunakan Bahasa Pemprograman Matlab<br>- Menggunakan deteksi warna | - Klasifikasi biji jagung meliputi biji jagung pecah,rusak dan utuh<br>- Penelitian ini semua objek jagung<br>- Peneliti juga berfokus pada bibit unggul yang ditanam |

| | | | | |
|---|---|---|---|---|
| 2. | Ri Munarto, Endi Permata, Rensi Salsabilla (2014), Klasifikasi Kualitas Biji Jagung Manis Berdasarkan Fitur Warna Menggunakan *Fuzzy Logic* | Fuzzy Logic | - Aplikasi yang di buat berupa Kualitas Jagung<br>- Mengunakan fitur warna<br>- Objek penelitian biji jagung manis | - Metode yang digunakan Fuzzy Logic<br>- Klasifikasi biji jagung manis meliputi biji jagung pecah,rusak dan utuh |
| 3. | Muhammad Imron Rosadi, Moch. Lutfy(2021), Identifikasi Jenis Penyakit Daun Jagung Menggunakan Deep Learning Pre-Trained Model | Deep Learning Pre-Trained Model | - Desain Sistem<br>- Mengunakan Bahasa pemrograman Matlab<br>- Identifikasi lewat fitur Warna | - Menggunaka algoritma Convolutionl Neural Network & Deep Learning Pre-Trained Model.<br>- Penelitian ini berfokus padan penyakit Daun Jagung |

# BAB III

# METODE PENELITIAN

## A. Jenis Penelitian

Jenis penelitian ini yang saya gunakan adalah studi literatur dan pengumpulan data

## B. Waktu Dan Tempat Penelitian

Kegiatan penelitian ini dilakukan di Pinrang Sulawesi Selatan, waktu yang dipergunakan untuk pelaksanaan ini ± 3 bulan di tahun 2024

## C. Alat dan Bahan Penlitian

Dalam melakukan penelitian, maka diperlukan alat dan bahan penelitian yang mendukung kegiatan penelitian tersebut. Alat dan bahan yang di perlukan antara lain :

1. **Perangkat Keras *(Hardware)***

   Perangkat keras yang digunakan selama proses penelitian yaitu Laptop Lenovo Thinkpad X230 dengan spesifikasi berikut :

   a. Processor : Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz 2.60 GHz
   b. RAM : 4,00 GB
   c. Hardisk : 500 GB
   d. Monitor : 12 Inch

**2. Perangkat Lunak *(Sofware)***

Perangkat lunak yang digunakan dalam pembuatan aplikasi adalah sebagai berikut :

a. Sistem Operasi : Windows 11 Pro 64 bit

b. Editor : Visual Studio Code

c. Bahasa Pemrograman : Python

## D. Tahapan Penelitian

Tahap – tahap penelitian yang dimaksud dalam penelitian ini adalah berkenan dengan proses pelaksanaan penelitian yaitu :

1. Persiapan Penelitian.

   Tahap persiapan adalah tahap yang dilakukan sebelum melakukan penelitian. Pada tahapan ini dimulai dengan mengkaji permasalahan yang telah ada kemudian melakukan studi literatul mengenai permasalahan yang sedang diteliti.

2. Pengumpulan Data

   Pada tahap ini peneliti melakukan observasi dengan peninjauan, pencatatan dan pengamatan langsung di tempat penelitian.

3. Analisis

   Pada tahap analisis, peneliti melakukan analisis terhadap sistem yang di terapkan sekarang berdasarkan kemudian merumuskan masalah yang menajdi pokok penelitian sehingga dapat dibuat *alternatif* pemecahan masalah.

4. Perancangan

   Peneliti kemudian merancang aplikasi yang ingin dibuat berdasarkan *alternatif* pemecahan masalah.

5. Pengujian

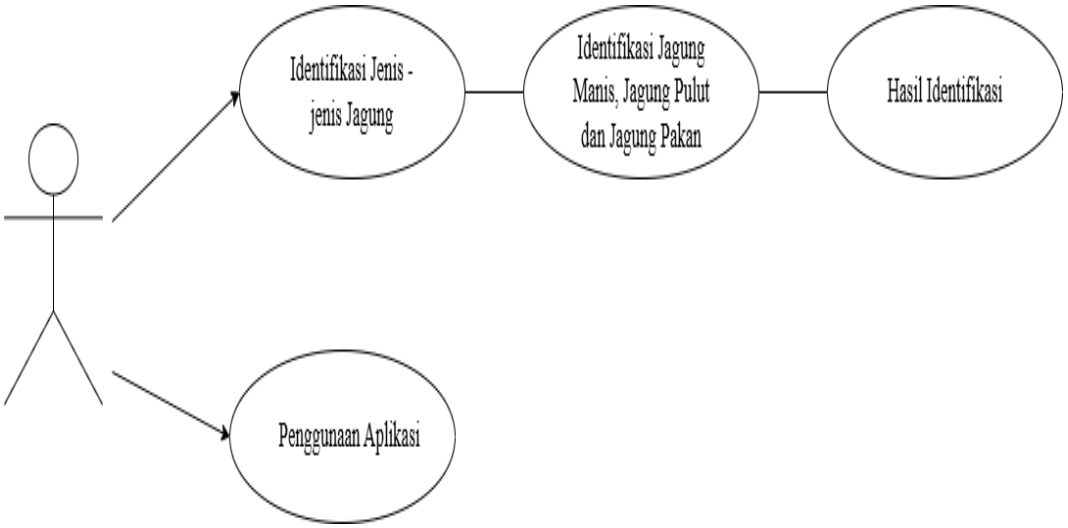
   Setelah melakukan pemecahan, peneliti kemudian menguji hasil pemecahan yang telah dibuat. Jika hasil perancangan terdapat kekurangan atau kelemahan maka kembali ke tahap analisis.

6. Implementasi

   Setelah pada perancangan tidak terdapat kekurangan maka aplikasi siap untuk di gunakan oleh *user*.

## E. Desain Sistem

Adapun desain sistem yang berjalan :



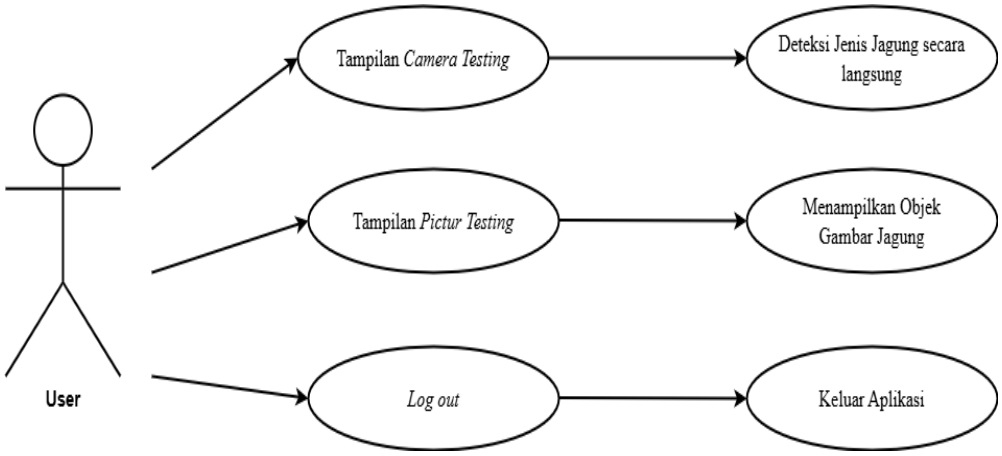

**Gambar 3. 1** Desain Sistem

# BAB IV

# HASIL DAN PEMBAHASAN

## A. Analisis Aliran Data dengan UML

### 1. *Use Case* Diagram




**Gambar 4. 1** Use Case Diagram User

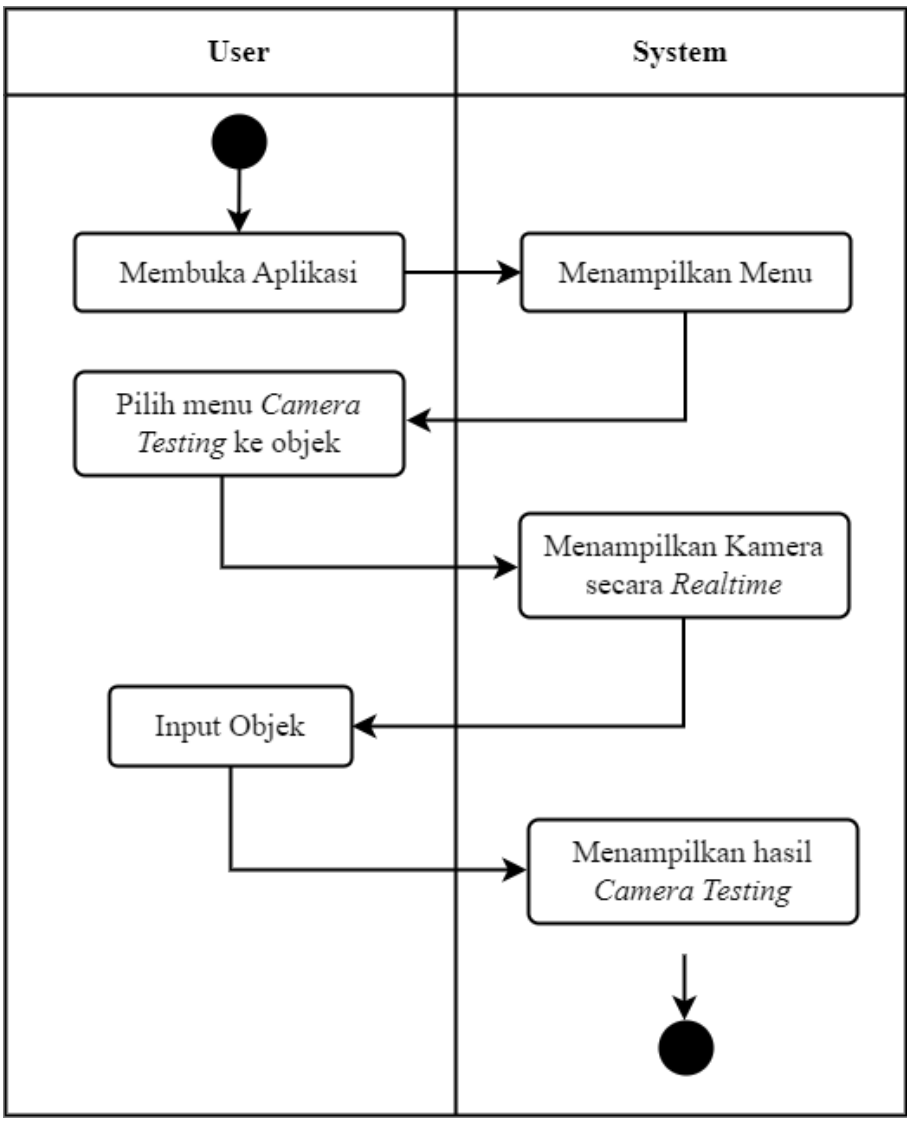
**Tabel 4. 1** Keterangan Use Case User

| **Nama *Use Case*** | **Deskripsi *Use Case*** |
|---|---|
| Tampilan *Camera Testing* | Yang menjelaskan bagaimana proses deteksi jagung secara langsung |
| Tampilan *Picture Testing* | Yang menjelaskan bagaimana proses deteksi objek jagung lewat gambar |
| *Logout* | Keluar Aplikasi |

**2. *Activity Diagram***

Activity Diagram ini menjelaskan tentang aktivitas-aktivitas yang terjadi dalam sebuah aliran proses pada sebuah sistem. Activity ini menjelaskan mengenai proses urutan yang akan dilalui oleh user dalam menjalankan sebuah aplikasi yang mulai dari halaman utama yang terdiri dari beberapa menu yaitu menu *Camera Testing ke* objek, *Picture Testing ke* gambar, dan *Logout.*

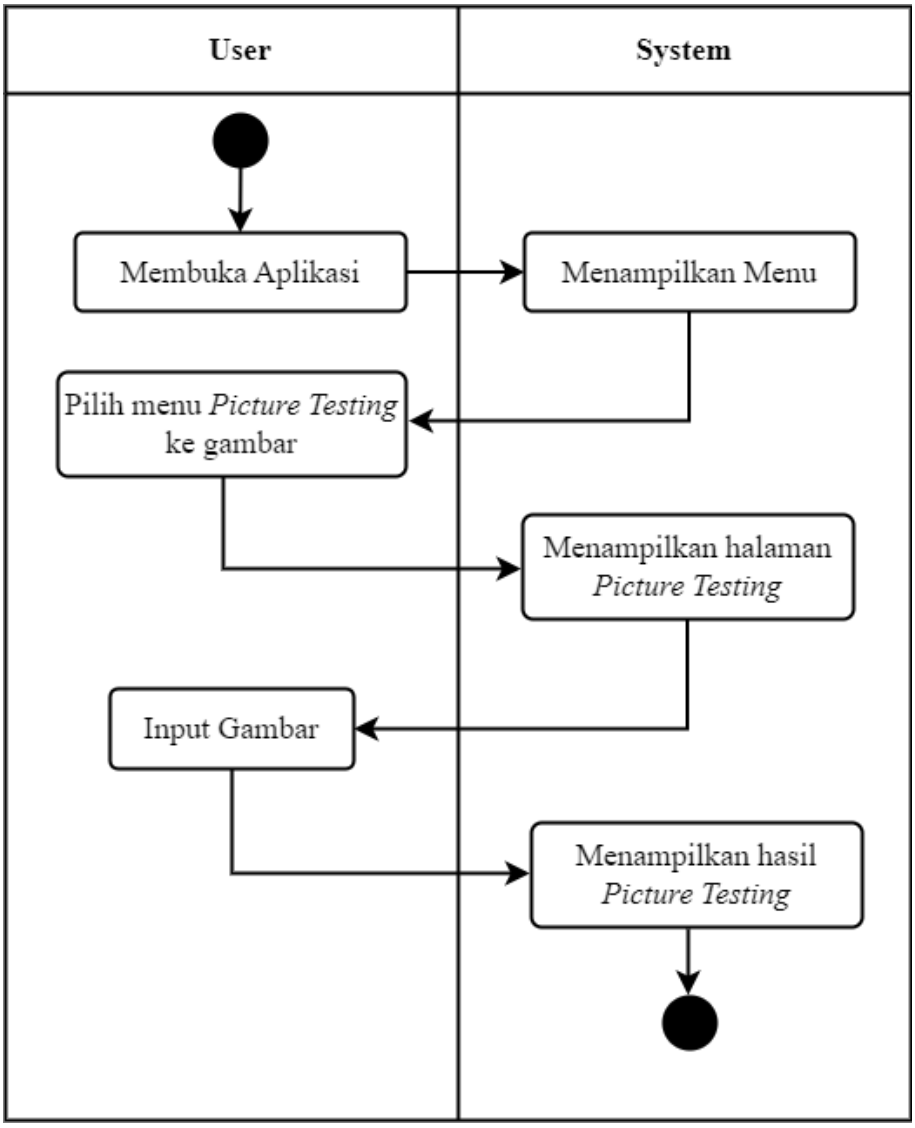
a. Activity diagram Camera Testing ke objek




**Gambar 4. 2** Activity Diagram Camera Testing

*Activity* diagram di atas menjelaskan bahwa jika *user* membuka aplikasi ini akan menampilkan beberapa menu kemudian *user* memilih menu *Camera Testing* setelah itu *user* menampilkan kamera secara *Realtime* setelah itu user menginput objek untuk deteksi jagung secara langsung dan maka menampilkan hasil dari *Camera Testing*.

b. Activity diagram *Picture Testing* ke gambar



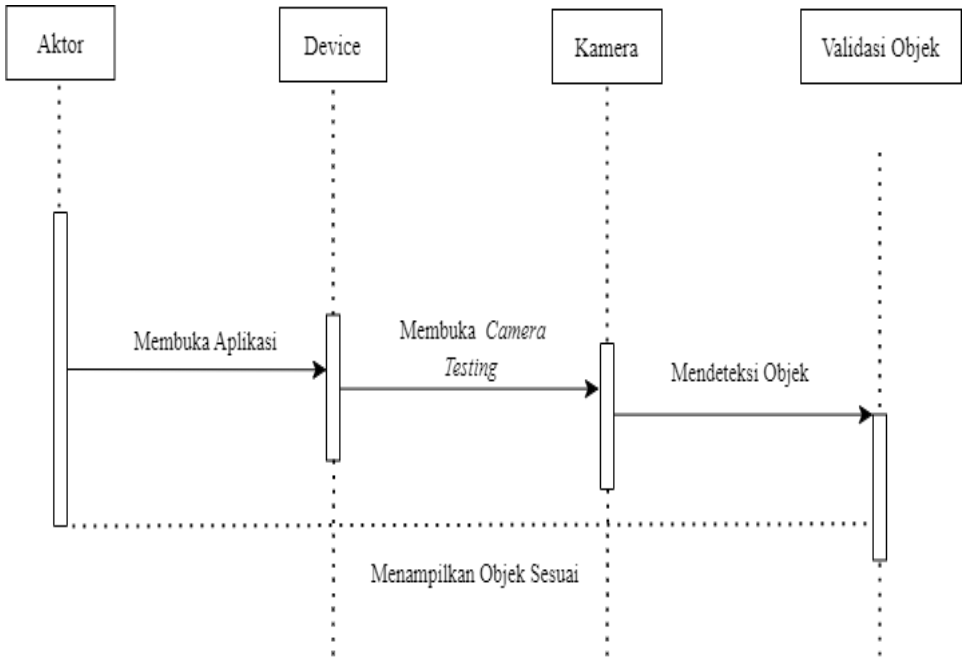

**Gambar 4. 3** Activity diagram Picture

*Activity* diagram di atas menjelaskan bahwa jika *user* membuka aplikasi ini akan menampilkan beberapa menu kemudian *user* memilih menu *Picture Testing* ke gambar setelah itu *user* setelah itu menampilkan halam Picture Testing kemudian *user* menginput gambar dari galeri maka menampilkan hasil dari *Picture Testing*

### 3. *Sequence Diagram*

*Sequence Diagram* adalah salah satu jenis diagram interaksi dalam UML (*Unified Modeling Language*) yang digunakan untuk menggambarkan interaksi antara objek atau komponen dalam sebuah sistem secara kronologis. Diagram ini mengilustrasikan alur pesan atau peristiwa di antara berbagai elemen sistem dalam rentang waktu tertentu.

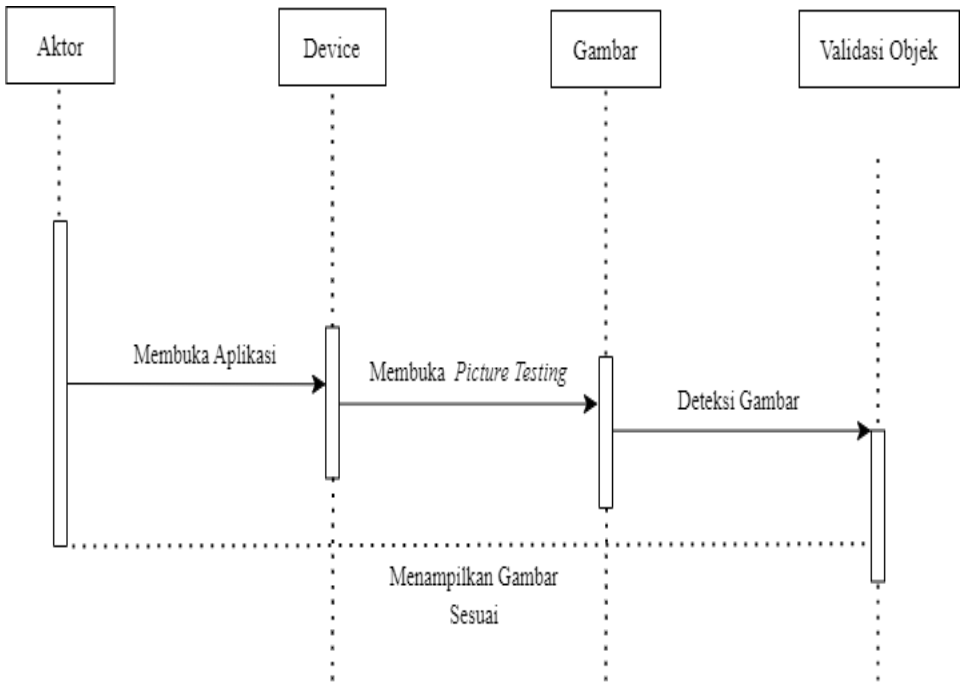
a. *Sequence* diagram *Camera Testing* ke objek




**Gambar 4. 4** Sequnce diagram Camera Testing

*Diagram Sequence* di atas menjelaskan bahwa ketika *user* memulai Aplikasi maka *device Android* yang di gunakan akan meload data untuk selanjutnya memulai aplikasi ini, setelah itu akan tampil menu *Camera Testing* ke objek, setelah itu *user* menginput *image from to gallery*, *Python* mengidentifikasi pada validasi objek maka akan tampil objek yang sesuai.

b. *Sequence* diagram *Picture Testing* ke gambar



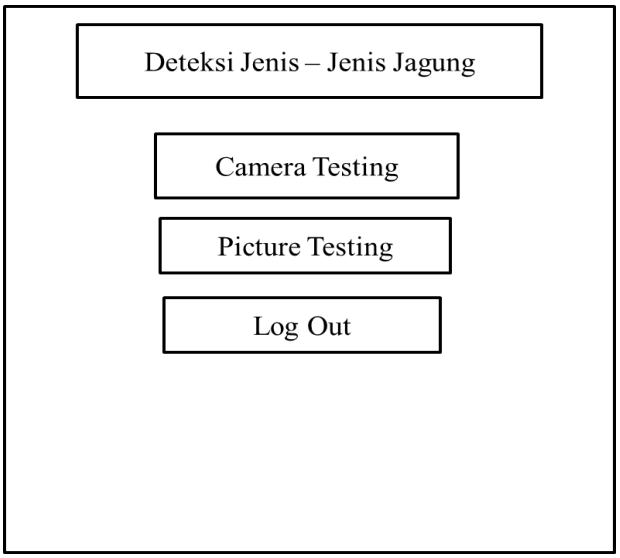

**Gambar 4. 5** Sequnce diagram Picture Testing

*Activity diagram* di atas menjelaskan bahwa ketika *user* membuka aplikasi ini akan menampilkan menu kemudian user memilih menu *picture testing* setelah itu menampilkan halaman test gambar kemudian user mengimput gambar dari galeri dan makan akan menampilkan hasil dari test gambar.

## B. Rancangan Sistem

1. Rancangan Tampilan Menu

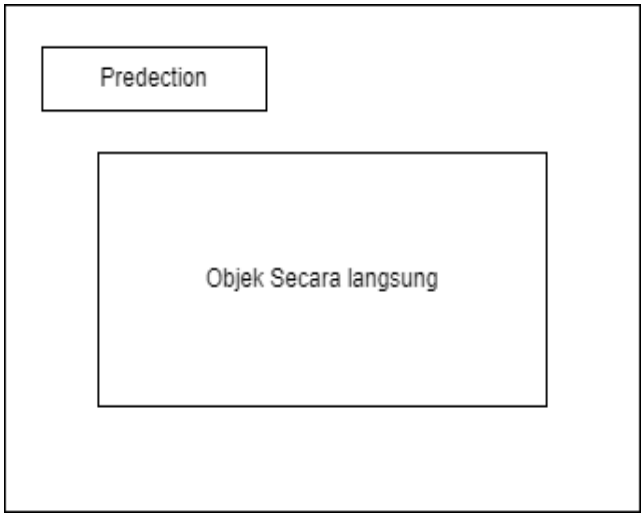
   Merupakan Tampilan Menu pada aplikasi Deteksi Jenis – jenis Jagung.




**Gambar 4. 6** Rancangan Tampilan Menu

2. Rancangan Tampilan Camera Testing

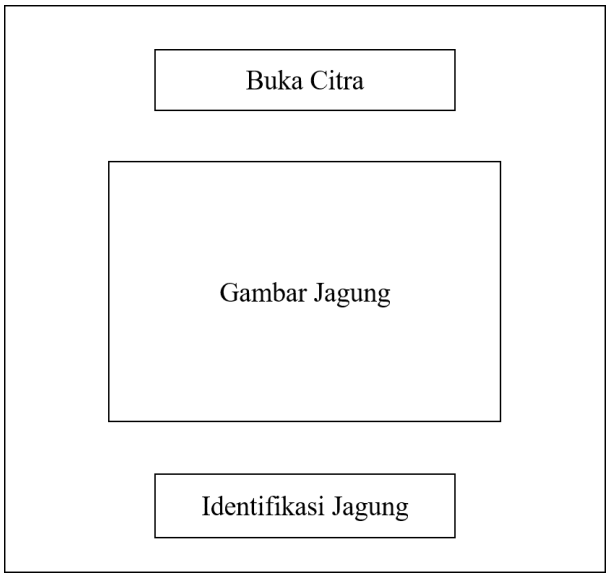
   Merupakan Rancangan Camera Testing berupa prediksi Objek Secara langsung.




**Gambar 4. 7** Rancangan Tampilan Camera Testing

3. Rancangan Tampilan Picture Testing

   Merupakan Rancangan Tampilan Picture Testing berupa Prediksi jenis – jenis Jagung.



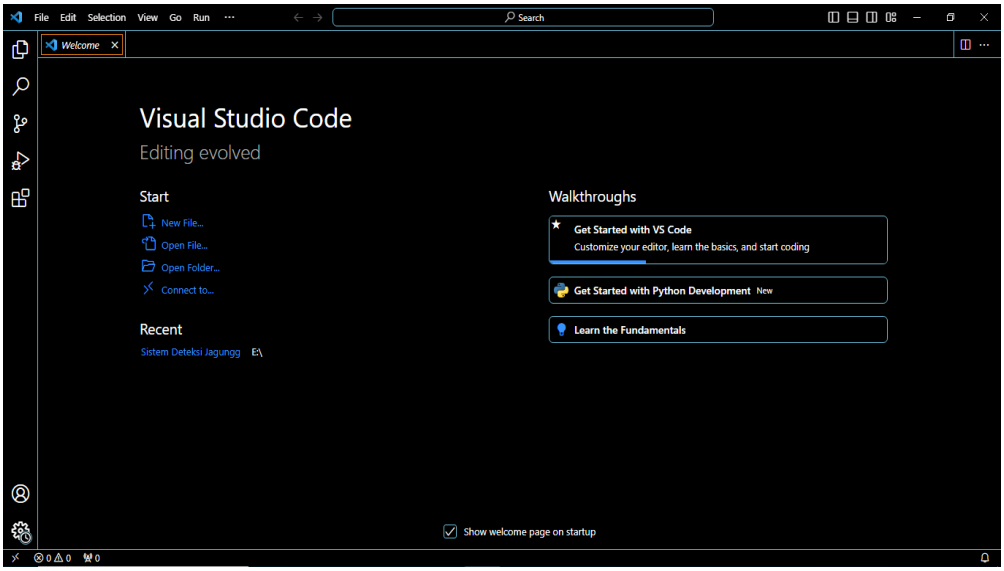

**Gambar 4. 8** Rancangan Tampilan Picture Testing

## C. Detail Sistem

1. Tampilan *Visual Studio*

   Tampilan Visual Studio Code (VS Code) terdiri dari beberapa komponen utama yang mendukung pengembangan aplikasi berbasis pemrosesan citra. *Explorer Sidebar* di sebelah kiri menampilkan struktur proyek, termasuk file *Python* seperti ekstraksiciriwarna.py untuk ekstraksi fitur warna dan realtimedeteksi.py untuk deteksi real-time dengan kamera. *Editor Window* di bagian tengah digunakan untuk menulis dan mengedit kode dengan fitur *syntax highlighting* serta *IntelliSense,* yang membantu dalam menulis kode lebih cepat dan efisien. *Terminal* bawaan di bagian bawah memungkinkan pengguna menjalankan skrip Python langsung tanpa perlu beralih ke aplikasi lain, seperti dengan mengetik

`python` identifikasi_jagung.py. Di bagian kanan, pengguna dapat membuka *Extensions Marketplace* untuk menambahkan pustaka pendukung seperti *Python, OpenCV, dan Matplotlib*. Selain itu, *Run and Debug Panel* menyediakan fitur debugging untuk mengidentifikasi kesalahan dalam algoritma pemrosesan citra. Dengan tampilan yang interaktif dan fleksibel, VS Code menjadi lingkungan ideal untuk mengembangkan sistem identifikasi jenis-jenis jagung berbasis pengolahan citra digital secara efisien dan terstruktur.




**Gambar 4. 9** Tampilan Visual Studio

2. Tampilan New File Visual Studio

Fitur *New File* memungkinkan pengguna membuat file baru yang diperlukan untuk mengembangkan proyek "Identifikasi Jenis-Jenis Jagung Berbasis Pengolahan Citra Digital". Saat pengguna mengklik *File > New File* atau menekan Ctrl + N, editor akan membuka tab kosong tempat pengguna dapat menulis kode Python untuk memproses citra jagung menggunakan pustaka

seperti *OpenCV, NumPy, dan Matplotlib*. Setelah menulis kode, pengguna dapat menyimpan file dengan Ctrl + S dan memberikan nama yang sesuai, misalnya identifikasi_jagung.py, untuk mengorganisir proyek dengan baik. File ini kemudian dapat dikelola dalam *Explorer Sidebar*, yang menampilkan struktur folder proyek, seperti dataset gambar jagung, skrip ekstraksi fitur (ekstraksiciriwarna.py), dan skrip deteksi real-time (*realtime*deteksi.py). Dengan fitur tambahan seperti *syntax highlighting, IntelliSense,* dan *integrasi Git,* VS Code memastikan pengembangan sistem identifikasi citra jagung berjalan lebih efisien dan terorganisir.




**Gambar 4. 10** Tampilan New File Visual Studio

3. Tampilan Menu Utama

Tampilan Menu Utama dalam sistem ini mencakup tiga fitur utama: *Camera Testing, Picture Testing*, dan *Log Out*, yang dirancang untuk memberikan pengalaman interaktif bagi pengguna dalam menguji model klasifikasi jagung. *Camera Testing* memungkinkan pengguna menangkap video real-time dari kamera, di mana setiap frame yang diambil diproses, diklasifikasikan, dan hasil

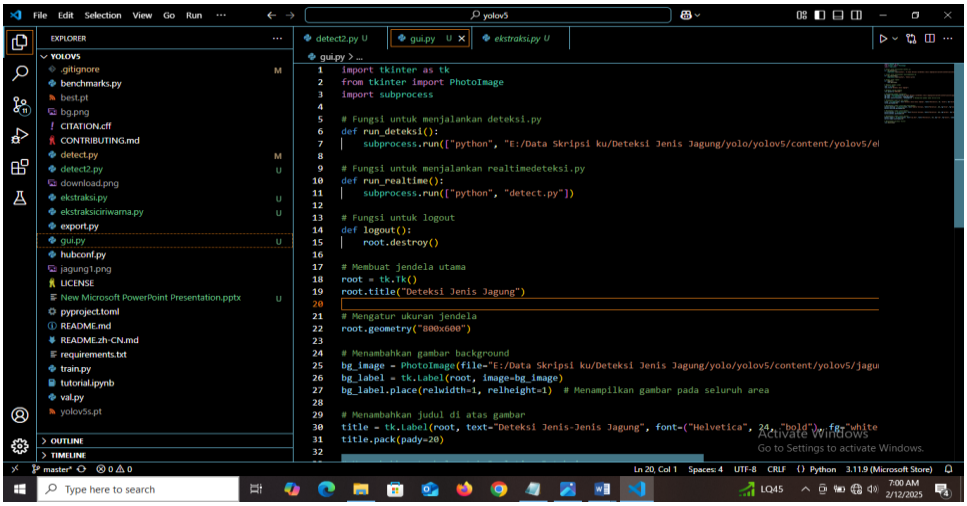
prediksi ditampilkan langsung di layar. *Picture Testing* memungkinkan pengguna mengunggah gambar statis yang kemudian diproses menggunakan model yang sama, menghasilkan prediksi tanpa perlu menggunakan kamera langsung. Kedua fitur ini memberikan fleksibilitas dalam pengujian model, baik melalui input langsung dari kamera maupun gambar yang telah disimpan sebelumnya. Sementara itu, opsi *Log Out* berfungsi untuk keluar dari sistem.




**Gambar 4. 11** Tampilan Menu Utama

Script Tampilan Menu Utama dalam sistem pengujian klasifikasi jagung berfungsi sebagai antarmuka awal yang memungkinkan pengguna memilih antara *Camera Testing, Picture Testing,* dan *Log Out*. Saat dijalankan, skrip akan menampilkan menu utama dengan opsi yang dapat dipilih oleh pengguna. Jika pengguna memilih *Camera Testing*, sistem akan mengaktifkan kamera, menangkap video secara *real-time*, dan melakukan klasifikasi setiap frame, lalu menampilkan hasil prediksi di layar. Jika pengguna memilih *Picture Testing*, sistem akan meminta pengguna untuk mengunggah gambar dari penyimpanan lokal, kemudian memproses dan mengklasifikasikan gambar tersebut dengan

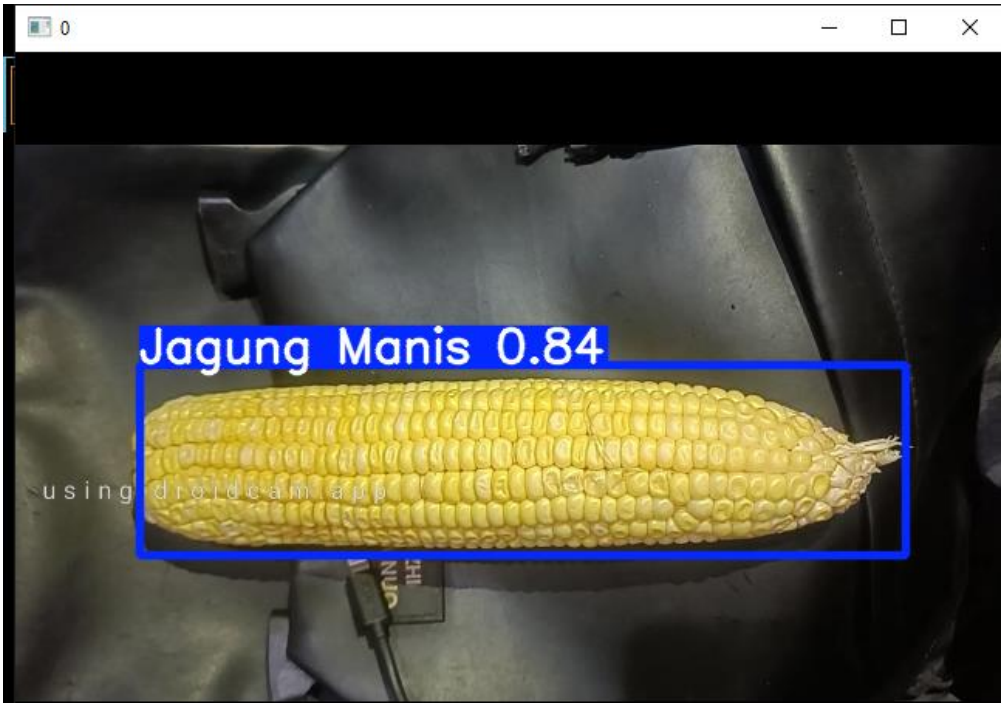
model yang sama. Pilihan *Log Out* berfungsi untuk keluar dari program. Script tampilan menu utama bertindak sebagai pusat navigasi system.




**Gambar 4. 12** Tampilan Scripct Menu Utama

4. Tampilan Menu *Camera Testing*

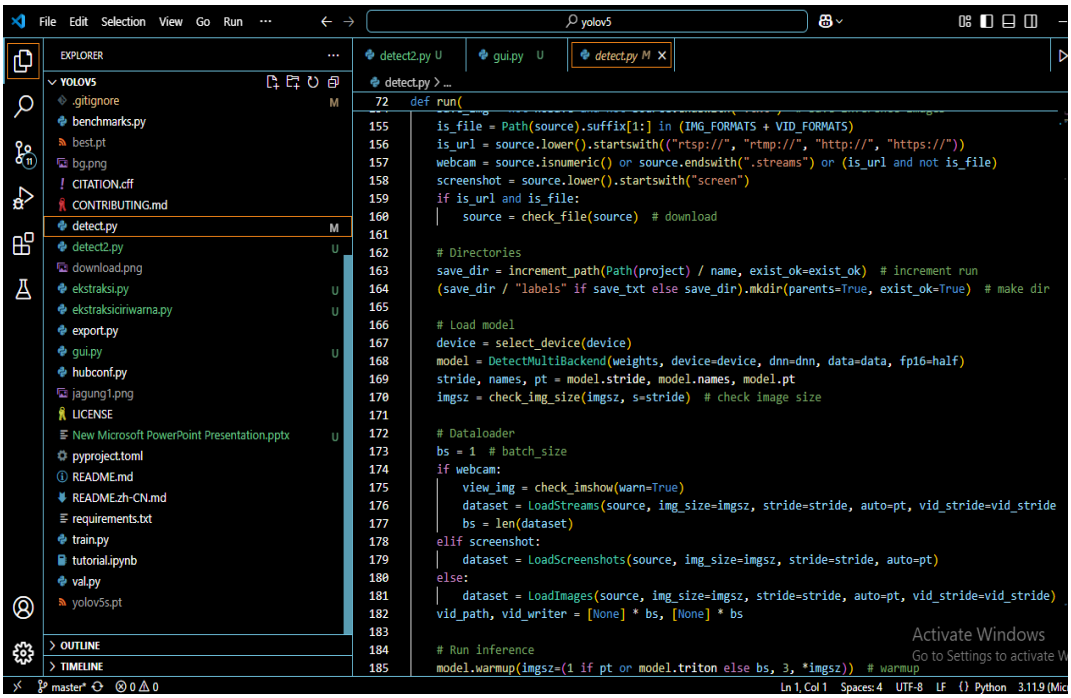
Tampilan menu *Camera Testing* dirancang untuk menguji dan mengolah citra jagung secara real-time menggunakan kamera. Saat tombol *"Camera Testing"* diklik dalam antarmuka aplikasi yang dibuat dengan *Tkinter*, sistem akan menjalankan skrip realtimedeteksi.py, yang membuka kamera menggunakan *OpenCV* (cv2.VideoCapture). Tampilan jendela kamera akan muncul, menampilkan umpan langsung (*live feed*) dari kamera yang digunakan untuk menangkap citra jagung. Dalam pemrosesan citra akan secara otomatis menganalisis warna dominan jagung dalam format *HSV (Hue, Saturation, Value)* untuk menentukan jenisnya, seperti jagung manis, jagung pakan, atau jagung pulut. Jika model deteksi lebih lanjut diterapkan, bounding box atau overlay teks dapat muncul dalam tampilan untuk memberikan informasi secara langsung.




**Gambar 4. 13** Tampilan Pengujian Jagung Manis

Skrip di menu camera testing dalam Visual Studio dengan menggunakan OpenCV untuk menangkap video dari kamera secara real-time dan mengklasifikasikan gambar yang telah dilatih sebelumnya. Saat skrip dijalankan melalui terminal VS Code, jendela baru akan muncul menampilkan feed langsung dari kamera. Hasil prediksi yang mengidentifikasi gambar sebagai salah satu dari tiga kelas jagung (Jagung Manis, Jagung Pulut, atau Jagung Pakan), akan ditampilkan langsung di atas video menggunakan teks berwarna hijau. Jendela ini akan terus diperbarui setiap frame untuk memberikan prediksi secara real-time. Jika pengguna ingin menghentikan

tampilan kamera, mereka dapat menekan tombol 'q' pada keyboard, yang akan menutup kamera dan menutup jendela tampilan.

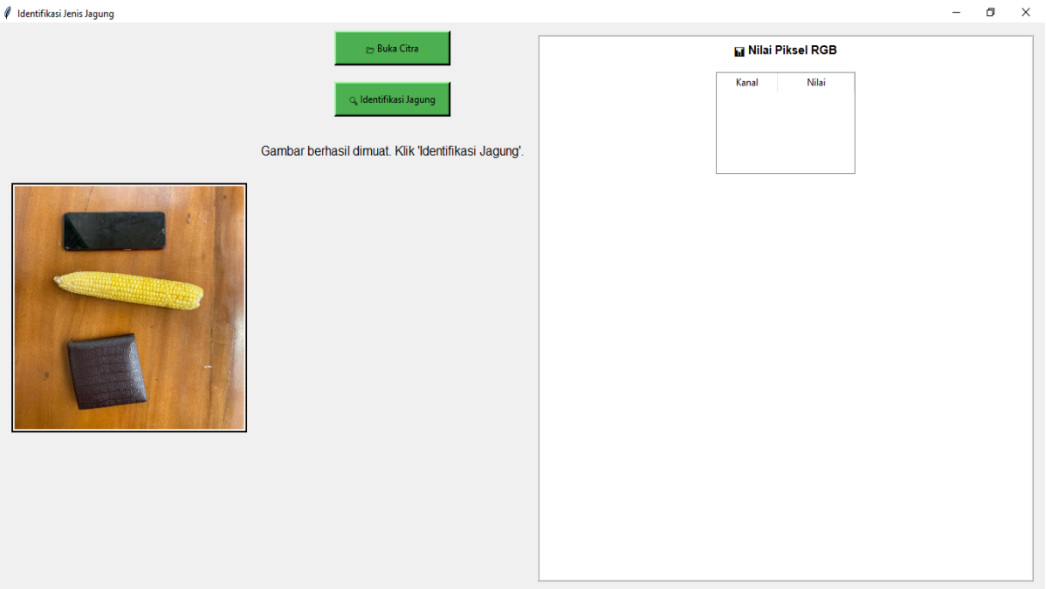



**Gambar 4. 14** Tampilan Script menu Camera Testing

5. Tampilan Menu Picture Testing

   a. Tampilan Buka Citra

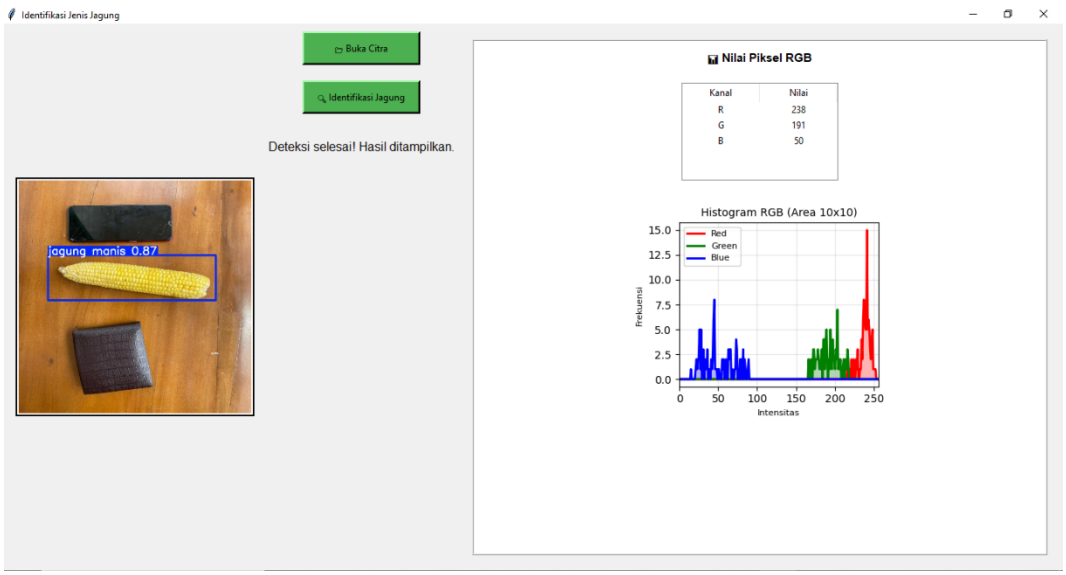
      Buka Citra berfungsi untuk membuka file gambar yang akan digunakan dalam proses identifikasi jenis jagung berbasis pengolahan citra digital. Saat tombol ini diklik, biasanya akan muncul jendela dialog untuk memilih gambar dari penyimpanan komputer. Setelah gambar dipilih, sistem akan nampilkannya di dalam antarmuka aplikasi sebagai input untuk tahap analisis lebih lanjut




**Gambar 4. 15** Tampilan Buka Citra

b. Tampilan Hasil Identifikasi

Hasil identifikasi dalam tampilan *Picture Testing* menampilkan klasifikasi gambar jagung yang diunggah oleh pengguna. Dalam mode ini, pengguna akan memilih atau mengunggah gambar jagung dari penyimpanan lokal, yang kemudian akan diproses dengan pra-pemrosesan gambar, normalisasi, dan konversi ke format yang sesuai sebelum dimasukkan ke dalam model. Setelah di identifikasi dilakukan, sistem akan menampilkan hasil identifikasi berupa label klasifikasi seperti "Jagung Manis", "Jagung Pulut", atau "Jagung Pakan". sistem dapat menampilkan gambar hasil unggahan bersama dengan label prediksi dalam bentuk teks di layar, bahkan bisa disertai dengan akurasi atau probabilitas prediksi.




**Gambar 4. 16** Tampilan hasil identifikasi Jagung Manis

## D. Implementasi

Implementasi *system* merupakan tahap penerapan dari suatu teknologi yang di desain untuk siap di operasikan. Tahap ini merupakan terjemahan perancangan dari bab hasil analisis sebelumnya dalam suatu Bahasa pemrograman. Bahasa pemrograman yang digunakan dalam membangun sebuah *Aplikasi* Deteksi Jenis - jenis Jagung berbasis Citra Digital menggunakan Bahasa pemrograman *Python.*

1. Kebutuhan Perangkat Keras

   Spesifikasi minimum perangkat keras sebagai berikut:

   **Tabel 4. 2 Kebutuhan Perangkat Keras**

| Jenis | Spesifikasi |
|---|---|
| *Laptop/PC* | Lenovo V130 – 14IKB |
| *Processor* | Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz<br>2.30 GHz |
| Memory | 4.00GB |

2. Kebutuhan Perangkat Lunak.

   Spesifikasi minimum perangkat lunak sebagai berikut:

   **Tabel 4. 3** Kebutuhan Perangkat Lunak

| **Jenis** | **Spesifikasi** |
|---|---|
| *Sistem Operasi* | *Windows 10* |
| *Text Editor* | *Visual Studio Code* |

## E. Pengujian Sistem

### *1. BlackBox*

Pengujian *system* dilakukan dengan cara pengujian *BlackBox*. *Blackox* adalah metode pengujian perangkat lunak yang menguji fungsionalitas aplikasi tanpa mengintip ke dalam struktur atau cara kerja internalnya. Metode pengujian ini dapat diterapkan secara *virtual* ke setiap tingkat pengujian perangkat lunak : unit, integrasi, *system*, dan penerimaan.

**Tabel 4. 4 Hasil BlackBox test**

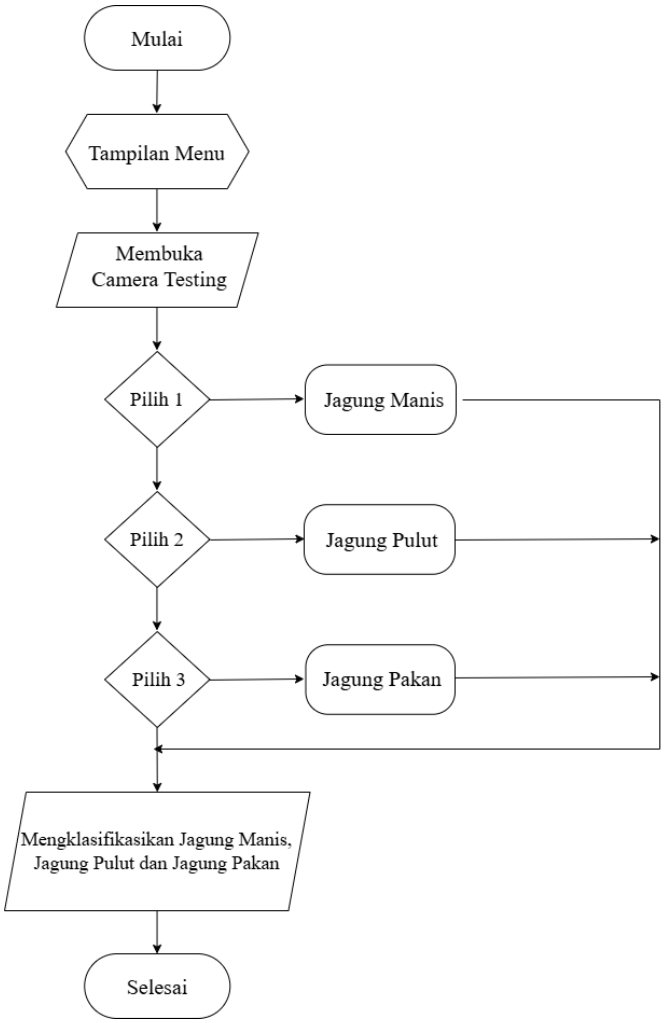
| Nama Pengujian | Tujuan | Skenario | Hasil Yang diharapkan | Hasil |
|---|---|---|---|---|
| Membuka Aplikas | Mengetahui aplikasi berjalan tanpa *error* | Mencoba Membuka aplikasi | Aplikasi Dapat berjalan dengan baik, Menampilkan menu utama | ✓ |
| Menu Utama | Mengakses menu pada aplikasi | Memilih menu | Menampilkan menu camera testing dan picture testing | ✓ |
| Menu *Camera Testing* | Untuk mendeteksi objek secara langsung | Memilih menu camera testing | Menampilkan hasil deteksi jenis – jenis jagung | ✓ |
| Menu *Picture testing* | Untuk mendeteksi gambar | Memilih Menu *Picture Testing* | Menampilkan Hasil deteksi berupa ekstraksi ciri warna dan hasil identifikasi | ✓ |

## 2. *WhiteBox*

Pengujian ini dilakukan dengan cara pengujian *WhiteBox*. *WhiteBox* adalah metode pengujian perangkat lunak yang menguji struktur internal atau cara kerja aplikasi yang bertentangan dengan fungsinya. Dalam pengujian kotak putih, perspektif internal sistem digunakan untuk merancang kasus uji.

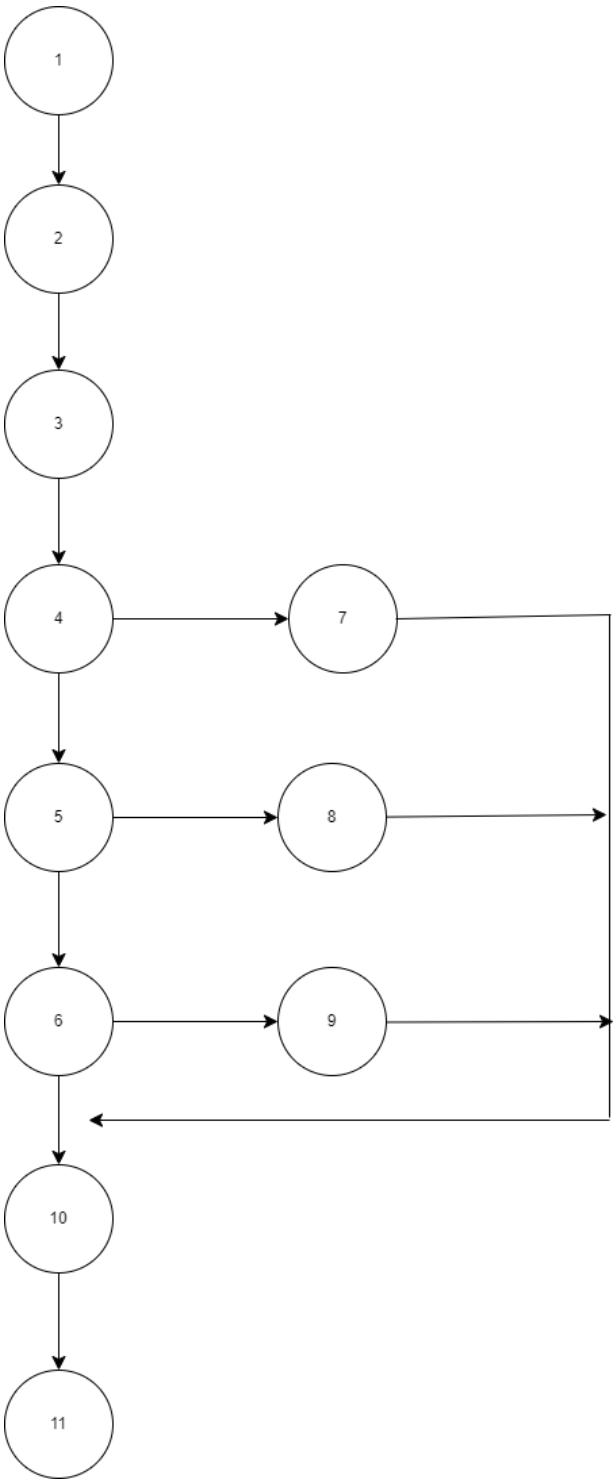
### a. *White Box Camera Testing*

1) *Flowchart*




**Gambar 4. 17** Flowchart Camera Testing

2) *Flowgraph*




**Gambar 4. 18** Flowgraph Camera Testing

Berdasarkan Gambar 4. 24 di atas dapat dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexcity* V(G) pada *egde* dan *node*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 3

N (*node*) = 4

P (Predikat node) = 0

Penyelesaian :

V(G) = E – N + 2

= 3 – 4 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

=1

(2) Berdasarkan perhitungan *Cyclomatic Complexcity* dari *flowgraph* di atas memiliki *Region* = 1

(3) *Independent path* pada *flowgraph* tersebut yakni:

Path 1 = 1 – 2 – 3 – 4 – 8 – 11

Path 2 = 1 – 2 – 3 – 5 – 9 – 11

Path 3 = 1 – 2 – 3 – 5 – 6 – 10 – 11

Path 4 = 1 – 2 – 3 – 5 – 6 – 7 – 11

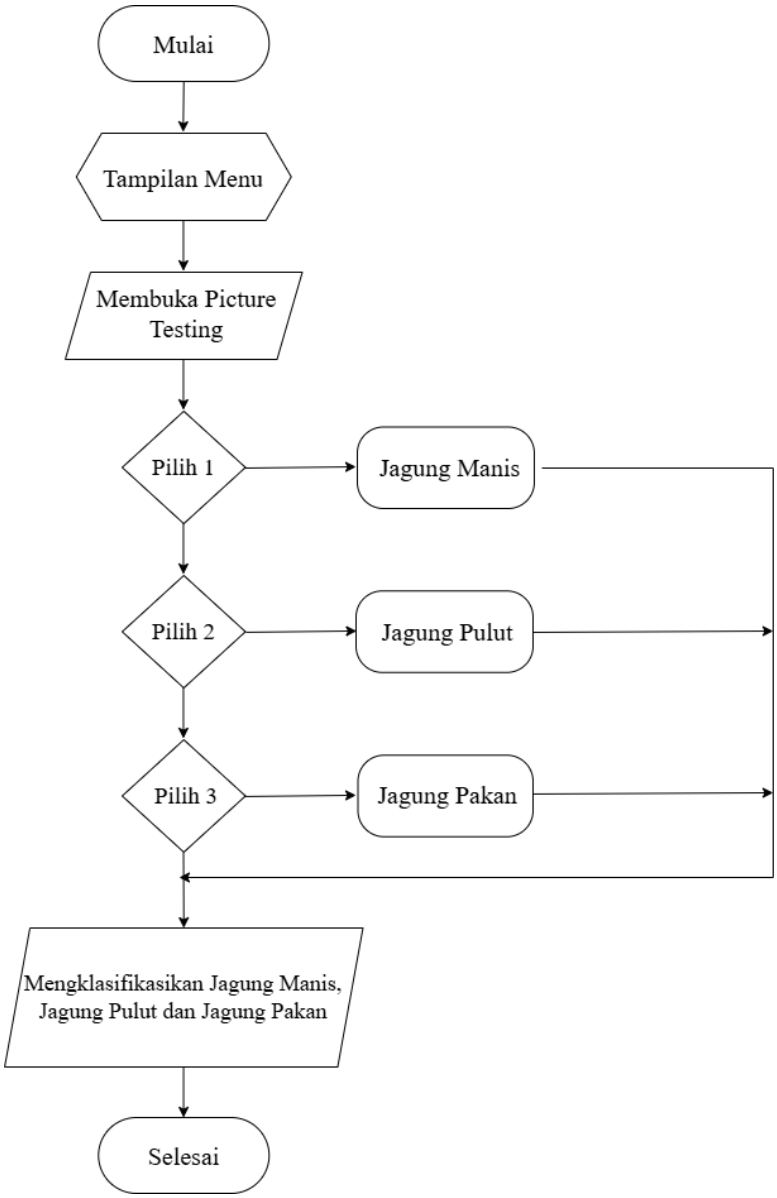
(4) Grafik matriks halaman menu Picture Testing

**Tabel 4. 5** *Grafik matriks Camera Testing*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | E-1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | 1 | | | | | | | | | | 1 – 1 = 0 |
| 2 | | | 1 | | | | | | | | | 1 – 1 = 0 |
| 3 | | | | 1 | 1 | | | | | | | 2 – 1 = 1 |
| 4 | | | | | | | | 1 | | | | 1 – 1 = 0 |
| 5 | | | | | | 1 | | | 1 | | | 2 – 1 = 1 |
| 6 | | | | | | | 1 | | | | | 1 – 1 = 1 |
| 7 | | | | | | | | | | 1 | | 1 – 1 = 0 |
| 8 | | | | | | | | | | 1 | | 1 – 0 = 0 |
| 9 | | | | | | | | | | 1 | | 1 – 0 = 0 |
| 10 | | | | | | | | | | 1 | | 1 – 0 = 0 |
| 11 | | | | | | | | | | | | 0 |
| **SUM ( E + 1 )** | | | | | | | | | | | | 4 + 1 = 5 |

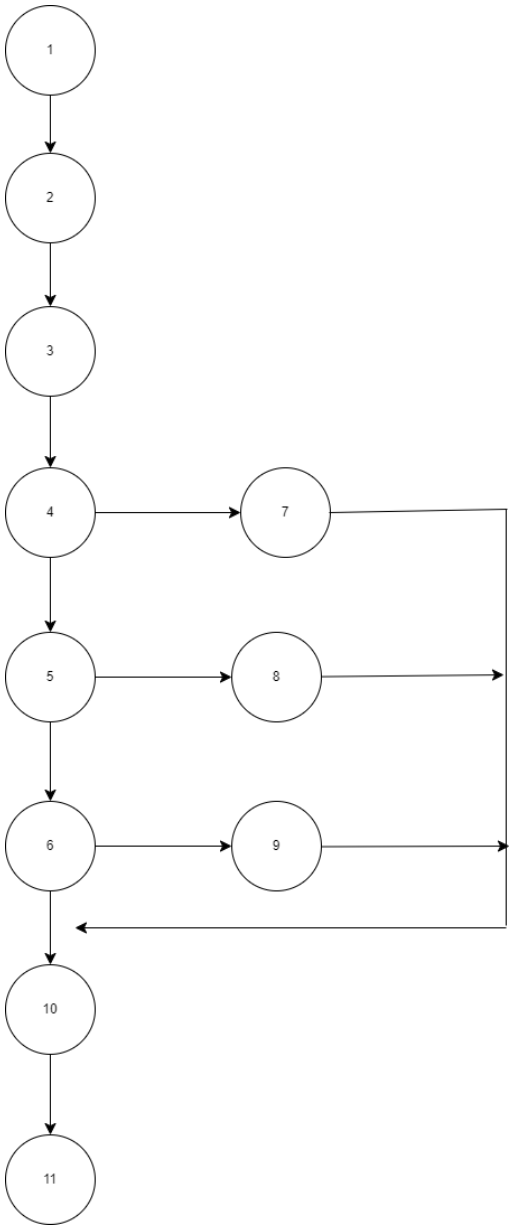
*b. White Box Picture Testing*

1) *Flowchart*




**Gambar 4. 19** Flowchart Picture Testing

*2) Flowgraph*




**Gambar 4. 20 Flowgraph**

Berdasarkan Gambar 4.20 di atas dapat dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexcity* V(G) pada *egde* dan *node* Pada rumus : V(G) = E – N + 2

E (*edge*) = 3

N (*node*) = 4

P (Predikat node) = 0

Penyelesaian :

V(G) = E – N + 2

= 3 – 4 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

=1

(2) Berdasarkan perhitungan *Cyclomatic Complexcity* dari *flowgraph* di atas memiliki *Region* = 1

(3) *Independent path* pada *flowgraph* tersebut yakni:

Path 1 = 1 – 2 – 3 – 4 – 8 – 11

Path 2 = 1 – 2 – 3 – 5 – 9 – 11

Path 3 = 1 – 2 – 3 – 5 – 6 – 10 – 11

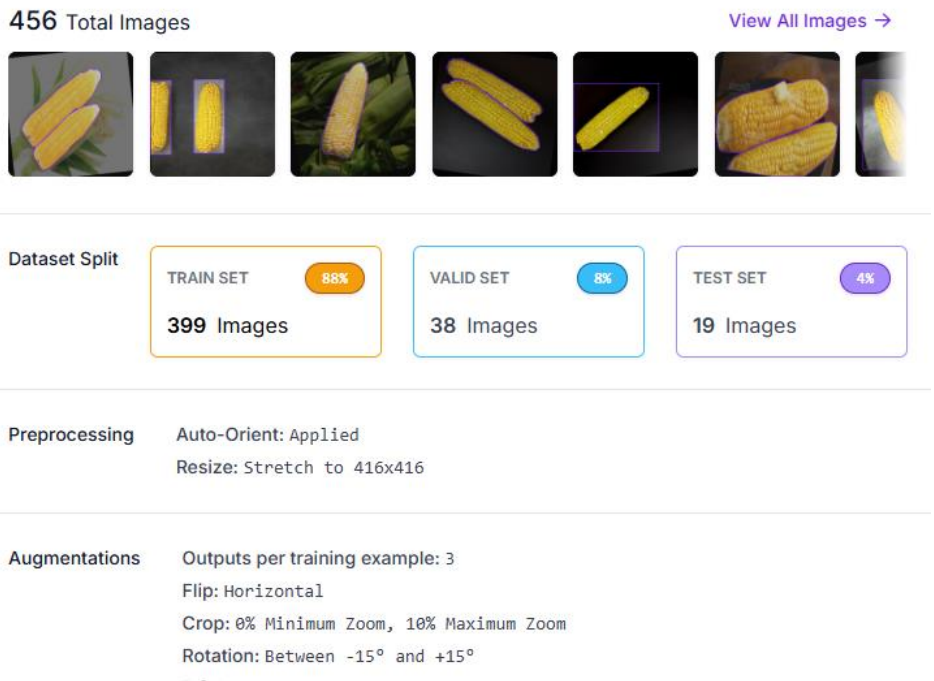
Path 4 = 1 – 2 – 3 – 5 – 6 – 7 – 11

(4) Grafik matriks halaman menu Picture Testing

**Tabel 4. 6** Grafik matriks Picture Testing

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | E-1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | 1 | | | | | | | | | | 1 – 1 = 0 |
| 2 | | | 1 | | | | | | | | | 1 – 1 = 0 |
| 3 | | | | 1 | 1 | | | | | | | 2 – 1 = 1 |
| 4 | | | | | | | | 1 | | | | 1 – 1 = 0 |
| 5 | | | | | | 1 | | | 1 | | | 2 – 1 = 1 |
| 6 | | | | | | | 1 | | | | | 1 – 1 = 1 |
| 7 | | | | | | | | | | 1 | | 1 – 1 = 0 |
| 8 | | | | | | | | | | 1 | | 1 – 0 = 0 |
| 9 | | | | | | | | | | 1 | | 1 – 0 = 0 |
| 10 | | | | | | | | | | 1 | | 1 – 0 = 0 |
| 11 | | | | | | | | | | | | 0 |
| **SUM ( E + 1 )** | | | | | | | | | | | | 4 + 1 = 5 |

## 3. Hasil Labeling Dataset

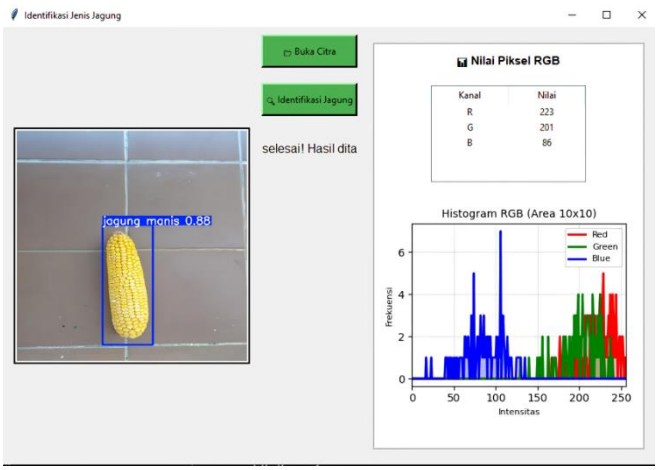
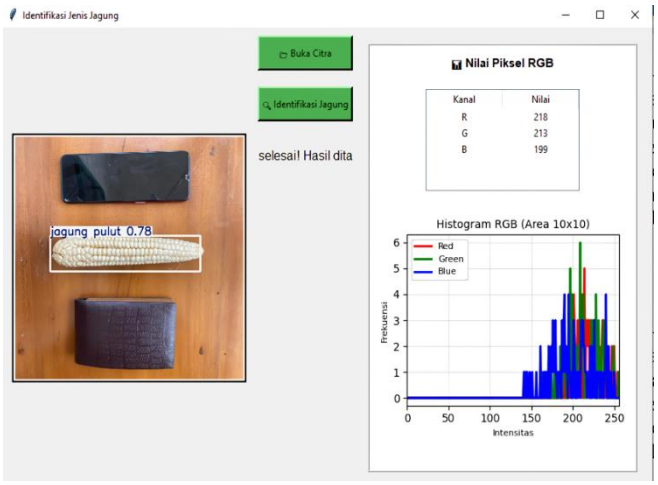
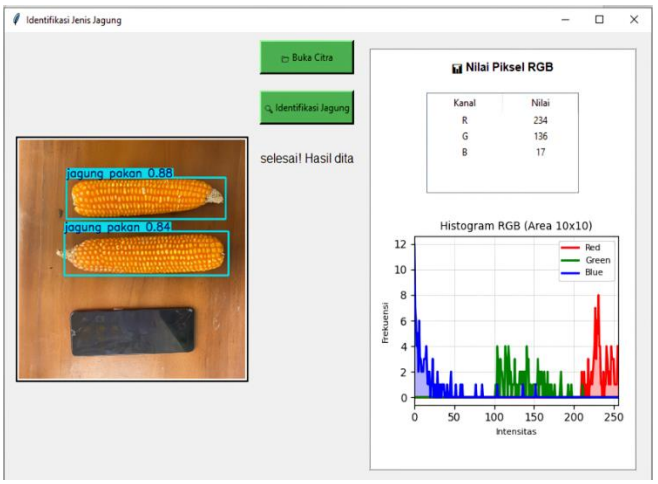


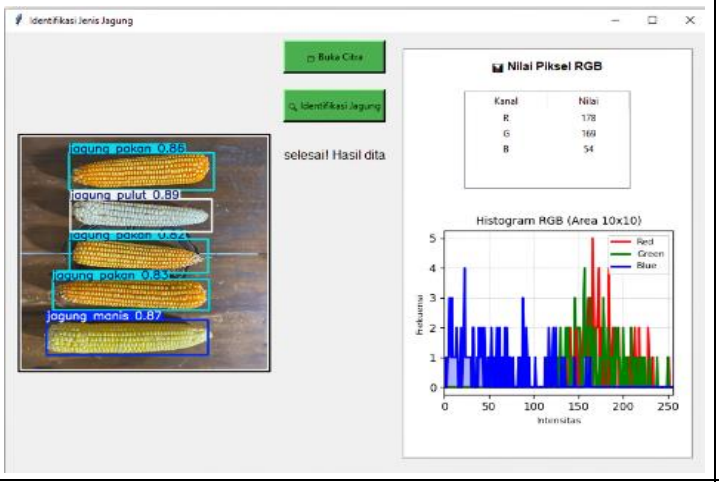
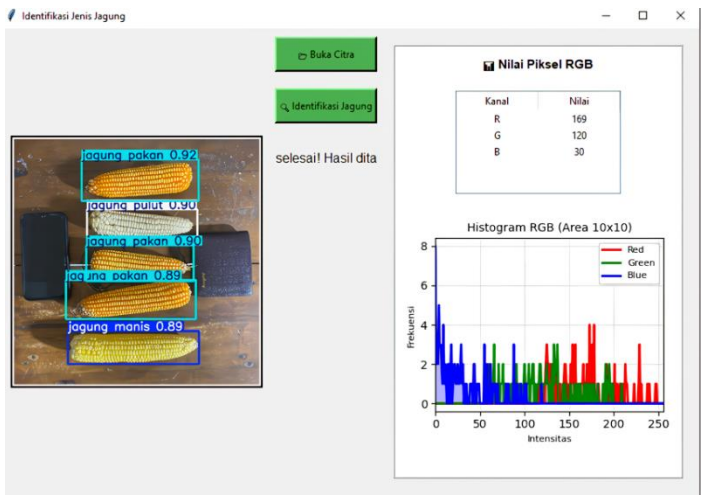
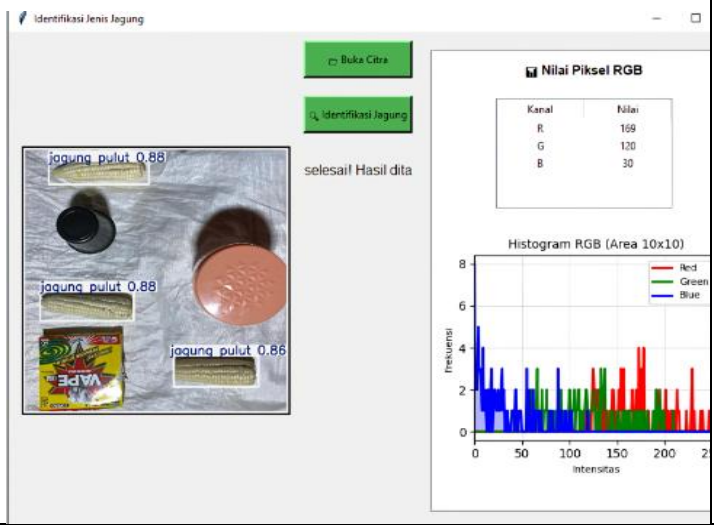

**Gambar 4. 21** Hasil Labeling Dataset

Gambar di atas menunjukkan rincian dataset citra jagung sebanyak 456 gambar yang telah dibagi menjadi tiga bagian, yaitu 399 gambar (88%) untuk pelatihan (train set), 38 gambar (8%) untuk validasi (valid set), dan 19 gambar (4%) untuk pengujian (test set). Dataset ini telah melalui tahap preprocessing dengan auto-orient yang diterapkan dan ukuran gambar yang disesuaikan (stretch) menjadi 416x416 piksel agar sesuai dengan input model YOLO. Selain itu, dilakukan augmentasi sebanyak 3 output per gambar pelatihan dengan teknik flip horizontal, zoom maksimal 10%, serta rotasi antara -15° hingga +15°, yang bertujuan untuk meningkatkan keberagaman data sehingga model dapat mengenali objek jagung secara lebih akurat dalam berbagai kondisi. Struktur dan perlakuan ini menunjukkan bahwa dataset sudah disiapkan dengan benar dan siap digunakan untuk proses pelatihan model deteksi objek berbasis YOLO.

## 4. Hasil Pengujian Kehandalan Input

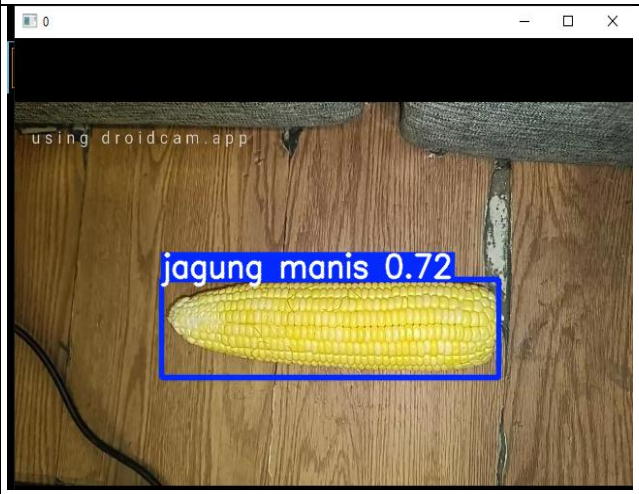
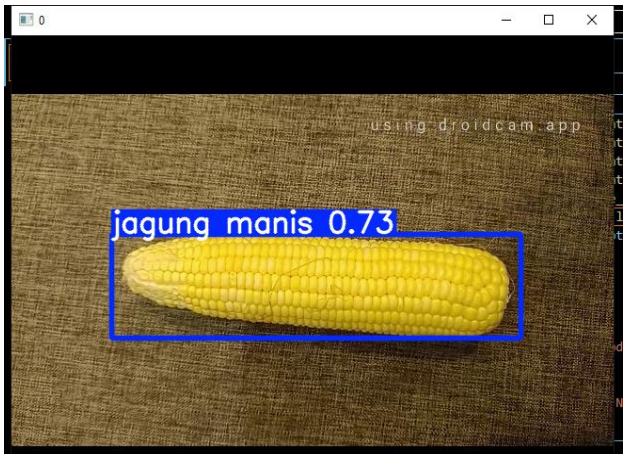
**Tabel 4. 7** Hasil Pengujian Kehandalan Input

| No. | Gambar | Keterangan |
|---|---|---|
| 1. |  | Terdeteksi secara akurat dengan tabel nilai pixel dan histogram |
| 2. |  | Jagung pulut terdeteksi dengan tambahan objek lain |
| 3. |  | Jagung pakan terdeteksi dengan akurat |

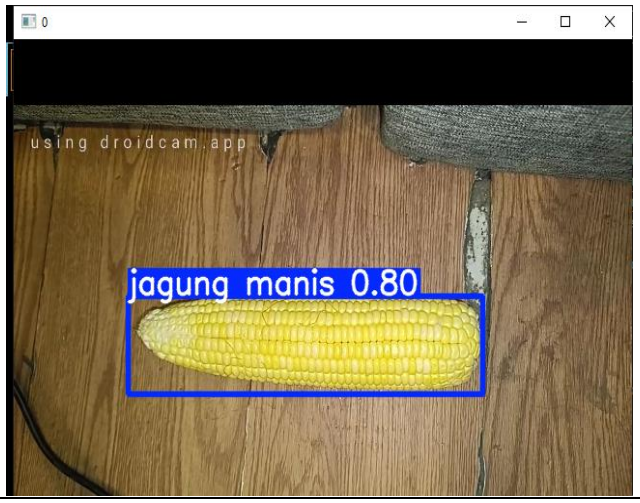
| No. | Gambar | Keterangan |
|---|---|---|
| 4. |  | Jenis-jenis jagung terdeteksi secara tepat dan akurat dengan background berwana coklat |
| 5. |  | Jenis objek Jagung terdeteksi secara akuran meskipun di tambahkan objek lain |
| 6. |  | Terdeteksi dengan tepat dari beberapa objek |

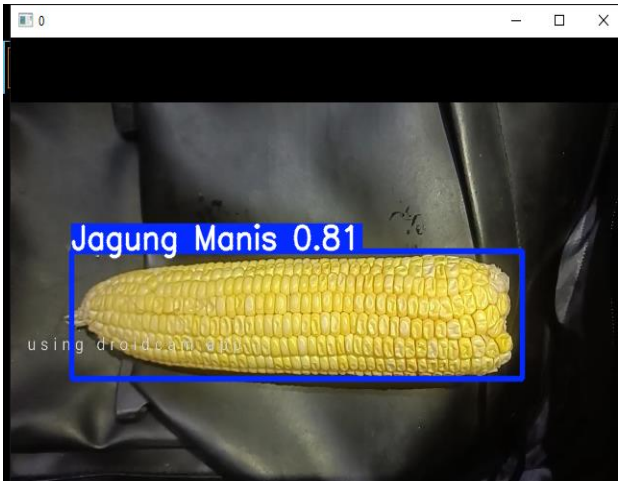
## 5. Hasil Pengujian Confidence Score

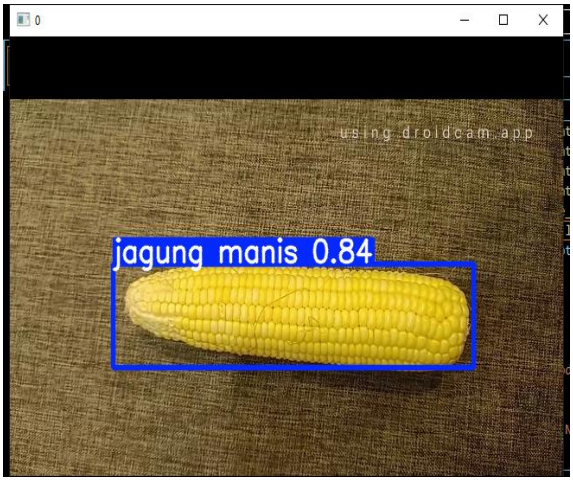
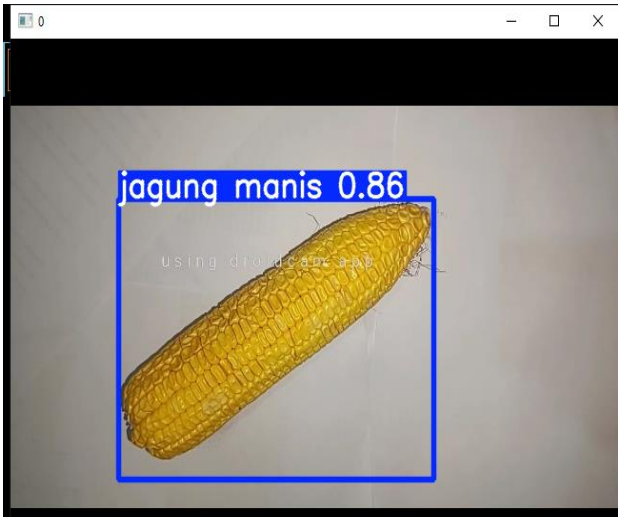
**Tabel 4. 6** Hasil Pengujian Confidence Score

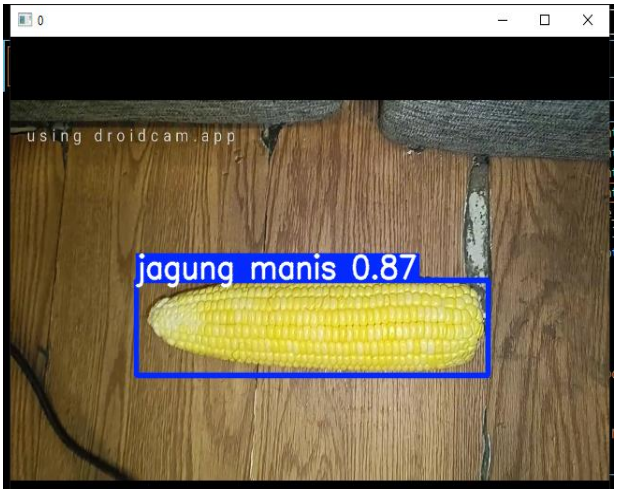
| No. | Gambar | Confidence Score |
|---|---|---|
| 1. | <br>Rumus :<br>objectness = 0.90<br>class object = 0.80<br>Hitung = 0.90 × 0.80<br>= (90/100) × (80/100)<br>= 7200/10000<br>= 0.72 | 72 % |
| 2. |  | 73% |

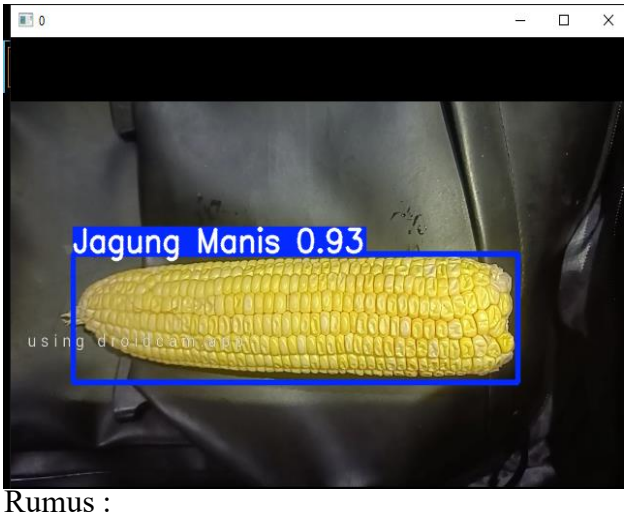
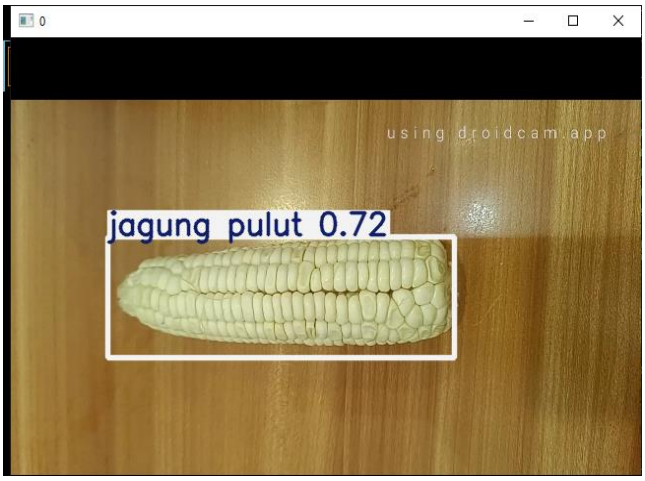
| No. | Gambar | Confidence Score |
|---|---|---|
| | Rumus :<br>objectness = 0.91<br>class object = 0.80<br>Hitung = 0.91 × 0.80<br>= (91/100) × (80/100)<br>= 7280/10000<br>= 0.7280<br>= 0.73 | |
| 3. | <br>Rumus :<br>objectness = 0.75<br>class object = 1.00<br>Hitung = 0.75 × 1.00<br>= (75/100) × (100/100)<br>= 7500/10000<br>= 0.75 | 75% |

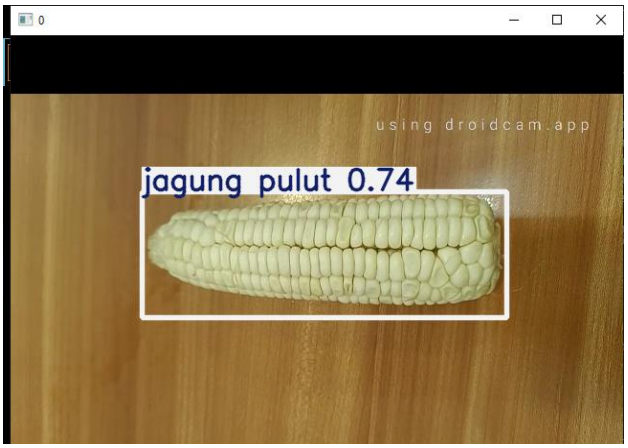
| No. | Gambar | Confidence Score |
|---|---|---|
| 4. | <br>Rumus :<br>objectness = 0.95<br>class object = 0.83<br>Hitung = 0.95 × 0.83<br>= (95/100) × (83/100)<br>= 7885/10000<br>= 0.7885<br>= 0.79 | 79% |
| 5. |  | 80% |

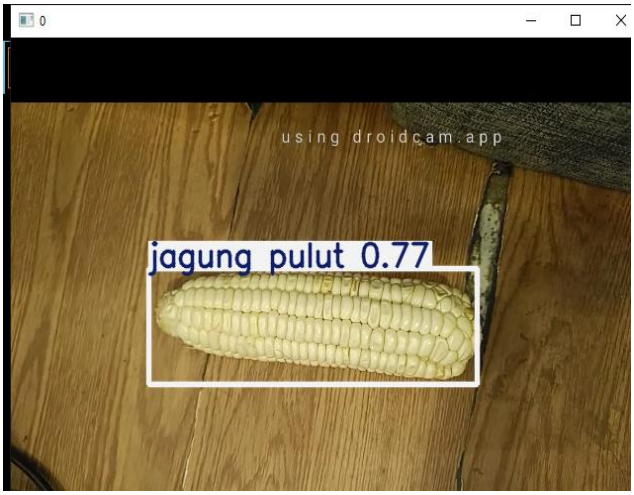
| No. | Gambar | Confidence Score |
|---|---|---|
| | Rumus :<br>objectness = 0.89<br>class object = 0.90<br>Hitung = 0.89 × 0.90<br>= (89/100) × (90/100)<br>= 8010/10000<br>= 0.801<br>= 0.80 | |
| 6. | <br>Rumus :<br>objectness = 0.90<br>class object = 0.90<br>Hitung = 0.90 × 0.90<br>= (90/100) × (90/100)<br>= 8100/10000<br>= 0.81 | 81 % |

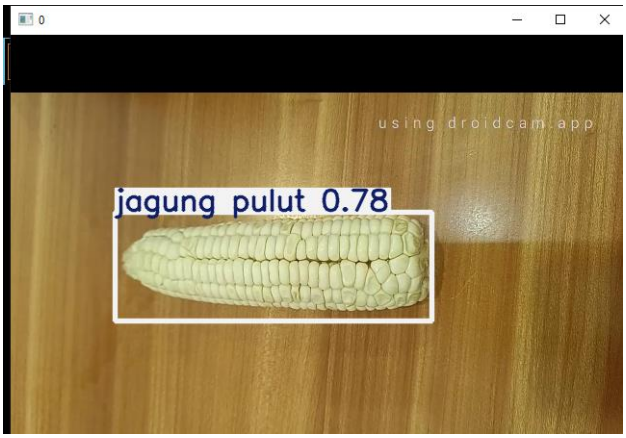
| No. | Gambar | Confidence Score |
|---|---|---|
| 7. | <br>Rumus :<br>objectness = 0.93<br>class object = 0.90<br>Hitung = 0.93 × 0.90<br>= (93/100) × (90/100)<br>= 8370/10000<br>= 0.837<br>= 0.84 | 84 % |
| 8. |  | 86 % |

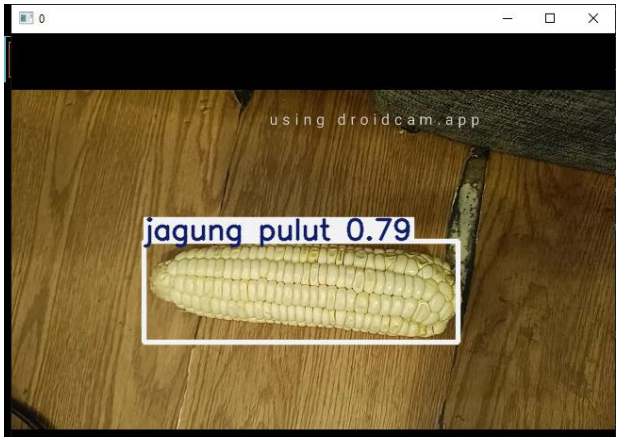
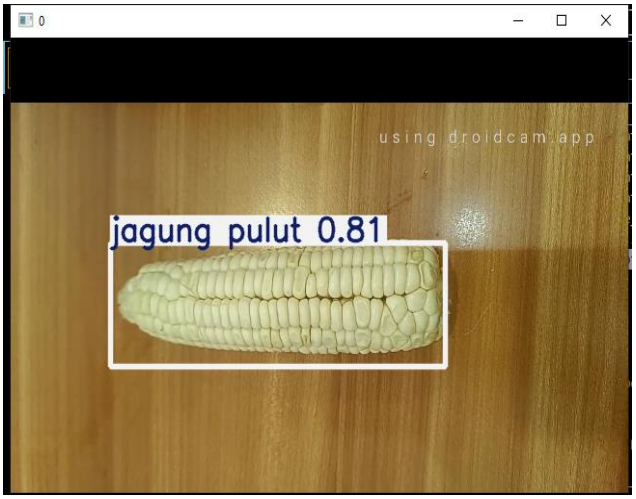
| No. | Gambar | Confidence Score |
|---|---|---|
| | Rumus :<br>objectness = 0.95<br>class object = 0.90<br>Hitung = 0.95 × 0.90<br>= (95/100) × (90/100)<br>= 8550/10000<br>= 0.855<br>= 86 | |
| 9. | <br>Rumus :<br>objectness = 0.95<br>class object = 0.92<br>Hitung = 0.95 × 0.92<br>= (95/100) × (92/100)<br>= 8740/10000<br>= 0.874<br>= 0.87 | 87% |

| No. | Gambar | Confidence Score |
|---|---|---|
| 10. | <br>Rumus :<br>objectness = 0.97<br>class object = 0.96<br>Hitung = 0.97 × 0.96<br>= (97/100) × (96/100)<br>= 9312/10000<br>= 0.9312<br>= 0.93 | 93% |
| 11. |  | 72 % |

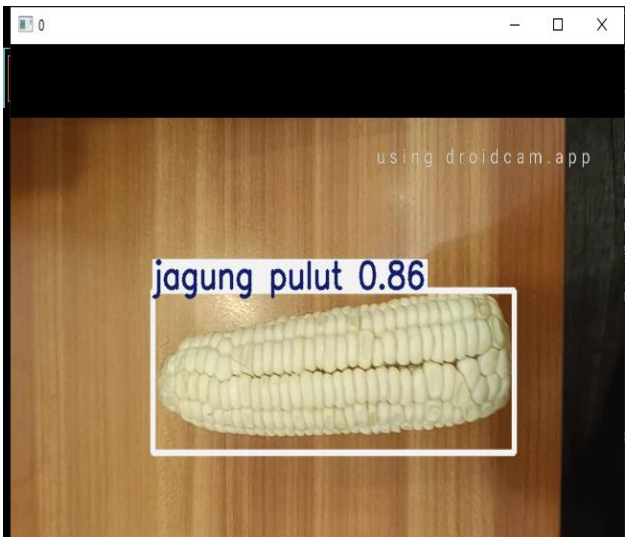
| No. | Gambar | Confidence Score |
|---|---|---|
| | Rumus :<br>objectness = 0.90<br>class object = 0.80<br>Hitung = 0.90 × 0.80<br>= (9/10) × (8/10)<br>= 72/100<br>= 0.72 | |
| 12. | <br>Rumus :<br>objectness = 0.95<br>class object = 0.78<br>Hitung = 0.95 × 0.78<br>= (95/100) × (78/100)<br>= 7410/10000<br>= 0.741<br>= 0.74 | 74 % |

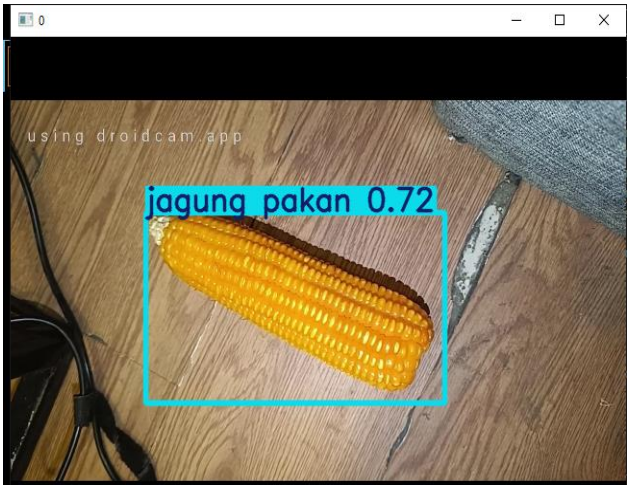
| No. | Gambar | Confidence Score |
|---|---|---|
| 13. | <br>Rumus :<br>objectness = 0.95<br>class object = 0.80<br>Hitung = 0.95 × 0.80<br>= (95/100) × (80/100)<br>= 7600/10000<br>= 0.76 | 76 % |
| 14. |  | 77 % |

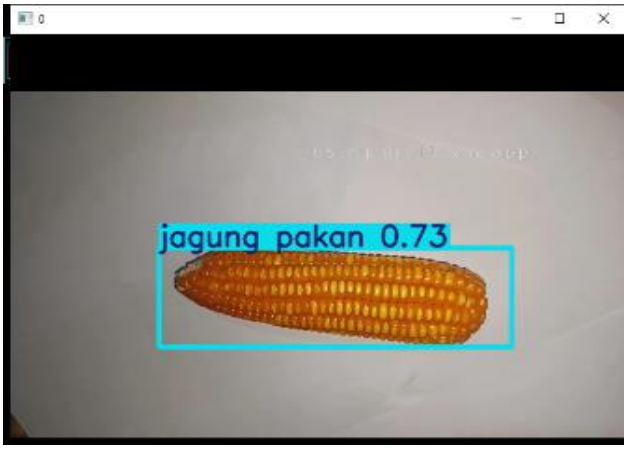
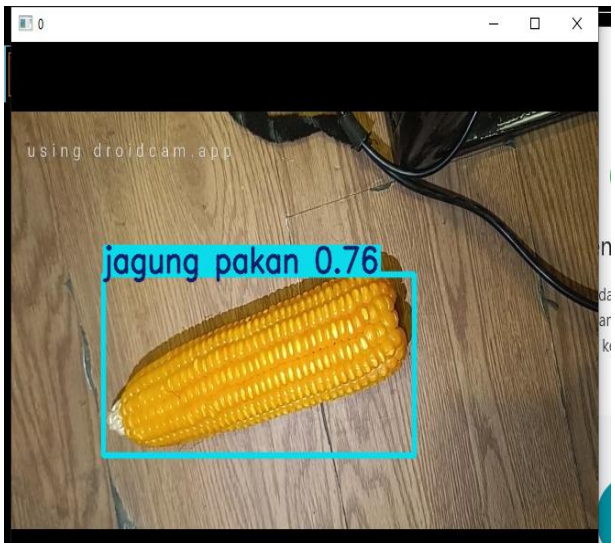
| No. | Gambar | Confidence Score |
|---|---|---|
| | Rumus :<br>objectness = 0.95<br>class object = 0.81<br>Hitung = 0.95 × 0.81<br>= (95/100) × (81/100)<br>= 7695/10000<br>= 0.7695<br>= 0.77 | |
| 15. | <br>Rumus :<br>objectness = 0.95<br>class object = 0.82<br>Hitung = 0.95 × 0.82<br>= (95/100) × (82/100)<br>= 7790/10000<br>= 0.7790<br>= 0.78 | 78 % |

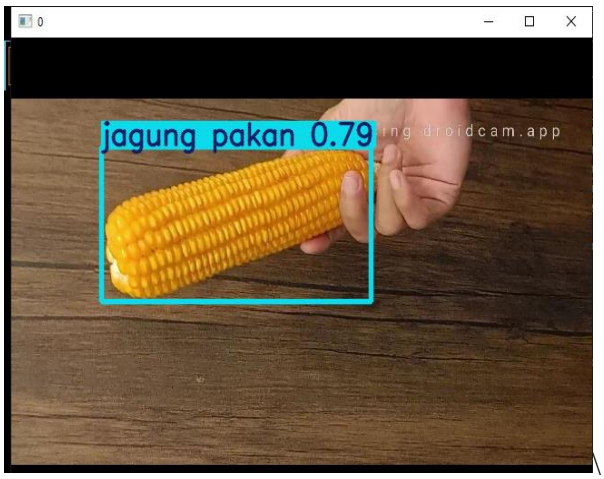
| No. | Gambar | Confidence Score |
|---|---|---|
| 16. | <br>Rumus :<br>objectness = 0.95<br>class object = 0.83<br>Hitung = 0.95 × 0.83<br>= (95/100) × (83/100)<br>= 7885/10000<br>= 0.7885<br>= 0.79 | 79 % |
| 17. |  | 81 % |

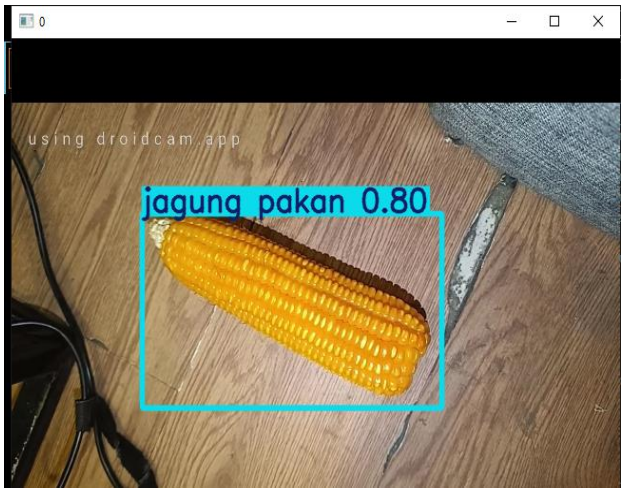
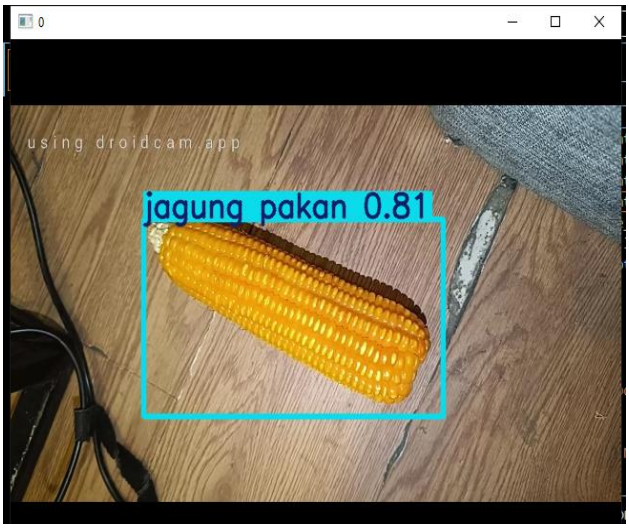
| No. | Gambar | Confidence Score |
|---|---|---|
| | Rumus :<br>objectness = 0.90<br>class object = 0.90<br>Hitung = 0.90 × 0.90<br>= (90/100) × (90/100)<br>= 8100/10000<br>= 0.81 | |
| 18. | <br>Rumus :<br>objectness = 0.93<br>class object = 0.90<br>Hitung = 0.93 × 0.90<br>= (93/100) × (90/100)<br>= 8370/10000<br>= 0.837<br>= 0.84 | 84 % |

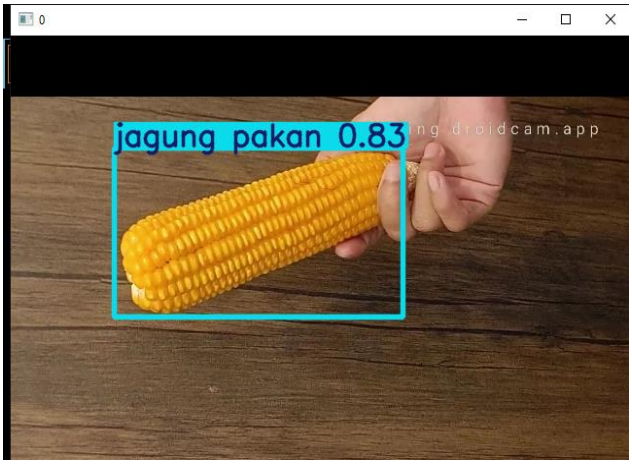
| No. | Gambar | Confidence Score |
|---|---|---|
| 19. | <br>Rumus :<br>objectness = 0.85<br>class object = 1.00<br>Hitung = 0.85 × 1.00<br>= (85/100) × (100/100)<br>= 8500/10000<br>= 0.85 | 85 % |
| 20. |  | 86 % |

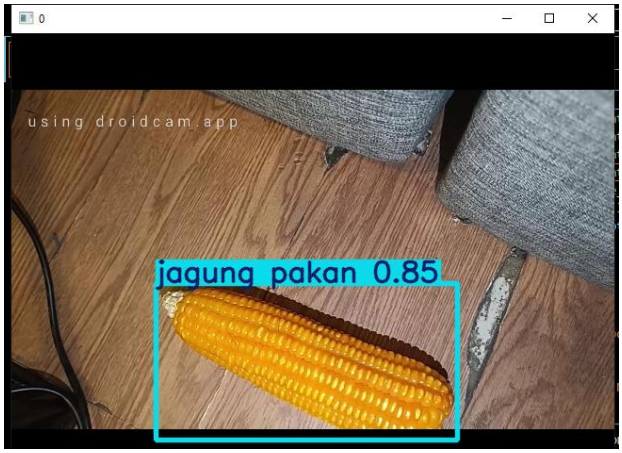
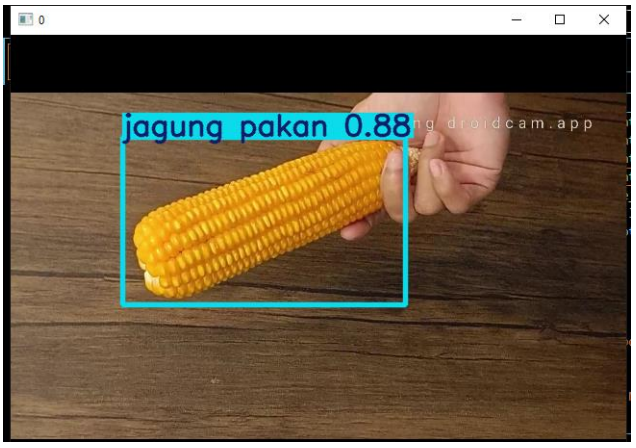
| No. | Gambar | Confidence Score |
|---|---|---|
| | Rumus :<br>objectness = 0.95<br>class object = 0.90<br>Hitung = 0.95 × 0.90<br>= (95/100) × (90/100)<br>= 8550/10000<br>= 0.855<br>= 86 | |
| 21. | <br>Rumus :<br>objectness = 0.90<br>class object = 0.80<br>Hitung = 0.90 × 0.80<br>= (90/100) × (80/100)<br>= 7200/10000<br>= 0.72 | 72 % |

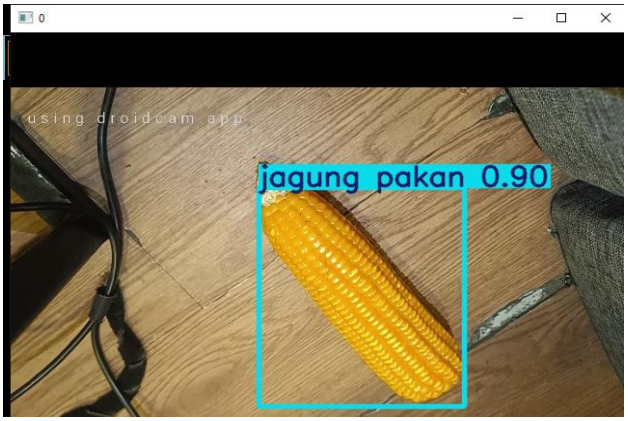
| No. | Gambar | Confidence Score |
|---|---|---|
| 22. | <br>objectness = 0.91<br>class object = 0.80<br>Hitung = 0.91 × 0.80<br>= (91/100) × (80/100)<br>= 7280/10000<br>= 0.7280<br>= 0.73 | 73 % |
| 23. |  | 76 % |

| No. | Gambar | Confidence Score |
|---|---|---|
| | Rumus :<br>objectness = 0.95<br>class object = 0.80<br>Hitung = 0.95 × 0.80<br>= (95/100) × (80/100)<br>= 7600/10000<br>= 0.76 | |
| 24. | <br>Rumus :<br>objectness = 0.95<br>class object = 0.83<br>Hitung = 0.95 × 0.83<br>= (95/100) × (83/100)<br>= 7885/10000<br>= 0.7885<br>= 0.79 | 79 % |

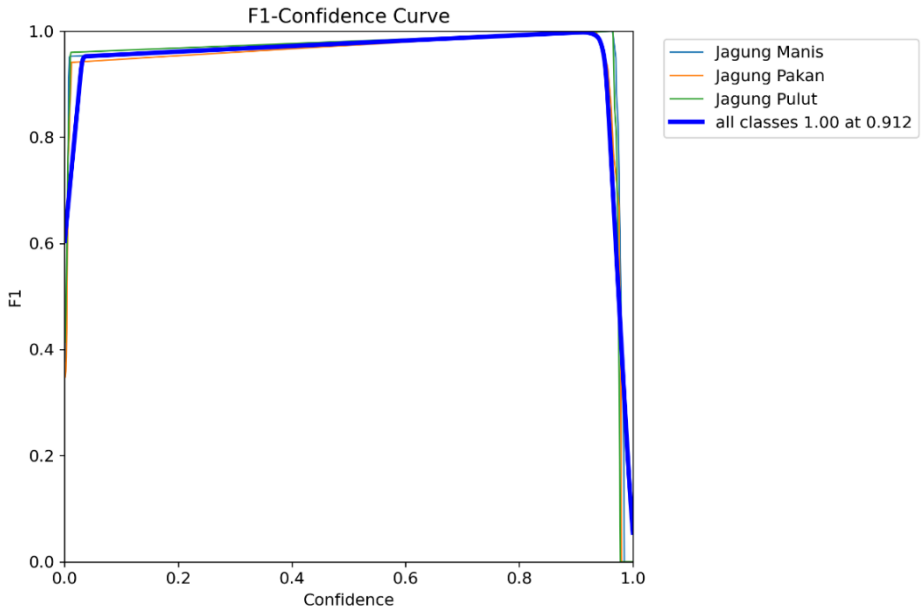
| | | |
|---|---|---|
| 25. | <br>Rumus :<br>objectness = 0.90<br>class object = 0.89<br>Hitung = 0.90 × 0.89<br>= (90/100) × (89/100)<br>= 8010/10000<br>= 0.8010<br>= 0.80 | 80 % |
| 26. |  | 81 % |

| No. | Gambar | Confidence Score |
|---|---|---|
| | Rumus :<br>objectness = 0.90<br>class object = 0.90<br>Hitung = 0.90 × 0.90<br>= (90/100) × (90/100)<br>= 8100/10000<br>= 0.81 | |
| 27. | <br>Rumus :<br>objectness = 0.92<br>class object = 0.90<br>Hitung = 0.92 × 0.90<br>= (92/100) × (90/100)<br>= 8280/10000<br>= 0.8280<br>= 0.83 | 83 % |

| No. | Gambar | Confidence Score |
|---|---|---|
| 28. | <br>Rumus :<br>objectness = 0.85<br>class object = 1.00<br>Hitung = 0.95 × 0.895<br>= (95/100) × (1.00/100)<br>= 8500/10000<br>= 0.8500<br>= 0.85 | 85 % |
| 29. |  | 88 % |

| No. | Gambar | Confidence Score |
|---|---|---|
| | Rumus :<br>objectness = 0.92<br>class object = 0.96<br>Hitung = 0.92 × 0.96<br>= (92/100) × (96/100)<br>= 8832/10000<br>= 0.8832<br>= 0.88 | |
| 30. | <br>Rumus :<br>objectness = 0.90<br>class object = 1.00<br>Hitung = 0.92 × 1.00<br>= (90/100) × (100/100)<br>= 9000/10000<br>= 0.9000<br>= 0.90 | 90 % |

### 6. Grafik Penentuan Confidence Score




**Gambar 4. 22** Grafik Penentuan Confidence Score

Pada grafik ini, penentuan nilai *confidence threshold* yang optimal dilakukan dengan menganalisis kurva hubungan antara nilai **confidence** dan **F1-score**. Grafik F1-Confidence Curve digunakan untuk mengevaluasi performa model dalam mendeteksi tiga jenis jagung, yaitu *Jagung Manis*, *Jagung Pakan*, dan *Jagung Pulut*, dengan melihat keseimbangan antara nilai **precision** dan **recall** pada berbagai tingkat kepercayaan (*confidence*) prediksi.

Grafik pada Gambar 4.3 memperlihatkan nilai F1-score pada sumbu Y dan nilai confidence pada sumbu X. Masing-masing kurva mewakili performa F1 untuk setiap kelas jagung, sedangkan kurva tebal berwarna biru menampilkan nilai rata-rata dari seluruh kelas. Dari grafik tersebut terlihat bahwa terdapat titik puncak pada kurva biru tebal yang menunjukkan nilai F1-score tertinggi, yaitu **1.00**, yang dicapai

pada nilai confidence sebesar **0.912**. Hal ini mengindikasikan bahwa model menghasilkan deteksi terbaik secara keseluruhan pada tingkat kepercayaan tersebut.

Secara matematis, **F1-score** dihitung menggunakan rumus berikut:

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

dengan:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}$$

Keterangan :

- **TP (True Positive)** adalah jumlah deteksi benar,
- **FP (False Positive)** adalah jumlah deteksi salah,
- **FN (False Negative)** adalah jumlah objek yang tidak terdeteksi.

Sementara itu, nilai **confidence score** dalam model deteksi objek seperti YOLO dihitung berdasarkan probabilitas keberadaan objek dan kelasnya:

$$\text{Confidence} = P(\text{objek}) \times \max(P(\text{kelas}))$$

Dengan menetapkan nilai ambang confidence pada **0.912**, sistem hanya menerima prediksi dengan tingkat keyakinan lebih dari atau sama dengan nilai tersebut, sehingga meningkatkan akurasi prediksi sekaligus menekan kemungkinan munculnya kesalahan deteksi. Sebaliknya, jika ambang terlalu rendah, maka akan meningkatkan jumlah *false positive*, sedangkan jika terlalu tinggi akan menurunkan *recall* karena banyak objek yang tidak terdeteksi (*false negative*).

Oleh karena itu, berdasarkan analisis kurva, **nilai confidence sebesar 0.912 dipilih sebagai threshold optimal** dalam tahap inferensi model deteksi jagung untuk mencapai kinerja tertinggi yang seimbang antara precision dan recal.

# BAB V

# PENUTUP

## A. Kesimpulan

Kesimpulan dari hasil penelitian dan pembahasan dari pembuatan Aplikasi Identfikasi Jenis - jenis Jagung berbasis Pengelolahan Citra Digital antara lain:

1. Dengan adanya *aplikasi* ini diharapkan membantu dalam mendeteksi Jenis - jenis Jagung dengan Camera Testing dan Picture Tesing.
2. Sistem mampu mendeteksi dan membedakan tiga jenis jagung (jagung manis, jagung pulut, dan jagung pakan) secara otomatis dengan akurasi tinggi.

## B. Saran

Pada penelitian ini penulis menyadari bahwa masih ada beberapa kekurangan yang sangat diperlukan perbaikan dan pengembangan di penelitian selanjutnya. Oleh karena itu, penulis memiliki bebrapa saran untuk pengembangan selanjutnya dan kekurangan, sebagai berikut:

1. Pengembangan
   a. Penelitian selanjutnya perlu menambah tampilan ataupun menu agar lebih menarik.
   b. *aplikasi* yang penulis buat masih sederhana, terutama tampilan dan yang masih kurang, sehingga untuk pengembangan berikutnya dapat diperbaharui tampilan lebih menarik dan.

2. Kekurangan
    a. sistem hanya mampu mengenali tiga jenis jagung, yaitu jagung manis, jagung pulut, dan jagung pakan, sehingga belum dapat digunakan untuk jenis lainnya.
    b. akurasi deteksi sangat dipengaruhi oleh kondisi pencahayaan pada citra yang diuji, sehingga citra yang terlalu gelap atau terlalu terang dapat menyebabkan kesalahan klasifikasi.

# DAFTAR PUSTAKA


Hasyim, Ade Hastuty. (2022). Dasar Pemrograman (Edisi ke-2). CV. Bangun Bumitama.

Hasyim, Ade Hastuty. (2023). Dasar Pemrograman (Edisi ke-3). CV. Bangun Bumitama.

Hastuty,Ade. &Pawelloi, Irmayani. &Junitasari,Ririn.(2024).Penerapan Algoritma C4.5 Decision Tree Untuk Mendiagnosa Penyakit Dalam. *Sintaks Logika(Silog).4(1):25-26*.

Hastuty,Ade. &Basri, Muh. &Amir, Asfiad. (2021). Sistem Pengenalan Ucapan Bahasa Daerah Menggunakan Metode Mel Frequency Cepstral Coeffient (MFCC) dan Adaptive Neuro - Fuzzy Inference System (ANFIS). *Sintaks Logika(Silog).1(2):76-77*

Hastuty,Ade. &Febrian. (2023). Penerapan Algoritma Vernam Chiper Pada File Tranfer Protocol Server Berbasis PHP. *Sitanks Logika(Silog).3(3):45-46.*

Selao, A., Malik, A. A., Yani, F. I., Mallawa, A., & Safruddin. (2019). Remote Chlorophyll-a and SST to Determination of Fish Potential Area in Makassar Strait Waters Using MODIS Satellite Data. *IOP Conference Series: Earth and Environmental Science, 270*, 012047.

Chamidah, Nurul. Santoni, Mayanda Mega. Prasvita, Desta Sandya. (2023). Pengolahan Citra Digital Menggunakan OpenCV – Python. CV. Budi Utama.s

Asmara, Andrie Rosa. (2018). Pengolahan Citra Digital. POLINEMA PRESS.

Rosnelly, Rika. (2023). Komparasi Metode Peningkatan Citra Tanaman Aggrek Berbasis Pengelolahan Citra Digital. CV. Sarnu Untung.

Ramadahan, Moch Arifqi. & Andarsyah, Roni. (2022). Klasifikasi Text Spam Menggunakan Metode Support Vector Machine Dan Naïve Bayes. Buku Pedia

Jollyta, Denny. & Prihandoko. & Hajjah, Alyauma. & Haerani, Elin. & Siddik, Muhammad. (2023). Algoritma Klasifikasi Untuk Pemula Solusi Python dan Rapidminer. Deepublish Digital (CV. Budi Utama).

Siahaan, Vivian. & Sianpiar, Rismon Hasiholan. (2021). Sinyal dan Citra Digital dengan Python GUI. BALIGE Publishing.

Rosiana, UD. & Mentari, M. & Prasatya, AN. (2019), Klasifikasi Kualitas Biji Jagung Berdasarkan Deteksi Warna dan Bentuk Menggunakan Metode K-Nearest Neighbor. *Simposium Nasional. 13:1-2.*

Munarto, R. & Permata, E. & Salsabilla, R. (2014), Klasifikasi Kualitas Biji Jagung Manis Berdasarkan Fitur Warna Menggunakan Fuzzy Logic. *Sudo Jurnal Teknik Informatika.2(1):1-4.*

Rosadi, M.I. & Lutfy,M. (2021), Identifikasi Jenis Penyakit Daun Jagung Menggunakan Deep Learning Pre-Trained Model. *Jurnal Explore IT. 13(2):1-3.*

Himmah, F.E. & Widyaningsih, M. & Maysaroh. (2020), Identifikasi Kematangan Buah Kelapa Sawit Berdasarkan Warna RGB dan HSV Menggunakan Metode *K-Means Claustering. Jurnal Sains dan Informatika. 9(2):1-4.*

Indrabayu. & Arifin, Nurhikma. & Areni, S.I. (2019), Klasifikasi Kematangan Stroberi Berbasis Segmentasi Warna dengan Metode HSV. *Jurnal Penelitian Enjinering. 23(2):1-4.*

Amrozi, Y. & Yulianti. D. & Susilo, A. & Novianto, N. & Ramadhan, R. (2022), Klasifikasi Jenis Buah Pisang Berdasarkan Citra Warna degan Metode SVM. *Jurnal SISFOKOM (Sistem Informasi dan Komputer). 11(3):1-3.*

Alimunthe, A. (2021), Deteksi Kematangan Buah Manggis Berdasarkan Fitur Warna Citra Kulit Menggunakan Metode Transformasi Ruang Warna HSV.Skripsi tidak dipublikasikan. Medan : Universitas Islam Negeri Sumatera Utara.