IMPLEMENTASI IOT UNTUK DETEKSI DINI SERANGAN HAMA PADA TANAMAN CABAI MERAH MENGGUNAKAN ALGORITMA ISOLATION FOREST

SKRIPSI

DISUSUN OLEH

SURYADARMA
NPM. 2109020036

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS MUHAMMADIYAH SUMATERA UTARA
MEDAN
2025

IMPLEMENTASI IOT UNTUK DETEKSI DINI SERANGAN HAMA PADA TANAMAN CABAI MERAH MENGGUNAKAN ALGORITMA ISOLATION FOREST

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer (S.Kom) dalam Program Studi Teknologi Informasi pada Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Muhammadiyah Sumatera Utara

SURYA DARMA
NPM. 2109020036

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS MUHAMMADIYAH SUMATERA UTARA
MEDAN

LEMBAR PENGESAHAN

Judul Skripsi	:	Implementasi IOT Untuk Deteksi Dini Serangan Hama pada Tanaman Cabai Merah Menggunakan Algoritma Isolation Forest
Nama Mahasiswa	:	Surya Darma
NPM	:	21090200
Program Studi	:	Teknologi Informasi

Menyetujui
Komisi Pembimbing

(Halim Maulana, S.T., M.Kom.)
NIDN. 0121119102

Ketua Program Studi

(Fatma Sari Hutagalung, S.Kom., M.Kom.)
NIDN. 0117019301

Dekan

(Dr. Al-Khowarizmi, S.Kom., M.Kom.)
NIDN. 0127099201

PERNYATAAN ORISINALITAS

IMPLEMENTASI IOT UNTUK DETEKSI DINI SERANGAN HAMA PADA TANAMAN CABAI MERAH MENGGUNAKAN ALGORITMA ISOLATION FOREST

SKRIPSI

Saya menyatakan bahwa karya tulis ini adalah hasil karya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing disebutkan sumbernya.

Medan, September 2025

Yang membuat pernyataan

Surya Darma

NPM. 2109020036

PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademika Universitas Muhammadiyah Sumatera Utara, saya bertanda tangan dibawah ini:

Nama	: Surya Darma
NPM	: 2109020036
Program Studi	: Teknologi Informasi
Karya Ilmiah	: Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Muhammadiyah Sumatera Utara Hak Bedas Royalti Non-Eksekutif *(Non-Exclusive Royalty free Right)* atas penelitian skripsi saya yang berjudul:

IMPLEMENTASI IOT UNTUK DETEKSI DINI SERANGAN HAMA PADA TANAMAN CABAI MERAH MENGGUNAKAN ALGORITMA ISOLATION FOREST

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non-Eksekutif ini, Universitas Muhammadiyah Sumatera Utara berhak menyimpan, mengalih media, memformat, mengelola dalam bentuk database, merawat dan mempublikasikan Skripsi saya ini tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis dan sebagai pemegang dan atau sebagai pemilik hak cipta.

Demikian pernyataan ini dibuat dengan sebenarnya.

Medan, September 2025

Yang membuat pernyataan

Surya Darma

NPM. 2109020036

RIWAYAT HIDUP

DATA PRIBADI

Nama Lengkap	: Surya Darma
Tempat dan Tanggal Lahir	: Binjai Serbangan, 08-April-2003
Alamat Rumah	: LK XI Binjai Serbangan
Telepon/Faks/HP	: 082249146996
E-mail	: suryadarmatba15@gmail.com
Instansi Tempat Kerja	: -
Alamat Kantor	: -

DATA PENDIDIKAN

SD	: SD Negeri 010243 Binjai Serbangan	TAMAT: 2015
SMP	: SMP Negeri 1 Air Joman	TAMAT: 2018
SMA	: SMK Negeri 1 Air Joman	TAMAT: 2021

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah melimpahkan Rahmat dan karunianya sehingga penulis dapat menyelesaikan skripsi dengan judul "Implementasi IoT Untuk Deteksi Dini Serangan Hama Pada Tanaman Cabai merah Menggunakan Algoritma Isolation Forest". Laporan skripsi ini disusun untuk memenuhi salah satu syarat dalam memperoleh gelar Sarjana (S.Kom) pada Program Studi Teknologi Informasi Fakultas Ilmu Komputer dan Teknologii Informasi Universitas Muhammadiyah Sumatera Utara.

Penulis tentunya berterima kasih kepada berbagai pihak dalam dukungan serta doa dalam penyelesaian skripsi. Penulis juga mengucapkan terima kasih kepada:

1. Bapak Prof. Dr. Agussani, M.AP., Rektor Universitas Muhammadiyah Sumatera Utara (UMSU).
2. Bapak Dr. Al-Khowarizmi, S.Kom., M.Kom. Dekan Fakultas Ilmu Komputer dan Teknologi Informasi (FIKTI) UMSU.
3. Ibu Fatma Sari Hutagalung, S.Kom., M.Kom. Ketua Program Studi Teknologi Informasi.
4. Bapak Muhammad Basri, S.Si., M.Kom., Sekretaris Program Studi Teknologi Informasi.

5. Bapak Halim Maulana, S.T., M.Kom., Wakil Dekan I Fakultas Ilmu Komputer dan Teknologi Informasi, Dosen Pembimbing Pembimbing Skripsi Saya.
6. Ibu Dr. Firahmi Rizky, S.Kom., M.Kom., selaku dosen penguji saya.
7. Orang Tua penulis tercinta, Bapak Samsuri dan Mamak Juliana, serta keluarga besar Kakak, Uwek, Abang, Nenek, Kakek dan seluruh keluarga saya yang selalu memberikan doa, kasih sayang, semangat, serta dukungan tanpa henti.
8. Seluruh dosen pembimbing, penguji, serta tenaga pengajar yang telah membimbing dan memberikan ilmu pengetahuan yang sangat berharga selama perkuliahan hingga penyusunan skripsi ini.
9. Teman-teman penulis, Afrida, Hana, dan Rifda, yang senantiasa menemani, mendukung, dan memberi semangat dalam suka maupun duka.
10. Teman-teman kos, Habibi, Rifki, dan Rio, yang telah menjadi keluarga kedua selama menjalani kehidupan perkuliahan.
11. Seluruh mahasiswa kelas TI-A1 angkatan 2021, yang telah memberikan pengalaman kebersamaan, kekompakan dan kerja sama yang tidak ternilai.
12. Teman seperjuangan dalam progam pertukaran mahasiswa, Nema, Gusman, dan Egi yang telah memberi banyak pengalaman dan cerita berharga.
13. Terima kasih juga penulis sampaikan untuk diri sendiri yang tetap berusaha melawan segala kekhawatiran, tidak berhenti berjuang, dan berani mencoba hal-hal baru hingga mampu menyelesaikan perkuliahan ini.

14. Semua pihak yang terlibat langsung ataupun tidak langsung yang tidak dapat penulis ucapkan satu-persatu yang telah membantu penyelesaian skripsi ini

IMPLEMENTASI IOT UNTUK DETEKSI DINI SERANGAN HAMA PADA TANAMAN CABAI MERAH MENGGUNAKAN ALGORITMA ISOLATION FOREST

ABSTRAK

Serangan hama merupakan salah satu faktor utama yang mengancam produksi cabai merah dan dapat menyebabkan kerugian besar bagi petani. Untuk mengantisipasi hal tersebut, diperlukan sistem deteksi dini yang akurat dan efisien. Penelitian ini merancang sistem deteksi anomali berbasis Internet of Things (IoT) dengan menggunakan sensor jarak, suhu, kelembapan, dan gerak yang terhubung ke mikrokontroler ESP32, serta ESP32-CAM untuk dokumentasi visual. Data sensor dikirim secara real-time dan diproses menggunakan algoritma Isolation Forest di Google Colab. Hasil pengujian menunjukkan akurasi sebesar 95%, recall 100% untuk kelas anomali, dan precision 48%, yang berarti semua anomali berhasil terdeteksi meskipun masih terdapat false positive. Integrasi ESP32-CAM memberikan bukti visual untuk mendukung hasil deteksi. Sistem ini terbukti mampu melakukan monitoring tanaman cabai merah secara otomatis, cepat, dan andal, serta berpotensi dikembangkan lebih lanjut untuk mendukung pertanian cerdas.

Kata Kunci: *Internet of Things, ESP32, ESP32-CAM, Isolation Forest, Deteksi Anomali, Cabai Merah.*

IMPLEMENTATION OF IOT FOR EARLY DETECTION OF PEST ATTACKS ON RED CHILI PLANTS USING THE ISOLATION FOREST ALGORITHM

ABSTRACT

Pest attacks are one of the main threats to red chili cultivation, often causing significant yield losses for farmers. To address this issue, an accurate and efficient early detection system is required. This study designs an Internet of Things (IoT)-based anomaly detection system using distance, temperature, humidity, and motion sensors connected to an ESP32 microcontroller, along with an ESP32-CAM for visual documentation. Sensor data are transmitted in real-time and processed using the Isolation Forest algorithm in Google Colab. The experimental results show an accuracy of 95%, a recall of 100% for the anomaly class, and a precision of 48%, indicating that all anomalies were successfully detected although some false positives occurred. The integration of ESP32-CAM enhances system reliability by providing visual evidence to support anomaly detection. Overall, the system enables automatic, fast, and reliable monitoring of red chili plants and has potential for further development to support smart farming practices.

Keywords: *Internet of Things, ESP32, ESP32-CAM, Isolation Forest, Anomaly Detection, Red Chili Plants.*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1. Latar Belakang Masalah

Cabai merah merupakan salah satu komoditas hortikultura yang memiliki nilai ekonomi tinggi di Indonesia dan menjadi bahan pokok penting dalam kebutuhan sehari-hari masyarakat. Namun, produktivitas tanaman cabai sering kali mengalami penurunan akibat serangan berbagai jenis hama, seperti ulat, kutu daun, dan lalat buah. Serangan hama tersebut tidak hanya mengurangi hasil panen, tetapi juga berdampak pada kualitas dan harga jual cabai di pasaran. Salah satu kendala utama adalah kesulitan dalam mendeteksi serangan hama secara dini, sehingga upaya pengendalian menjadi terlambat dan kurang efektif. Selain itu, penggunaan teknologi modern seperti Internet of Things (IoT) di bidang pertanian, terutama untuk deteksi hama sejak awal, masih belum dimanfaatkan secara optimal, padahal teknologi ini memiliki potensi besar untuk meningkatkan efisiensi dan produktivitas tanaman cabai (Talli et al., 2023).

Selama ini, petani masih mengandalkan metode manual berupa pengamatan langsung yang memerlukan waktu, kurang akurat, dan tidak efisien. Keterlambatan penanganan hama, yang pada akhirnya berdampak pada turunnya produksi dan mutu cabai merah. Belum adanya sistem otomatis dan real-time yang dapat melakukan deteksi dini serangan hama pada tanaman cabai merah.

Serangan hama menjadi penyebab utama kerugian besar di sektor budidaya cabai. Berdasarkan beberapa penelitian, tingginya tingkat serangan organisme pengganggu tanaman menjadi kendala serius yang sering kali berakhir pada gagal panen dan kerugian ekonomi yang signifikan. Deteksi hama yang lambat membuat langkah penanganan tidak dapat dilakukan tepat waktu, sedangkan metode deteksi manual yang masih banyak digunakan tidak mampu menyediakan data kondisi tanaman secara real-time yang diperlukan untuk pengambilan keputusan cepat (Alamsyah & Kurniawan, 2021).

Penelitian ini mengusulkan pengembangan sistem monitoring berbasis IoT yang dipadukan dengan algoritma kecerdasan buatan guna mengatasi masalah tersebut. Sistem ini menggunakan ESP32 untuk mengumpulkan data dari sensor suhu, kelembaban, jarak, dan gerakan, serta ESP32-CAM untuk mengambil gambar tanaman. Data sensor akan dianalisis menggunakan algoritma Isolation Forest untuk mendeteksi anomali, sementara gambar tanaman akan diproses dengan algoritma YOLOv5 untuk mengenali tanda-tanda serangan hama.

Pemanfaatan IoT di sektor pertanian telah terbukti dapat meningkatkan efisiensi pemantauan tanaman sekaligus mempercepat proses pengambilan keputusan. Dengan integrasi algoritma Isolation Forest, sistem dapat mendeteksi pola data yang menyimpang dan menjadi indikasi awal adanya serangan hama. Deteksi dini ini memungkinkan petani untuk melakukan tindakan pencegahan lebih cepat, sehingga dapat mengurangi kerusakan tanaman dan meningkatkan hasil panen. Sejumlah penelitian sebelumnya juga menunjukkan bahwa sistem monitoring berbasis IoT dapat mempermudah pengawasan tanaman secara otomatis dan real-time (Budiani et al., 2024). Selain itu, YOLOv5 terbukti dapat mendeteksi

penyakit cabai dengan nilai precision 0,946; recall 0,936; dan mAP 0,959. Oleh karena itu, kolaborasi antara IoT, Isolation Forest, dan YOLOv5 diharapkan mampu memberikan deteksi dini yang lebih cepat dan akurat dibandingkan metode konvensional.

Secara teknis, data sensor dikirim ke server database (iot_db), sedangkan gambar yang diambil oleh ESP32-CAM disimpan di Google Drive untuk dianalisis oleh YOLOv5. Hasil analisis dari sensor dan gambar kemudian dikirimkan kepada pengguna melalui notifikasi dari Telegram Bot, sehingga petani dapat menerima informasi secara real-time mengenai kondisi lingkungan, skor anomali, dan bukti visual kondisi tanaman.

Dengan implementasi sistem ini, diharapkan deteksi dini serangan hama pada tanaman cabai merah menjadi lebih akurat, cepat, dan terintegrasi. Tujuan akhirnya adalah mengurangi kerugian petani, meningkatkan produktivitas dan kualitas panen, serta mendorong praktik pertanian yang lebih efisien dan berkelanjutan.

1.2. Rumusan Masalah

Berdasarkan latar belakang tersebut, tujuan penelitian ini adalah merancang sistem yang mengintegrasikan teknologi IoT dan kecerdasan buatan untuk meningkatkan efektivitas deteksi dini hama pada tanaman cabai merah. Pertanyaan penelitian yang diajukan meliputi:

1. Bagaimana merancang dan mengimplementasikan sistem monitoring berbasis IoT yang dapat memantau kondisi lingkungan terbatas tanaman cabai merah dengan menggunakan sensor suhu, kelembaban, gerakan, dan jarak?

2. Bagaimana algoritma Isolation Forest dapat diterapkan untuk menganalisis data sensor guna mendeteksi anomali yang mengindikasikan potensi serangan hama pada lahan uji coba?
3. Bagaimana algoritma YOLOv5 digunakan untuk menganalisis gambar tanaman yang diambil oleh ESP32-CAM guna mengidentifikasi tanda-tanda visual dari serangan hama?
4. Bagaimana mekanisme notifikasi otomatis melalui Telegram Bot dapat dibangun untuk mengirimkan data sensor, skor anomali, serta hasil deteksi visual secara periodik maupun real-time?
5. Bagaimana kinerja dan efektivitas sistem ini dapat dievaluasi, dengan mengukur akurasi, precision, recall, dan F1-score dari hasil deteksi simulasi?

1.3. Batasan Masalah

Penelitian ini difokuskan pada aspek-aspek utama untuk menjamin kelayakan dan efektivitas proyek:

1. Sistem yang dikembangkan hanya berfokus pada deteksi dini potensi serangan hama pada tanaman cabai merah menggunakan sensor suhu, kelembaban, sensor gerakan (PIR), dan sensor jarak ultrasonik.
2. Penelitian ini secara spesifik menggunakan algoritma Isolation Forest untuk analisis data anomali sensor, tanpa membandingkannya dengan algoritma deteksi anomali lainnya.
3. Analisis visual tanaman akan dilakukan secara eksklusif menggunakan algoritma YOLOv5, dengan dataset yang terbatas pada gambar tanaman cabai merah yang diperoleh langsung dari ESP32-CAM.

4. Pengambilan gambar tanaman oleh ESP32-CAM dijadwalkan secara periodik (setiap satu jam) dan dapat diaktifkan berdasarkan permintaan pengguna melalui Telegram Bot.
5. Pengujian sistem hanya dilakukan pada area uji coba skala kecil.
6. Fitur notifikasi sistem terbatas pada pengiriman data sensor, skor anomali, dan hasil deteksi gambar ke Telegram Bot, tanpa adanya fungsi kendali otomatis untuk pengendalian hama.

1.4. Tujuan Penelitian

Tujuan utama dari penelitian ini adalah untuk:

1. Mendesain dan mengimplementasikan sistem monitoring IoT yang mampu memantau kondisi lingkungan tanaman cabai merah dengan menggunakan sensor suhu, kelembaban, gerakan, dan jarak pada area uji coba terbatas.
2. Menerapkan algoritma Isolation Forest untuk menganalisis data sensor guna mendeteksi anomali yang dapat menjadi indikasi awal serangan hama.
3. Menggunakan algoritma YOLOv5 untuk menganalisis gambar tanaman dari ESP32-CAM dalam rangka mengidentifikasi tanda-tanda visual yang mengindikasikan serangan hama.
4. Membangun mekanisme notifikasi otomatis melalui Telegram Bot yang dapat menyampaikan informasi data sensor, skor anomali, dan hasil deteksi gambar secara periodik dan real-time.
5. Mengevaluasi kinerja dan akurasi sistem dalam mendeteksi potensi serangan hama, dengan menggunakan metrik seperti akurasi, precision, recall, dan F1-score pada pengujian skala kecil.

1.5. Manfaat Penelitian

Hasil dari penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Bagi Petani

Sistem ini akan membantu petani mendeteksi gangguan lingkungan dan tanda-tanda serangan hama secara dini, memungkinkan mereka mengambil tindakan pencegahan yang lebih cepat, tepat, dan didasarkan pada data yang akurat.

2. Bagi Masyarakat Umum

Penelitian ini dapat mempermudah pemantauan kondisi pertanian dari jarak jauh melalui notifikasi otomatis yang berisi data sensor dan gambar tanaman secara real-time melalui Telegram.

3. Bagi Mahasiswa

Bagi mahasiswa, penelitian ini dapat menjadi sumber referensi dan studi kasus dalam memahami penerapan praktis teknologi IoT, algoritma deteksi anomali seperti Isolation Forest, serta metode deteksi visual seperti YOLOv5 dalam konteks aplikasi nyata.

4. Bagi Peneliti dan Akademisi

Penelitian ini memberikan kontribusi pada pengembangan sistem pertanian cerdas dengan mengintegrasikan IoT dengan machine learning dan deep learning. Integrasi Isolation Forest dan YOLOv5 dapat menjadi dasar bagi penelitian lanjutan di bidang pertanian presisi, kecerdasan buatan, dan sistem deteksi otomatis.

BAB II

LANDASAN TEORI

2.1. Internet of Things (IoT)

Internet of Things (IoT) merupakan konsep di mana berbagai perangkat fisik saling terhubung melalui jaringan internet untuk bertukar dan mengolah data secara otomatis. Teknologi ini bekerja dengan memanfaatkan sensor dan perangkat cerdas yang dapat mengamati kondisi lingkungan dan mengirimkan informasi secara real-time kepada pengguna. Dalam bidang pertanian, penerapan IoT membantu petani dalam mengatasi tantangan pengelolaan lahan, seperti memantau suhu, kelembapan, kesuburan tanah, hingga mendeteksi keberadaan hama secara lebih efisien (Sari et al., 2024).

2.2. Pertanian

Penerapan teknologi IoT dalam dunia pertanian menghadirkan solusi inovatif untuk meningkatkan produktivitas dan kualitas hasil panen. Melalui sensor IoT, petani dapat memantau kondisi tanaman secara langsung, termasuk kadar kelembapan tanah dan aktivitas hama. Teknologi ini mampu menghemat waktu, tenaga, serta mengurangi penggunaan sumber daya yang berlebihan, sehingga lebih ramah lingkungan dan berkelanjutan (Sari et al., 2024).

2.3. Hama pada Tanaman Cabai Merah

Hama pada tanaman sayuran adalah organisme pengganggu yang dapat merusak tanaman dan mengganggu proses pertumbuhannya. Faktor yang mempengaruhi datangnya hama antara lain kondisi lingkungan, seperti kelembapan dan suhu, serta adanya tanaman yang rentan atau tidak sehat. Pada tanaman cabai

merah, beberapa hama yang umum ditemukan adalah ulat grayak, kutu daun, dan lalat buah. Dampak dari hama ini termasuk penurunan kualitas dan kuantitas hasil panen, serta potensi kerusakan yang dapat mengakibatkan kerugian finansial bagi petani.

2.4. Isolation Forest

Isolation Forest merupakan algoritma unsupervised learning yang digunakan untuk mendeteksi data anomali berdasarkan prinsip isolasi. Metode ini bekerja dengan membangun struktur pohon acak (tree structure) yang memisahkan setiap data ke dalam subgrup lebih kecil hingga data anomali dapat terisolasi dengan cepat. Proses isolasi dilakukan dengan memilih fitur dan nilai pembagi secara acak. Data yang dianggap anomali umumnya membutuhkan jumlah pemisahan lebih sedikit dibandingkan data normal. Dengan demikian, semakin pendek jarak suatu data dari akar pohon, semakin besar kemungkinan data tersebut tergolong anomali. Hasil dari pembentukan banyak pohon ini kemudian digunakan untuk menilai apakah data baru termasuk anomali atau tidak (Wibawa & Karyawati, 2023).

Tahapan kerjanya meliputi:

1. Pembuatan Isolation Trees

Membentuk beberapa pohon isolasi dengan memilih fitur secara acak, kemudian memilih nilai split acak dalam rentang nilai fitur tersebut. Proses pembagian berlanjut sampai hanya tersisa 1 data, atau pohon telah mencapai kedalaman maksimum.

2. Menghitung Panjang Jalur (h(x))

Mengukur jumlah langkah yang dibutuhkan agar satu data terisolasi. Semakin pendek jalurnya maka,semakin besar peluang data tersebut anomali.

3. Perhitungan Skor Anomali

Rumus Skor anomali Isolation Forest:

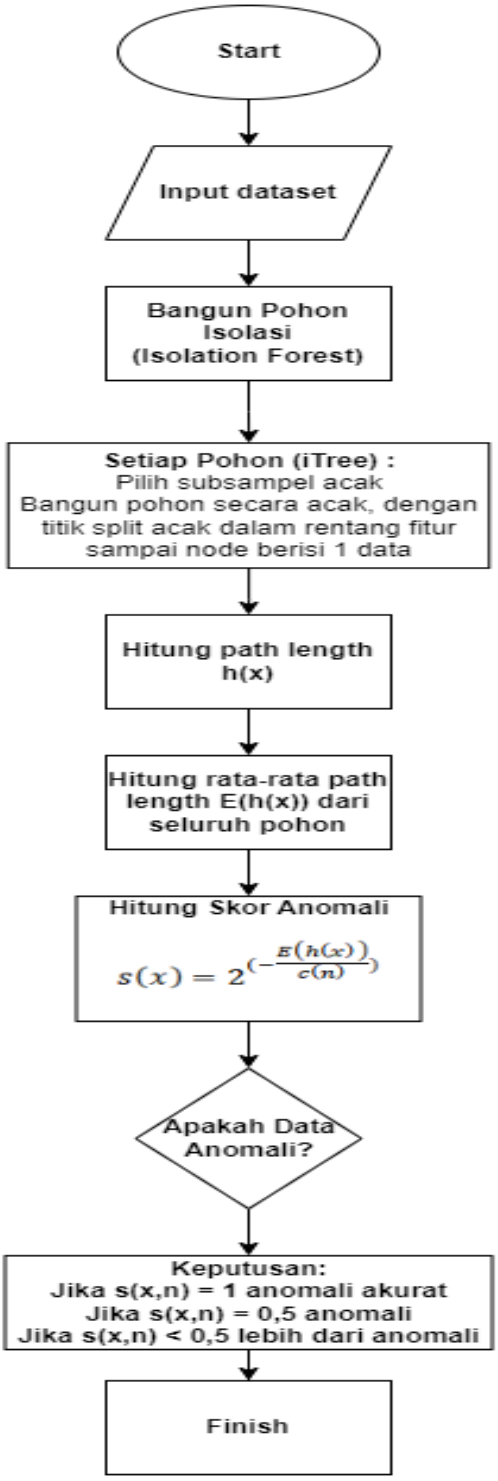
$$s(x) = 2^{(-\frac{E(h(x))}{c(n)})}$$

4. Interpretasi Hasil

Jika skor mendekati 1 maka, kemungkinan besar anomali dan jika skor mendekati 0 maka, data normal.

Pada dasarnya, algoritma ini tidak dapat langsung digunakan jika hanya terdapat satu data karena proses deteksi anomali memerlukan perbandingan antar data. Oleh karena itu, pada tahap awal pengujian (1 jam pertama), sistem belum dapat menentukan skor anomali. Penghitungan skor baru dapat dilakukan setelah terkumpul minimal 5 hingga 10 data agar perhitungannya lebih relevan. Selama proses pengumpulan data tersebut, sistem tetap mengirimkan data dan gambar ke pengguna, meskipun status anomali belum ditampilkan.

Secara bertahap, proses pengujian sistem akan berjalan lebih stabil. Pada awalnya data hanya dikumpulkan, kemudian dilakukan analisis, hingga akhirnya sistem mampu bekerja secara otomatis dalam mendeteksi anomali dan mengirimkan notifikasi real-time. Jika tidak ditemukan anomali, sistem akan tetap mengirimkan pemberitahuan bahwa kondisi terpantau normal. Dengan langkah ini, kinerja sistem menjadi lebih akurat sesuai dengan metode Isolation Forest.



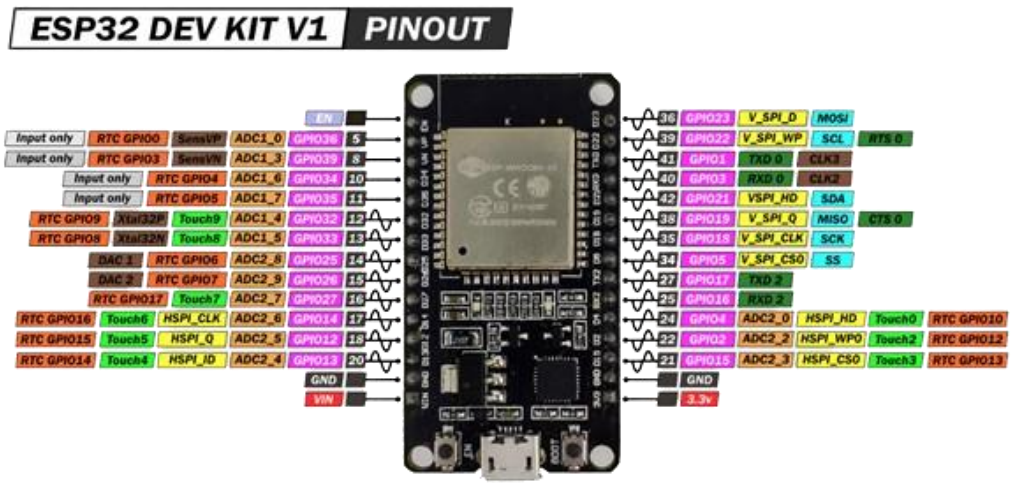
Gambar 2.1 Flowchart Isolation Forest

2.5 YOLOv5

You Only Look Once versi 5 (YOLOv5) adalah algoritma deteksi objek real-time yang sangat efisien dalam mengenali dan melokalisasi objek pada citra digital. Algoritma ini membagi gambar menjadi beberapa grid dan secara bersamaan memprediksi kelas serta posisi objek dalam bentuk bounding box. Keunggulan YOLOv5 terletak pada kecepatan pemrosesan serta akurasi deteksinya yang tinggi, menjadikannya sangat relevan untuk mendeteksi hama atau penyakit tanaman secara otomatis di bidang pertanian (Riva et al., 2023).

2.6 Mikrokontroler

Mikrokontroler yang digunakan dalam sistem pertanian berbasis Internet of Things (IoT) berfungsi sebagai otak utama untuk mengendalikan dan memproses data dari berbagai sensor seperti suhu dan kelembapan tanah. Mikrokontroler ini menjalankan program yang mengatur perilaku perangkat elektronik yang terhubung, termasuk pengendalian sistem deteksi dini terhadap hama dan penyakit tanaman dapat dilakukan lebih cepat dan akurat.



Gambar 2.2 ESP32

Salah satu mikrokontoller yang sering digunakan dalam aplikasi IoT pertanian adalah ESP32. ESP32 memiliki kemampuan konektivitas WiFi dan Bluetooth yang memudahkan pengiriman data ke cloud untuk pemantauan jarak jauh dan pengendalian sistem secara otomatis (Wahyudi et al., 2025).

2.7 Sensor Ultrasonik

Sensor ultrasonik merupakan alat elektronik yang digunakan untuk mengukur jarak atau mendeteksi keberadaan suatu objek dengan memanfaatkan gelombang suara berfrekuensi tinggi (lebih dari 20 kHz), yang tidak dapat didengar oleh telinga manusia. Prinsip kerjanya adalah dengan memancarkan gelombang ke arah objek, lalu menangkap pantulannya kembali. Selisih waktu antara pengiriman dan penerimaan gelombang digunakan untuk menghitung jarak dengan tingkat presisi yang tinggi (Putra et al., 2024).

Gambar 2.3 Sensor Ultrasonik

2.8 Sensor Suhu DHT11

Sensor DHT11 adalah sensor digital yang umum digunakan untuk mengukur suhu dan kelembapan udara secara efisien. Sensor ini mampu mengukur suhu pada rentang 0°C–50°C dan kelembapan antara 20%–80%, dengan tingkat akurasi ±2°C untuk suhu dan ±5% untuk kelembapan. DHT11 memiliki empat pin utama, yaitu VCC, Data, GND, dan NC, serta mampu melakukan pembacaan data hingga satu kali per detik (No et al., 2023).

Gambar 2.4 Sensor DHT11

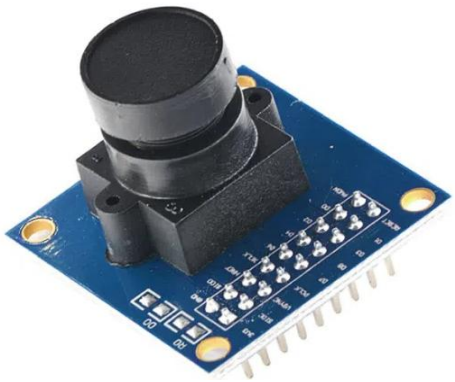
2.9 Sensor PIR

Sensor PIR berfungsi untuk mendeteksi gerakan berdasarkan perubahan radiasi inframerah yang dipancarkan oleh makhluk hidup di sekitarnya. Sensor ini bersifat pasif karena tidak memancarkan sinyal sendiri, melainkan hanya menangkap radiasi dari lingkungan. PIR banyak diterapkan pada sistem deteksi atau pengusiran hama di pertanian, khususnya pada tanaman hortikultura seperti cabai dan sayuran lain, dengan jarak deteksi efektif antara 50 hingga 150 cm.

Gambar 2.5 Sensor PIR

2.10 ESP32-CAM

ESP32-CAM merupakan modul mikrokontroler yang dilengkapi dengan kamera OV2640 dan digunakan untuk mengambil gambar secara otomatis. Modul ini memungkinkan pengambilan citra kondisi tanaman secara berkala dan mengirimkannya ke server atau penyimpanan awan. Dengan fitur konektivitas Wi-Fi bawaan, gambar yang dihasilkan dapat dikirim langsung untuk dianalisis atau dipantau oleh pengguna. Dalam konteks penelitian ini, modul ESP32-CAM digunakan untuk memantau kondisi tanaman cabai merah secara visual (As'da et al., 2024).



Gambar 2.6 ESP32-CAM

2.11 Arduino IDE

Arduino IDE (Integrated Development Environment) adalah perangkat lunak sumber terbuka yang berfungsi untuk menulis, menyusun, dan mengunggah program ke papan mikrokontroler Arduino. IDE ini menyediakan antarmuka teks sederhana, konsol status, dan tombol kompilasi untuk memverifikasi serta mengunggah kode. Bahasa pemrograman yang digunakan berbasis C/C++ yang disederhanakan dan didukung berbagai library agar pengguna dapat mengoperasikan perangkat keras dengan mudah. Arduino IDE juga dilengkapi fitur serial monitor untuk komunikasi data real-time antara komputer dan mikrokontroler (Santoso & Wijayanto, 2022).

2.12 Telegram

Telegram adalah aplikasi pesan instan berbasis cloud yang memungkinkan pengguna untuk mengirim teks, gambar, video, file, serta melakukan panggilan suara dan video secara gratis melalui internet. Aplikasi ini menekankan aspek kecepatan, keamanan, dan privasi, dengan fitur enkripsi end-to-end pada secret chat untuk menjaga kerahasiaan data pengguna (Aisyah & Febriyanto, 2024).

Selain fungsi komunikasi, Telegram juga menyediakan fitur bot, yaitu program otomatis yang mampu memberikan notifikasi, pengingat, serta integrasi dengan sistem eksternal. Kemampuan inilah yang membuat Telegram sering dimanfaatkan dalam penelitian dan sistem otomatisasi, termasuk untuk memberikan peringatan atau laporan deteksi hama secara real-time (Darusman, 2023).

Dalam penelitian ini, sistem IoT digunakan untuk mendeteksi keberadaan hama cabai merah menggunakan data dari sensor suhu, dan kelembaban. Setelah data diproses dengan Isolation Forest untuk mendeteksi anomali (aktivitas tidak wajar yang mengindikasikan keberadaan hama) dan hasil akhir dikomunikasikan kepada pengguna melalui notifikasi otomatis ke Telegram.

2.13 Penelitian Terdahulu

Tabel 2.1 Penelitian Terdahulu

No	Penulis (Tahun)	Judul Penelitian	Hasil Penelitian
1.	Muflih Riyadi (2023) (Riyadi, 2023)	Sistem Cerdas Untuk Monitoring Pengukuran Suhu dan Kelembapan Tanah pada Tanaman Cabai Berbasis Internet of Things (IoT) Menggunakan Apikasi Telegram	Sistem monitoring IoT ini menggunakan NodeMCU ESP8266 dan sensor untuk mengukur suhu dan kelembapan tanah, dengan notifikasi via Telegram. Meskipun efektif, sistem ini memiliki sedikit delay dan belum diuji ketahanan atau akurasinya.
2.	(Talli et al., 2023)	Rancanag Bangun Sistem Monitoring Kualitas Tanah untuk Tanaman	Sistem IoT ini memantau kualitas tanah cabai menggunakan sensor pH, soil moisture, dan DHT11. Data ditampilkan di website dan Telegram. Efektif untuk

		Cabai Berbasis IoT (Internet of Things)	pemantauan jarak jauh, meskipun memiliki error sensor dan belum dilengkapi fitur otomatisasi atau aplikasi mobile.
3.	(Wibawa & Karyawati, 2023)	Isolation Forest dengan Exploratory Data Analysis pada Anomaly Detection untuk Data Transaksi	Algoritma Isolation Forest digunakan untuk mendeteksi anomali transaksi keuangan dengan akurasi 89,9%. Meskipun efektif, precision rendah akibat data tidak seimbang, sehingga diperlukan pengembangan lebih lanjut.
4.	(Zulfikar et al., 2023)	Deteksi Anomali menggunakan Isolation Forest Belanja Barang Persediaan Konsumsi pada Satuan Kerja Kepolisian Republik Indonesia	Menggunakan algoritma Isolation Forest, penelitian ini berhasil mendeteksi 99 anomali pada transaksi konsumsi Polri. Meskipun efektif, dibutuhkan pengembangan lanjutan untuk meningkatkan akurasi.
5.	(Musyaffa et al., 2023)	Smart Plant Monitoring System Kelembapan Tanah	Sistem IoT ini secara otomatis mengontrol kelembaban tanah cabai dengan akurasi 91%, namun

		Pada Tumbuhan Cabai Berbasis IoT	ketahanan sistem jangka panjang belum teruji.

BAB III

METODOLOGI PENELITIAN

3.1. Tempat dan Kerangka Penelitian

3.1.1. Tempat Penelitian

Pada penelitian implementasi IoT untuk mendeteksi hama pertanian dengan metode isolation forest, tempat penelitian dilakukan di Jl. Gunung Sibayak No 6, Kelurahan Gelugur Darat 2. Kecamatan Medan Timur.

3.1.2. Kerangka Waktu Penelitian

Tabel 3.1 Waktu Penelitian

No	Urutan Kegiatan	Bulan I	Bulan II	Bulan III	Bulan IV	Bulan V	Bulan VI
1.	Identifikasi Masalah dan Studi Literatur	■					
2.	Perancangan Sistem		■				
3.	Implementasi Sistem			■			
4.	Pengujian Sistem				■		
5.	Analisis dan Evaluasi					■	
6.	Penyusunan Laporan Penelitian						■

3.2. Alat dan Bahan

Perangkat Keras (Hardware)

Tabel 3.2 Hardware

No	Nama	Fungsi
1.	ESP32	Untuk membaca sensor dan mengirim data ke laptop atau cloud.
2.	Sensor DHT11	Mengukur suhu dan kelembapan di lingkungan tanaman.
3.	Sensor Ultrasonik	Mendeteksi gerakan atau keberadaan objek di sekitar hama.
4.	Kabel Jumper	Membuat koneksi antar komponen tanpa solder.
5.	Sensor PIR	Mendeteksi pergerakan dengan mendeteksi perubahan radiasi inframerah yang dipancarkan.
6.	ESP32-CAM	Mengambil gambar secara berkala
7.	Baterai Li-ion	Memberi daya untuk mikrokontroler dan sensor.
8.	Kotak Pelindung	Melindungi perangkat dari cuaca di lahan.
9.	Laptop	Untuk memproses data, menjalankan program Phyton, dan mengelola bot Telegram.

Perangkat Lunak (Software)

Tabel 3.3 Software

No	Nama	Fungsi
1.	Arduino IDE	Untuk memprogram ESP32 agar membaca sensor dan kirim data.

2.	Google Colab	Menjalankan pemrosesan data dan model anomali menggunakan cloud.
3.	Library Phython	Untuk analisis data, deteksi nomali, dan integrasi bot Telegram.
4.	VS Code	Untuk menjalankan dan mengelola script Phython.
5.	Bot Telegram	Sebagai antarmuka notifikasi ke pengguna secara langsung melalui chat.
6.	MySQL	Sistem manajemen basis data untuk menyimpan data sensor (iot_db).
7.	PyTorch	Kerangka kerja (framework) untuk mengimplementasikan algoritma YOLOv5 untuk mendeteksi objek dan hama pada gambar.

Bahan Pendukung

Tabel 3.4 Bahan Pendukung

No	Nama	Fungsi
1.	Tanaman cabai merah	Sebagai objek uji coba system deteksi hama dan lingkungan.
2.	Koneksi internet	Untuk ESP32 mengirim data dan komunikasi sistem dengan bot Telegram.
3.	File CSV	Menyimpan data hasil pembacaan sensor untuk dianalisis.

3.3. Desain Sistem

Sistem yang dikembangkan dalam penelitian ini terdiri atas beberapa tahapan utama yang saling berkaitan dan dirancang untuk mencapai hasil yang optimal.

1. Pengumpulan Data Sensor : ESP32 akan membaca data dari sensor DHT11, PIR, dan ultrasonik. Data ini kemudian dikirimkan secara real-time ke server basis data (iot_db).
2. Pengambilan Gambar : ESP32-CAM akan mengambil gambar tanaman secara periodik (setiap 20 detik). Gambar yang diambil akan disimpan di Google Drive untuk analisis.
3. Analisis Anomali Sensor (Isolation Forest) : Data sensor akan diolah di Google Colab menggunakan algoritma Isolation Forest untuk mengidentifikasi pola data yang menyimpang, yang dapat mengindikasikan potensi serangan hama.
4. Analisis Visual Tanaman (YOLOv5): Gambar yang tersimpan di Google Drive akan dianalisis menggunakan YOLOv5. Proses ini akan mendeteksi dan menandai keberadaan objek atau tanda visual yang mengindikasikan hama. Hasilnya mencakup label objek, tingkat kepercayaan (confidence score), dan lokasi objek.
5. Notifikasi Otomatis: Hasil analisis dari data sensor dan gambar akan dikirimkan secara otomatis kepada pengguna melalui Telegram Bot. Notifikasi ini berisi data sensor terkini, skor anomali dari Isolation Forest, dan gambar tanaman yang sudah dianalisis oleh YOLOv5.
6. Evaluasi Kinerja: Sistem akan diuji di lahan skala kecil. Kinerjanya akan dievaluasi berdasarkan akurasi deteksi anomali (Isolation Forest), akurasi deteksi visual (YOLOv5), dan efektivitas integrasi notifikasi Telegram.

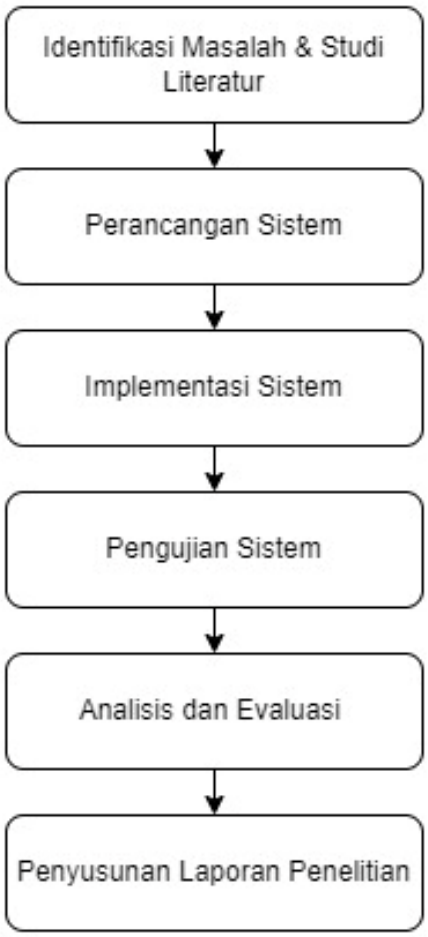
3.4. Metodologi Penelitian

Penelitian ini menggunakan pendekatan kuantitatif, dengan fokus pada proses pengumpulan serta analisis data numerik dan visual guna memperoleh hasil yang objektif dan terukur. Data numerik berupa informasi dari sensor suhu, kelembaban, jarak, dan gerakan, yang akan dikirim ke basis data (database) iot_db. Data ini kemudian diolah menggunakan algoritma Isolation Forest untuk mengidentifikasi anomali lingkungan yang dapat mengindikasikan potensi serangan hama. Secara bersamaan, data visual berupa citra yang diambil oleh kamera ESP32-CAM akan disimpan di Google Drive sebelum dianalisis. Analisis ini menggunakan algoritma YOLOv5 untuk mendeteksi objek hama atau tanda-tanda serangan pada tanaman cabai merah. Pendekatan gabungan ini memungkinkan sistem melakukan deteksi dini dari dua aspek berbeda, lingkungan (melalui data sensor) dan visual (melalui analisis gambar).

Untuk memfasilitasi pemantauan real-time, sistem diintegrasikan dengan bot Telegram yang berfungsi sebagai media notifikasi otomatis. Bot ini akan mengirimkan data sensor, skor anomali dari Isolation Forest, dan hasil deteksi visual langsung kepada pengguna. Integrasi ini memungkinkan petani atau pengguna untuk memantau kondisi tanaman secara cepat dan praktis dari jarak jauh. Pendekatan kuantitatif ini juga memungkinkan evaluasi objektif terhadap kinerja sistem. Untuk menilai tingkat keandalan sistem dalam mendeteksi anomali dan hama, digunakan beberapa metrik evaluasi, yaitu accuracy, precision, recall, dan F1-score yang berfungsi mengukur performa dan efektivitas sistem secara menyeluruh. Dengan metode ini, diharapkan penelitian dapat memberikan solusi pengendalian hama yang lebih efisien, responsif, dan berbasis teknologi.

3.5. Tahap Penelitian

Rangkaian tahapan dalam penelitian ini dirancang secara berurutan dan sistematis agar proses pengembangan sistem deteksi hama berbasis Internet of Things (IoT) dapat terlaksana sesuai dengan tujuan penelitian. Setiap tahapan memiliki peran penting dalam menghasilkan sistem yang fungsional, efektif, dan dapat diimplementasikan di lingkungan pertanian secara nyata.



Gambar 3.1 Tahap Penelitian

3.5.1. Identifikasi Masalah dan Studi Literatur

Budidaya cabai merah masih menghadapi berbagai permasalahan, terutama disebabkan oleh serangan hama seperti ulat, kutu daun, dan lalat buah. Keterlambatan dalam mendeteksi kehadiran hama menyebabkan penurunan hasil panen yang signifikan, menimbulkan kerugian ekonomi, serta meningkatkan penggunaan pestisida secara berlebihan yang berdampak negatif bagi lingkungan dan kesehatan manusia.

Sebagai solusi terhadap permasalahan tersebut, penelitian ini mengadopsi pendekatan Internet of Things (IoT) untuk memungkinkan pemantauan kondisi tanaman secara real-time. Sistem yang dirancang menggunakan beberapa jenis sensor, yaitu sensor DHT11, berfungsi untuk membaca suhu dan kelembapan udara, sensor PIR dan sensor ultrasonik, digunakan untuk mendeteksi pergerakan atau aktivitas yang mencurigakan di sekitar tanaman, modul ESP32-CAM, dimanfaatkan untuk mengambil gambar kondisi visual tanaman cabai. Data yang diperoleh dari sensor kemudian dianalisis menggunakan algoritma Isolation Forest untuk mendeteksi adanya anomali lingkungan yang dapat menjadi tanda awal serangan hama. Sementara itu, citra yang diambil oleh ESP32-CAM diolah dengan algoritma YOLOv5 untuk mengenali keberadaan hama secara visual. Hasil analisis baik dari data sensor maupun hasil deteksi gambar kemudian dikirimkan kepada pengguna melalui Telegram Bot. Fitur ini berfungsi sebagai sistem notifikasi real-time untuk memberikan peringatan dini dan informasi kondisi tanaman secara cepat, efisien, dan akurat.

3.5.2. Perancangan Sistem

Sistem yang dirancang dalam penelitian ini bertujuan untuk mendeteksi potensi serangan hama pada tanaman cabai merah dengan mengintegrasikan teknologi Internet of Things (IoT). Proses dimulai dengan pengumpulan data dari berbagai perangkat. ESP32 berfungsi untuk membaca data dari sensor suhu, kelembaban (DHT11), gerakan (PIR), dan jarak (ultrasonik). Secara bersamaan, ESP32-CAM mengambil gambar kondisi tanaman cabai secara berkala atau sesuai permintaan pengguna.

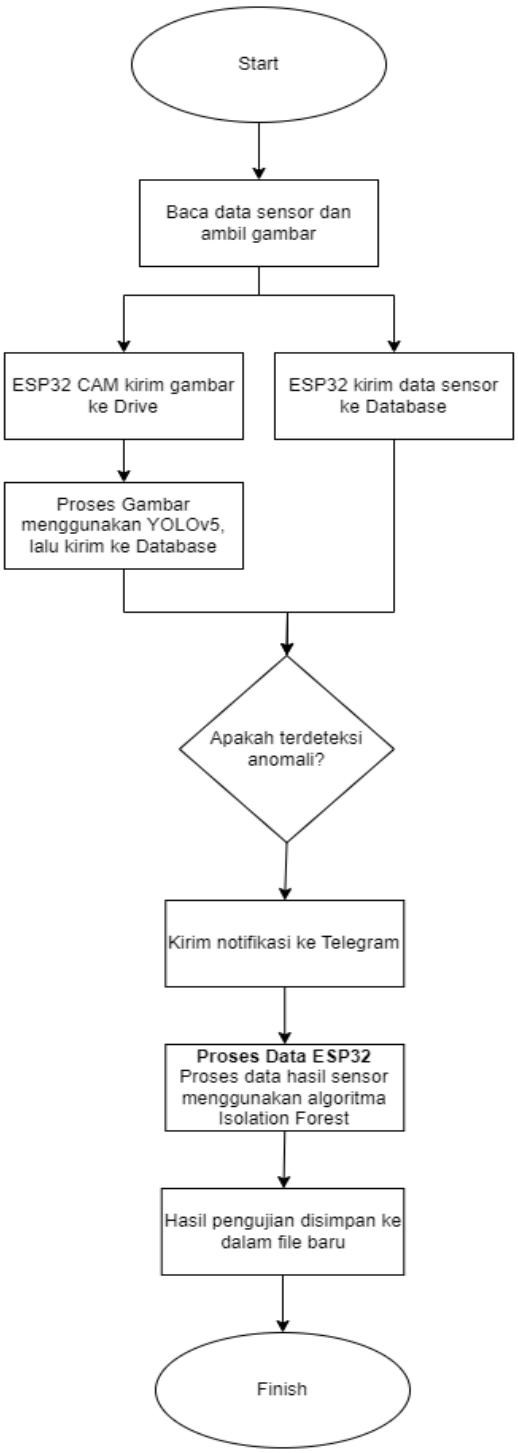
Setelah gambar diambil, data akan diunggah ke Google Drive sebagai tempat penyimpanan sementara. Gambar-gambar ini kemudian dianalisis menggunakan YOLOv5 untuk mendeteksi keberadaan hama atau tanda-tanda visual anomali. Hasil dari deteksi gambar, yang mencakup label objek, bounding box, dan tingkat kepercayaan, berfungsi sebagai informasi awal tentang kondisi tanaman.

Selanjutnya, data sensor dan hasil deteksi gambar dikirim ke server database (iot_db) untuk penyimpanan, sekaligus dikirimkan ke Telegram Bot untuk memberikan notifikasi real-time kepada pengguna. Dengan demikian, petani dapat memantau kondisi tanaman langsung dari perangkat seluler mereka.

Secara paralel, data sensor yang telah tersimpan di server akan diimpor ke Google Colab. Di sana, data tersebut diproses dengan algoritma Isolation Forest untuk mendeteksi pola anomali yang dapat mengindikasikan adanya gangguan lingkungan atau serangan hama.

Akhirnya, hasil dari pengujian yang dilakukan akan tersimpan kedalam file baru sebagai hasil uji data dengan algoritma Isolation Forest. Skor anomali yang

ada akan menunjukkan data yang ada normal atau anomali. Data dengan skor anomali dibawah 0,5 atau mendekati 0,5 dianggap normal, dan data mendekati 1 dianggap anomali.



Gambar 3.2 Perancangan Sistem

3.5.3. Implementasi Sistem

Pada tahap implementasi, perakitan perangkat keras dimulai dengan menghubungkan sensor DHT11, PIR, dan ultrasonik ke mikrokontroler ESP32. Mikrokontroler ini diprogram menggunakan Arduino IDE agar mampu membaca data sensor secara periodik dan mengirimkan hasil pembacaan ke server basis data (iot_db) melalui koneksi Wi-Fi.

Modul ESP32-CAM digunakan untuk mengambil gambar tanaman cabai merah secara otomatis. Gambar yang diperoleh kemudian dikirimkan ke Google Drive dan server. Dari sini, data visual diolah menggunakan algoritma YOLOv5 untuk mendeteksi keberadaan hama dengan menghasilkan bounding box, label objek, dan tingkat kepercayaan (confidence level) pada setiap hasil deteksi.

Secara bersamaan, data sensor yang tersimpan di server diolah menggunakan Google Colab dengan bahasa pemrograman Python dan algoritma Isolation Forest untuk mengidentifikasi pola anomali yang menunjukkan potensi serangan hama. Hasil pengolahan ini kemudian disimpan ke dalam file baru sebagai data evaluasi lanjutan.

3.5.4. Pengujian Sistem

Tahap pengujian dilakukan untuk memastikan bahwa sistem deteksi dini hama berbasis IoT dapat berfungsi dengan baik mulai dari proses pengumpulan data hingga penyampaian notifikasi. Pengujian dilakukan pada seluruh komponen utama sistem, meliputi:

1. Lingkungan Pengujian

Pengujian sistem ini dilakukan di lingkungan yang terkontrol, yaitu di dalam area kost. Keterbatasan akses terhadap lahan pertanian skala besar dan risiko yang

tidak terkontrol membuat pengujian pada area terbatas menjadi pilihan yang paling memungkinkan. Untuk mensimulasikan kondisi lingkungan dinamis, beberapa tanaman cabai merah ditanam dalam pot dan diletakkan dalam area pengujian. Lingkungan ini memungkinkan pengumpulan data sensor (suhu, kelembaban, gerakan, dan jarak) yang mereplikasi fluktuasi kondisi di lapangan.

2. Prosedur Pengujian

Tahapan pengujian sistem terdiri dari beberapa langkah berikut:

a. Pengaktifan Sistem

ESP32 dan ESP32-CAM dihidupkan untuk memulai pembacaan sensor dan pengambilan gambar.

b. Pengumpulan Data Sensor

Data sensor dikirim ke server iot_db, sementara ESP32-CAM mengambil gambar kondisi lingkungan dan tanaman.

c. Pemrosesan Data

Gambar dari Google Drive diproses oleh YOLOv5 untuk deteksi hama, sedangkan data sensor dianalisis oleh Isolation Forest di Google Colab untuk mendeteksi anomali.

d. Pengujian Notifikasi Telegram

Sistem diuji untuk memastikan hasil deteksi dari YOLOv5 dan Isolation Forest berhasil dikirimkan ke pengguna melalui Telegram Bot.

e. Uji Ketahanan Sistem

Pengujian dilakukan selama beberapa hari dengan kondisi lingkungan bervariasi untuk menilai kinerja dan stabilitas perangkat keras di lapangan.

3. Parameter yang Diuji

Prameter pengujian pada penelitian ini meliputi, sebagai berikut:

a. Akurasi Deteksi Anomali

Seberapa baik Isolation Forest membedakan data normal dari anomali..

b. Respons Notifikasi

Kinerja dalam mengidentifikasi hama.

c. Konsistensi Pembacaan Sensor

Stabilitas pembacaan dari setiap sensor.

d. Keberlanjutan Operasional

Kemampuan sistem untuk beroperasi tanpa henti.

4. Teknik Evaluasi

Evaluasi hasil dilakukan dengan membandingkan hasil deteksi dari sistem terhadap kondisi aktual di lapangan, serta menghitung:

a. Precision, recall, dan F1-score untuk mengukur kinerja deteksi anomali dan visual.

b. Rata-rata waktu notifikasi dari saat deteksi hingga pesan diterima.

c. Jumlah kegagalan sensor atau komunikasi untuk menilai stabilitas sistem.

3.5.5. Analisis dan Evaluasi

Setelah sistem deteksi hama berbasis IoT selesai diterapkan dan diuji, dilakukan analisis untuk menilai kinerja sistem dalam mendeteksi anomali serta mengevaluasi keandalan sistem secara keseluruhan. Evaluasi difokuskan pada beberapa aspek berikut:

1. Analisis Akurasi Deteksi Anomali

Evaluasi ini mengukur kemampuan sistem dalam mendeteksi data sensor yang menyimpang dari kondisi normal menggunakan Isolation Forest dan mengidentifikasi hama pada gambar.

Rumus Skor anomali Isolation Forest:

$$s(x) = 2^{(-\frac{E(h(x))}{c(n)})}$$

Keterangan:

s(x) = Skor anomali untuk data x.

E(h(x)) = Rata-rata panjang jalur isolasi untuk data x di semua pohon isolasi.

c(n) = Konstanta normalisasi, tergantung jumlah data n, dihitung dengan:

$$c(n) = 2^{\frac{(\ln(n-1)+\gamma)-(2(n-1)}{n}}$$

$$\gamma \approx 0{,}5772$$

Agar proses penentuan status anomali dari data sensor lebih mudah dipahami, berikut disajikan kriteria yang digunakan sistem sebagai dasar dalam menentukan apakah data termasuk kondisi normal atau berpotensi sebagai indikasi serangan hama.

2. Evaluasi Respon Sistem

Evaluasi mencakup kecepatan sistem dalam mengirim notifikasi ke Telegram, ketepatan isi notifikasi, serta durasi waktu dari pembacaan sensor hingga pesan diterima oleh pengguna.

3. Konsistensi Penyimpanan Data

Data disimpan dalam format CSV dengan struktur, Waktu, Suhu, Jarak, Gerakan, Status Anomali. Evaluasi ini memastikan data tercatat secara lengkap, konsisten, dan dapat diakses kapan saja.

4. Visualisasi Hasil

Hasil pengujian divisualisasikan dalam bentuk grafik agar memudahkan analisis, dengan titik anomali ditandai secara khusus untuk memperlihatkan kapan dan dalam kondisi apa anomali terdeteksi.

5. Evaluasi Fungsional Sistem

Pengujian fungsional memastikan setiap komponen sistem beroperasi sesuai dengan rancangan: sensor membaca data dengan stabil, ESP32 dan ESP32-CAM berhasil mengirimkan data ke server, algoritma YOLOv5 dan Isolation Forest berjalan secara otomatis, dan bot Telegram merespons dengan akurat.

3.5.6. Penyusunan Laporan

Tahap akhir penelitian adalah penyusunan laporan yang sistematis dan terorganisir. Laporan ini mendokumentasikan setiap tahapan penelitian, mulai dari identifikasi masalah, perancangan, implementasi, pengujian, hingga analisis hasil.

Laporan ini dibuat untuk menyampaikan temuan secara komprehensif dan dapat dipertanggungjawabkan secara ilmiah. Penggunaan aplikasi pengelola referensi seperti Mendeley membantu memastikan pengelolaan sitasi dan daftar pustaka sesuai standar akademis. Laporan akhir diharapkan dapat menjadi sumber referensi untuk pengembangan sistem IoT berbasis machine learning dan deep learning dalam deteksi hama pertanian di masa depan, serta berkontribusi pada praktik pertanian cerdas yang lebih modern, efisien, dan berkelanjutan.

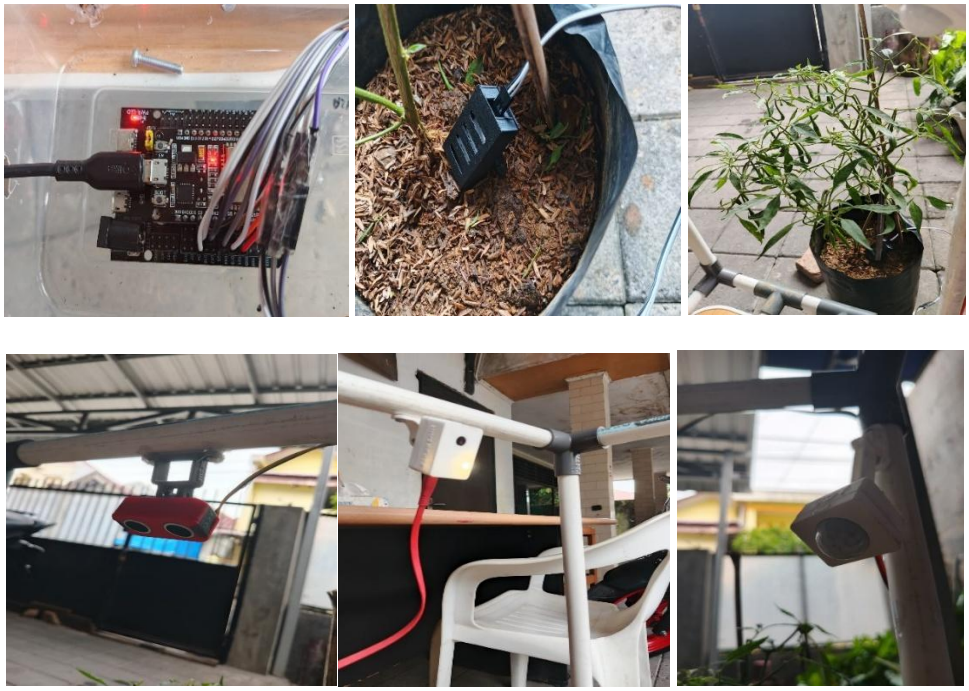
BAB IV

HASIL DAN PEMBAHASAN

4.1. Rangkaian Sistem

4.1.1. Sensor dan Komponen yang digunakan

Pada penelitian ini sistem deteksi anomali tanaman cabai merah menggunakan beberapa perangkat keras utama berupa mikrokontroler dan sensor. Mikrokontroler yang digunakan adalah ESP32 sebagai pusat pengendali, sedangkan sensor yang dipasang terdiri dari sensor jarak, sensor suhu, sensor kelembapan, dan sensor gerak. Selain itu, digunakan pula modul ESP32-CAM yang berfungsi untuk mengambil gambar kondisi tanaman secara visual. Seluruh sensor ini digunakan untuk memantau lingkungan tanaman cabai merah agar data yang diperoleh dapat diolah lebih lanjut dengan algoritma Isolation Forest.



Gambar 4.1 Alat Uji Coba

Setelah seluruh komponen siap, tahap berikutnya adalah melakukan perakitan menjadi sebuah prototype sistem. Proses perakitan dilakukan dengan menyusun sensor jarak, suhu, kelembapan, gerak, dan modul ESP32-CAM ke dalam satu kesatuan yang terhubung dengan mikrokontroler ESP32. Prototype ini menunjukkan bahwa seluruh komponen telah terintegrasi dengan baik dan siap digunakan untuk pengujian dalam pengambilan data sensor maupun pengambilan gambar tanaman cabai merah melalui ESP32-CAM.

Gambar 4.2 Prototype Sistem

4.2. Uji Coba Sistem

4.2.1. Langkah Uji Coba

Pengujian sistem deteksi anomali pada tanaman cabai merah ini dilakukan setelah perakitan perangkat keras yang terdiri dari mikrokontroler ESP32, ESP32-CAM, serta berbagai sensor (DHT11, Ultrasonik, dan PIR). Tujuan utama pengujian ini adalah memastikan sistem berfungsi optimal dalam mendeteksi anomali, baik melalui data sensor maupun bukti visual.

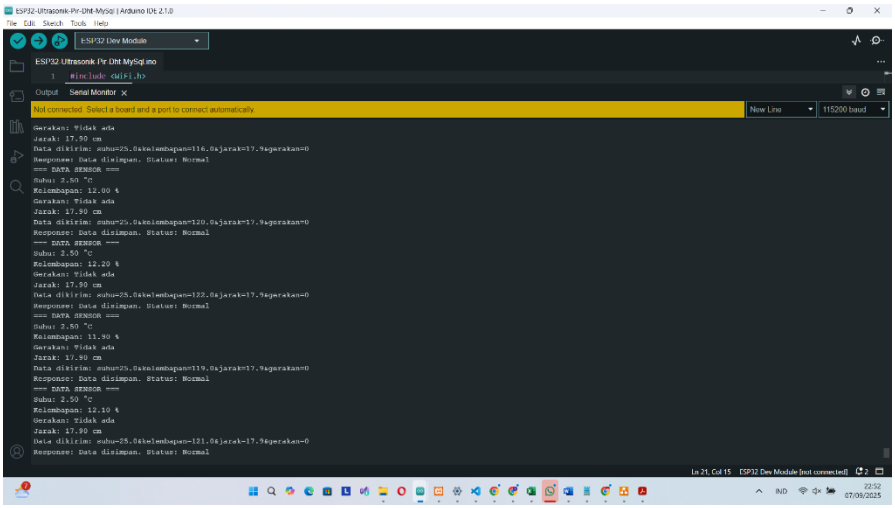
Proses uji coba diawali dengan mengumpulkan data dari sensor-sensor yang terhubung ke ESP32. Data tersebut dikirim secara real-time dan disimpan dalam format CSV untuk memudahkan pengolahan lebih lanjut. Data yang terkumpul kemudian diproses menggunakan algoritma Isolation Forest di Google Colab. Tahapan preprocessing, seperti normalisasi dan penanganan nilai kosong, dilakukan sebelum algoritma dijalankan untuk mengidentifikasi data normal dan anomali berdasarkan distribusi nilai keempat sensor.

Selain pemrosesan data sensor, sistem juga menggunakan ESP32-CAM untuk mengambil gambar kondisi visual tanaman secara bersamaan. Foto-foto ini berfungsi sebagai dokumentasi dan bukti pendukung apabila anomali terdeteksi, memungkinkan analisis yang lebih mendalam. Dengan mengintegrasikan data sensor dan visual, sistem mampu memberikan informasi yang lebih komprehensif tentang kondisi tanaman cabai.

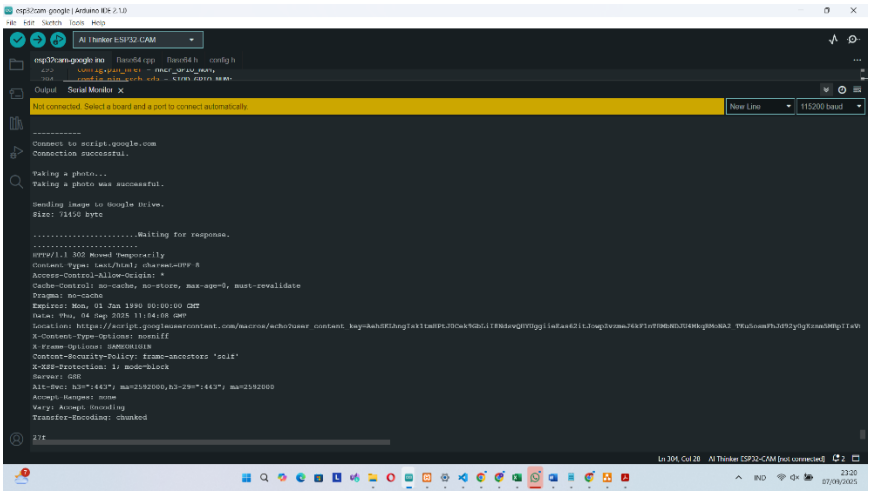
Hasil pengujian menunjukkan bahwa algoritma Isolation Forest berhasil mengidentifikasi anomali secara efektif. Hal ini terlihat dari skor deteksi, di mana skor negatif menunjukkan anomali dan skor positif menunjukkan kondisi normal. Evaluasi lebih lanjut menggunakan metrik Precision, Recall, dan F1-score menegaskan bahwa sistem memiliki akurasi yang baik dalam membedakan data normal dan anomali. Grafik distribusi skor anomali dan visualisasi prediksi semakin memperkuat bahwa sistem ini berfungsi sesuai harapan..

4.2.2. Uji Coba Sensor

Tahap uji coba sensor bertujuan untuk memastikan sensor yang ada dibuat sesuai dengan yang diharapkan., dengan output yang dihasilkan dari sensor dapat diamati dengan memanfaatkan tools yang ada pada Arduino IDE.



Gambar 4.3 Pengujian ESP32 dengan sensor DHT11, Ultrasonik, dan PIR di Arduino IDE



Gambar 4.4 Pengujian ESP32 CAM Pada Arduino IDE

Dari Gambar diatas, dapat kita amati output yang dihasilkan dari sensor yang dilakukan di Arduino IDE, dimana suhu, kelembapan, jarak, dan gerak, serta gambar dapat terdeteksi.

Tabel 4.1 Data Pengamatan ESP32

No.	Kode_id	Jarak	Suhu	Kelembapan	Gerak	Anomali	Waktu
1	1534	49.23	22	85	0	0	03/09/2025 16:22
2	1535	49.23	25	81	0	0	03/09/2025 16:22
3	1536	49.23	27	78	0	0	03/09/2025 16:22
4	1537	49.23	29	75	0	0	03/09/2025 16:22

5	1538	49.23	31	72	0	0	03/09/2025 16:22
6	1544	49.57	20	92	0	0	03/09/2025 16:24
7	1545	49.23	22	89	0	0	03/09/2025 16:24
8	1546	49.23	21	91	0	0	03/09/2025 16:25
9	1547	49.23	24	90	0	0	03/09/2025 16:25
.....	.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....	.....
1361	3379	16.9	29	110	0	0	07/09/2025 17:54
1362	3380	16.9	30	111	0	0	07/09/2025 17:54

Tabel 4.2 Data Pengamatan ESP32-CAM

No.	Kode IDE	Nama File	Waktu	Keterangan
1	4183	20250903-164046.jpg	08/09/2025 05:52	Tidak Anomali
2	4184	20250903-164110.jpg	08/09/2025 05:52	Tidak Anomali
3	4185	20250903-165633.jpg	08/09/2025 05:52	Tidak Anomali
4	4186	20250903-165711.jpg	08/09/2025 05:53	Tidak Anomali
5	4187	20250903-165737.jpg	08/09/2025 05:53	Tidak Anomali
6	4188	20250903-165750.jpg	08/09/2025 05:53	Tidak Anomali
7	4189	20250903-165811.jpg	08/09/2025 05:53	Tidak Anomali
8	4190	20250903-165830.jpg	08/09/2025 05:53	Tidak Anomali
9	4191	20250903-165851.jpg	08/09/2025 05:53	Tidak Anomali
.....	.....	.....	.....	.....
.....	.....	.....	.....	.....
601	4783	20250907-175346.jpg	08/09/2025 06:22	Tidak Anomali
602	4784	20250907-175410.jpg	08/09/2025 06:22	Tidak Anomali

Perhitungan manual skor anomali:

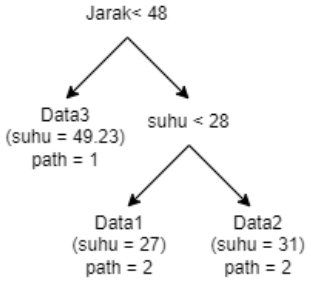
No	Jarak	Suhu	Kelembapan	Gerak
1.	49.23	22	85	0
2.	49.23	25	81	0

3.	49.23	27	78	0

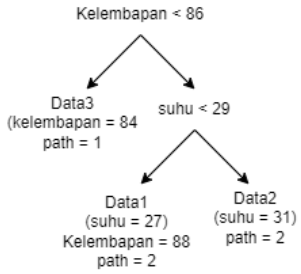
a. Buat Pohon Isolasi

Pohon (Menggunakan Suhu dan Kelembapan)

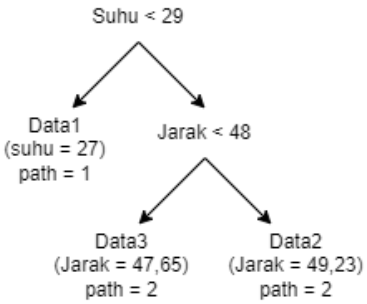
Pohon 1



Pohon 2



Pohon 3



Pohon 1

1552 = path = 2

1553 = path = 2

1554 = path = 1

Pohon 2

1552 = path = 2

1553 = path = 2

1554 = path = 1

Pohon 3

1552 = path = 1

1553 = path = 2

1554 = path = 2

E(h(x)) = $\frac{Junlah\ path\ di\ semua\ pohon}{jumlah\ pohon}$

E(h(x)) = $\frac{h1+h2+h3}{t}$

1552 = E(h(x)) = $\frac{2+2+1}{3} = 1{,}667$

1553 = E(h(x)) = $\frac{2+2+2}{3} = 2$

1554 = E(h(x)) = $\frac{1+1+2}{3} = 1{,}333$

b. Perhitungan c(n)

n = 3, maka:

$c(n) = 2H(n-1) - \frac{2(n-1)}{n}$

$H(2) = 1 + 1/2 = 1{,}5$

$c(3) = 2(1{,}5) - \frac{2(2)}{3}$

c(n) =c(n) = 1,667.

c. Rata-rata path dari Google Colab

Dari hasil model (pohon acak), diperoleh:

Data 1 = E(h(x)) = 1,667.

Data 2 = E(h(x)) = 2

Data 3 = E(h(x)) = 1,333.

d. Perhitungan Skor Anomali (s(x))

$s(x) = 2^{(-\frac{E(h(x))}{c(n)})}$

Data 1= $s(x) = 2^{\left(-\frac{1,667}{1,667}\right)} = 0,5$ = Normal

Data 1= $s(x) = 2^{\left(-\frac{2}{1,667}\right)} = 0,435$ = Lebih Normal

Data 1= $s(x) = 2^{\left(-\frac{1,333}{1,667}\right)} = 0,574$ = Anomali

Berdasarkan ketentuan yang ada, dari hasil yang didapat, disimpulkan terdapat 2 data normal dan 1 data anomali.

4.2.3. Implementasi Isolation Forest

Langkah awal yang dilakukan adalah pra-pemrosesan data sensor. Data yang diperoleh dari pengukuran sensor pada tanaman cabai merah terdiri atas empat fitur, yaitu jarak, suhu, kelembapan, dan gerakan. Data ini terlebih dahulu diperiksa untuk memastikan tidak ada nilai kosong. Jika terdapat data yang hilang, nilainya diisi menggunakan median dari kolom masing-masing agar tetap konsisten. Setelah itu, data dinormalisasi menggunakan StandardScaler sehingga seluruh fitur berada pada skala yang sama. Normalisasi ini penting dilakukan karena algoritma Isolation Forest cukup sensitif terhadap perbedaan skala antarfitur.

```
# PREPROCESSING
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler

data = pd.read_csv("/content/data cabai.csv", delimiter=";")
features = ["jarak", "suhu", "kelembapan", "gerak"]
X = data[features].copy()

# Isi nilai kosong dengan median
X = X.fillna(X.median(numeric_only=True))

# Normalisasi
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Gambar 4.5 Koding Processing Isolation Forest

Setelah tahap preprocessing, dilakukan pelatihan model Isolation Forest. Algoritma ini dipilih karena mampu mendeteksi anomali tanpa membutuhkan data berlabel (unsupervised learning). Parameter yang digunakan dalam penelitian adalah contamination = 0,1, dengan asumsi sekitar 10% dari dataset merupakan data anomali. Selanjutnya, model membangun sejumlah pohon isolasi yang digunakan untuk menentukan tingkat kemudahan suatu titik data dipisahkan dari data lain. Titik yang mudah diisolasi umumnya dianggap sebagai anomali, sedangkan titik yang membutuhkan pemisahan lebih panjang cenderung dikategorikan sebagai normal.

```
# PELATIHAN MODEL
from sklearn.ensemble import IsolationForest

iso = IsolationForest(
    contamination=0.1,  # asumsi 10% data anomali
    random_state=42,
    n_jobs=-1
)
iso.fit(X_scaled)
```

Gambar 4.6 Koding Pelatihan Model Isolation Forest

Tahap berikutnya adalah perhitungan skor anomali. Pada tahap ini, Isolation Forest menghasilkan dua jenis skor: score_samples, yaitu skor mentah di mana nilai lebih kecil menunjukkan kemungkinan anomali lebih tinggi, dan decision_function, yaitu skor yang telah dinormalisasi dengan nilai negatif sebagai indikasi anomali. Selain itu, fungsi predict() dimanfaatkan untuk menghasilkan label praktis: -1 untuk anomali dan 1 untuk normal. Agar lebih mudah dianalisis, label tersebut diubah ke bentuk biner: 1 untuk anomali dan 0 untuk normal. Seluruh hasil perhitungan kemudian disimpan kembali ke dalam dataset dalam bentuk skor dan label, yang selanjutnya dapat diekspor ke file CSV untuk kepentingan analisis lanjutan maupun dikirimkan melalui sistem notifikasi Telegram Bot.

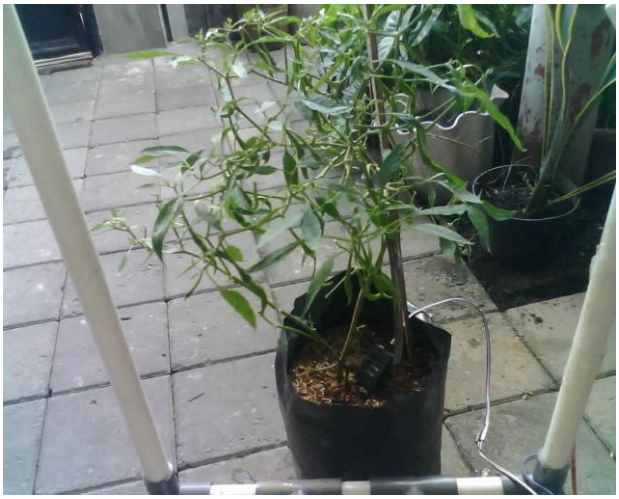
```
# SKOR ANOMALI
data["anomaly_score_raw"] = iso.score_samples(X_scaled)
data["anomaly_score_decision"] =
iso.decision_function(X_scaled)

# Prediksi label praktis
y_pred = iso.predict(X_scaled)   # -1 = anomali, 1 = normal
data["predicted_anomaly"] = np.where(y_pred == -1, 1, 0)

# Simpan hasil
out_cols = features + ["anomaly_score_raw",
"anomaly_score_decision", "predicted_anomaly"]
data[out_cols].to_csv("/content/hasil_isolation_forest_scores.
csv", index=False)
print(data[out_cols].head(10))
```

Gambar 4.7 Koding Skor Anomali Isolation Forest

Berbeda dengan perhitungan manual, Isolation Forest pada scikit-learn secara otomatis menghitung skor anomali. Mekanisme perhitungan berbasis pohon isolasi dilakukan di dalam algoritma sehingga peneliti hanya perlu memanggil fungsi bawaan untuk memperoleh hasil. Dengan cara ini, proses deteksi anomali dapat dilakukan dengan lebih efisien, praktis, dan tetap akurat, tanpa harus menyertakan rumus matematis secara manual.

4.2.4. Hasil Dokumentasi ESP32 CAM

Pada sistem deteksi anomali tanaman cabai merah ini digunakan modul ESP32-CAM yang berfungsi sebagai kamera untuk mengambil gambar kondisi tanaman secara visual. Pengambilan gambar dilakukan secara bersamaan dengan proses pencatatan data sensor, sehingga setiap kali sistem mendeteksi adanya anomali, dapat diperoleh bukti pendukung berupa foto kondisi tanaman pada saat itu.



Gambar 4.8 Hasil Dokumentasi ESP32 CAM

Hasil foto tanaman cabai merah dari ESP32-CAM. Dokumentasi visual ini berfungsi sebagai bukti tambahan yang memperkuat data sensor. Dengan adanya foto, pengguna dapat melakukan verifikasi langsung terhadap kondisi tanaman, apakah benar terjadi penyimpangan atau gangguan yang menyebabkan data anomali. Integrasi data sensor dengan bukti visual ini membuat sistem lebih komprehensif, karena tidak hanya mendeteksi anomali berdasarkan angka sensor, tetapi juga memberikan gambaran nyata mengenai keadaan tanaman.

4.3. Hasil Pengujian Deteksi Anomali

4.3.1. Tabel Hasil Deteksi

Dari hasil uji coba yang telah dilakukan bertujuan untuk menguji sistem deteksi anomali pada tanaman cabai merah berbasis IoT dengan kombinasi sensor jarak, suhu, kelembapan, gerak, serta modul kamera ESP32-CAM. Data sensor yang diperoleh kemudian diolah menggunakan algoritma Isolation Forest, dan setelah dilakukan analisis, sistem menghasilkan akurasi sebesar 95%. Hasil ini menunjukkan bahwa secara umum sistem mampu membedakan data normal dan anomali dengan baik.

Adapun hasil dari uji coba sensor yang telah dilakukan dapat dilihat pada tabel dibawah ini.

Tabel 4.3 Hasil Uji Coba ESP32

No.	Kode id	Jarak	Suhu	Kelembapan	Gerak	Anomali	Waktu	Predicted Anomaly	Anomaly Score
1	1534	49.23	22	85	0	0	03/09/2025 16:22	0	0.0713
2	1535	49.23	25	81	0	0	03/09/2025 16:22	0	0.0985
3	1536	49.23	27	78	0	0	03/09/2025 16:22	0	0.0836
4	1537	49.23	29	75	0	0	03/09/2025 16:22	0	0.0383
5	1538	49.23	31	72	0	0	03/09/2025 16:22	0	0.0031
6	1544	49.57	20	92	0	0	03/09/2025 16:24	0	0.0402
7	1545	49.23	22	89	0	0	03/09/2025 16:24	0	0.0694
8	1546	49.23	21	91	0	0	03/09/2025 16:25	0	0.0569
9	1547	49.23	24	90	0	0	03/09/2025 16:25	0	0.0900
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....
1361	3379	16.9	29	110	0	0	07/09/2025 17:54	0	0.1380
1362	3380	16.9	30	111	0	0	07/09/2025 17:54	0	0.1437

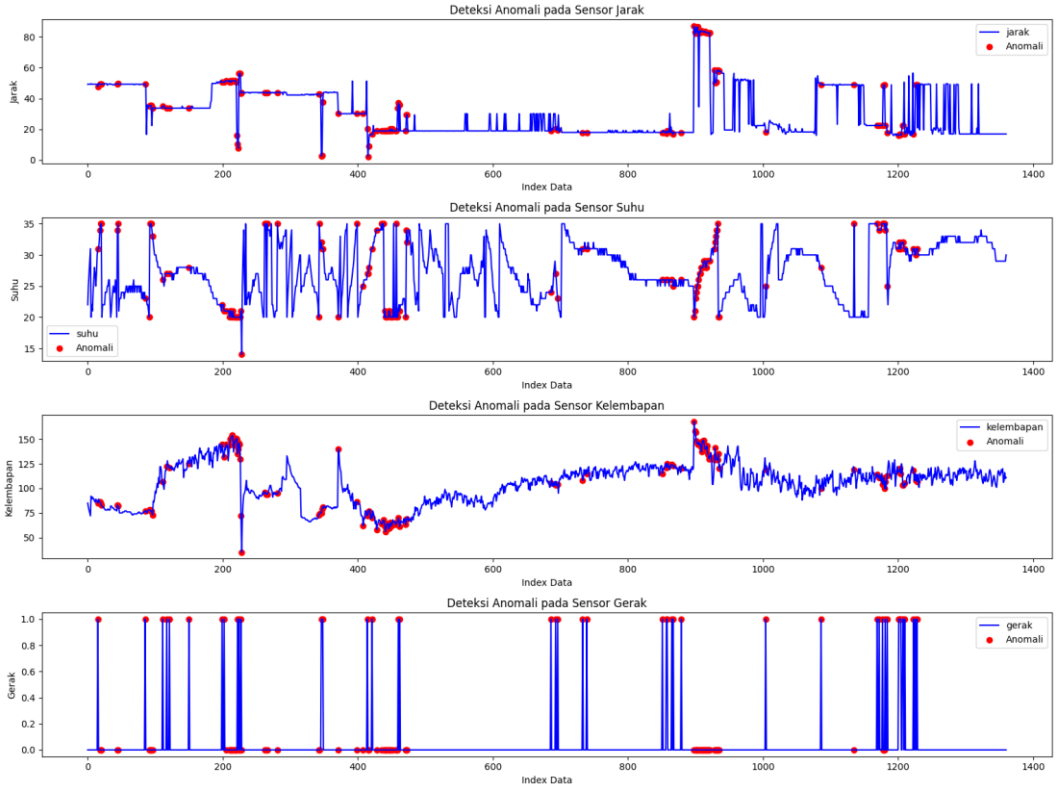
Adapun permasalahan yang muncul selama pengujian adalah kecenderungan sistem menghasilkan false positive, yaitu data normal yang salah dikenali sebagai anomali. Hal ini menyebabkan nilai precision pada kelas anomali menjadi rendah. Faktor ini dapat disebabkan oleh keterbatasan data sensor yang digunakan, fluktuasi lingkungan yang cepat berubah, atau sensitivitas algoritma Isolation Forest yang sangat peka terhadap perubahan nilai data. Selain itu, pengambilan data sensor di lapangan dipengaruhi oleh kondisi eksternal seperti

variasi suhu dan kelembapan yang tidak selalu stabil, sehingga menyebabkan beberapa data normal ikut terklasifikasi sebagai anomali.

Meskipun demikian, sistem ini tetap menunjukkan performa yang baik karena tidak ada data anomali yang terlewat, sehingga dapat dikatakan sistem lebih aman untuk mendukung monitoring kondisi tanaman. Integrasi dengan ESP32-CAM juga menambah keandalan sistem, karena setiap data sensor yang terindikasi anomali dapat didukung dengan bukti visual berupa foto kondisi tanaman. Dengan demikian, sistem deteksi anomali ini sudah mampu bekerja sesuai fungsinya meskipun masih terdapat beberapa keterbatasan yang perlu diperbaiki pada tahap pengembangan selanjutnya.

4.3.2. Visualisasi Data Sensor

Hasil deteksi anomali yang diperoleh dari algoritma Isolation Forest kemudian divisualisasikan dalam bentuk grafik untuk masing-masing sensor. Grafik ini menampilkan data sensor jarak, suhu, kelembapan, dan gerak, dengan titik-titik berwarna merah yang menandakan adanya anomali. Visualisasi ini sangat membantu dalam memahami pola data serta membandingkan kondisi normal dengan kondisi yang terdeteksi sebagai anomali.

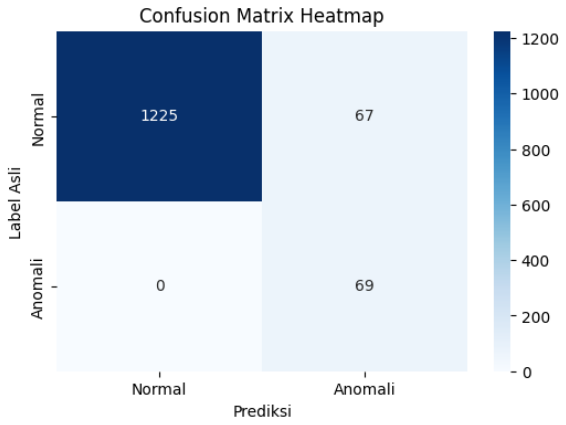


Gambar 4.9 Visualisasi Data Sensor

Pada grafik sensor jarak, terlihat bahwa data sensor yang berada jauh dari pola umum ditandai sebagai anomali, misalnya ketika jarak terdeteksi terlalu kecil atau terlalu besar secara tiba-tiba. Selanjutnya, memperlihatkan visualisasi data sensor suhu, fluktuasi suhu yang drastis, baik penurunan maupun kenaikan mendadak, berhasil dikenali sebagai anomali oleh sistem. Visualisasi sensor kelembapan menunjukkan, data kelembapan yang keluar dari pola normal, misalnya penurunan tajam dalam waktu singkat, teridentifikasi sebagai anomali. Sementara itu, pada grafik sensor gerak, setiap adanya pergerakan di sekitar tanaman, sensor memberikan respon yang ditandai sebagai anomali, karena kondisi ini dianggap menyimpang dari keadaan normal tanpa adanya gangguan.

4.3.3. Evaluasi Data Sensor

Evaluasi kinerja sistem deteksi anomali tanaman cabai merah dilakukan dengan menggunakan confusion matrix dan classification report yang mencakup metrik precision, recall, dan F1-score.



Gambar 4.10 Confusion Matrix

Berdasarkan confusion matrix, sistem berhasil mengklasifikasikan 1.225 data normal secara tepat, sementara terdapat 67 data normal yang salah terdeteksi sebagai anomali (false positive). Untuk kelas anomali, sistem berhasil mendeteksi seluruh 69 data anomali dengan benar (true positive) tanpa ada anomali yang terlewat (false negative = 0).

```
Confusion Matrix (angka):
[[1225   67]
 [   0   69]]

Classification Report:
              precision    recall  f1-score   support

      Normal       1.00      0.95      0.97      1292
     Anomali       0.51      1.00      0.67        69

    accuracy                           0.95      1361
   macro avg       0.75      0.97      0.82      1361
weighted avg       0.98      0.95      0.96      1361
```

Gambar 4.11 Classification Report

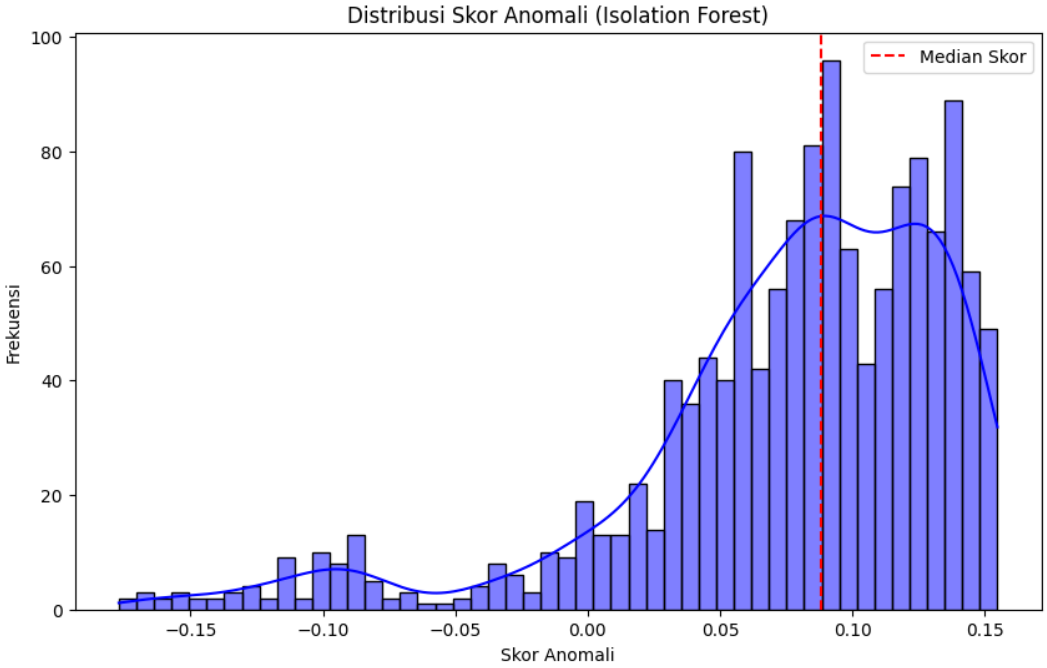
Hasil classification report menunjukkan bahwa sistem memiliki akurasi sebesar 95%, yang berarti sebagian besar data dapat diklasifikasikan dengan benar. Nilai recall pada kelas Anomali mencapai 100%, sehingga sistem mampu mendeteksi seluruh anomali dengan sangat baik. Hal ini penting karena dalam kasus monitoring tanaman, setiap anomali yang terlewat dapat berakibat pada kegagalan deteksi masalah pada tanaman.

Namun, nilai precision pada kelas Anomali masih cukup rendah yaitu 51%, yang menunjukkan adanya sejumlah data normal yang terdeteksi sebagai anomali (false positive). Kondisi ini menyebabkan sistem terkadang memberikan peringatan anomali meskipun kondisi sebenarnya normal. Meskipun demikian, trade-off ini masih dapat diterima karena lebih baik sistem memberikan peringatan berlebih dibandingkan gagal mendeteksi adanya anomali yang benar-benar terjadi.

Dengan demikian, hasil evaluasi ini membuktikan bahwa algoritma Isolation Forest yang digunakan dalam penelitian mampu bekerja dengan baik dalam mendeteksi anomali pada tanaman cabai merah, meskipun masih terdapat ruang perbaikan pada aspek precision untuk mengurangi jumlah false positive.

4.3.4. Distribusi Skor Anomali

Selain confusion matrix, hasil deteksi anomali juga dapat dianalisis melalui distribusi skor yang dihasilkan oleh algoritma Isolation Forest. Setiap data sensor diberikan skor yang menunjukkan tingkat keunikan atau penyimpangan dari pola umum. Skor dengan nilai negatif menunjukkan bahwa data tersebut cenderung anomali, sedangkan skor dengan nilai positif menunjukkan bahwa data tersebut tergolong normal.



Gambar 4.12 Distribusi Skor Anomali

Distribusi skor anomali ditampilkan dalam bentuk histogram. Dari grafik tersebut terlihat bahwa sebagian besar data memiliki skor positif yang menandakan kondisi normal. Namun terdapat pula sejumlah data dengan skor negatif yang relatif sedikit jumlahnya, dan data inilah yang dikategorikan sebagai anomali. Garis putus-putus merah menunjukkan nilai median skor, yang dapat digunakan sebagai pembatas antara data normal dan anomali.

Dengan adanya distribusi ini, analisis anomali tidak hanya bergantung pada label prediksi semata, tetapi juga memperlihatkan sebaran skor yang lebih detail. Hal ini memberikan gambaran mengenai tingkat keyakinan model dalam membedakan data normal dan anomali pada sensor tanaman cabai merah.

4.4. Pembahasan

Hasil pengujian sistem deteksi anomali tanaman cabai merah menunjukkan bahwa algoritma Isolation Forest mampu bekerja dengan baik dalam mengidentifikasi data yang menyimpang dari kondisi normal. Hal ini ditunjukkan dengan capaian recall yang sangat tinggi pada kelas anomali (100%), artinya seluruh data anomali berhasil terdeteksi tanpa ada satupun yang terlewat. Kondisi ini merupakan keunggulan penting dalam sistem monitoring, karena setiap anomali yang muncul dapat segera diketahui dan ditindaklanjuti.

Namun demikian, sistem masih memiliki kelemahan pada sisi precision yang relatif rendah. Hal ini berarti terdapat data normal yang salah terdeteksi sebagai anomali (false positive). Beberapa faktor yang diduga menjadi penyebab terjadinya error antara lain adanya fluktuasi pada data sensor, gangguan noise lingkungan seperti perubahan suhu atau kelembapan secara mendadak, serta sensitivitas model Isolation Forest yang tinggi terhadap variasi data.

Meskipun begitu, secara keseluruhan sistem tetap memiliki keunggulan karena tidak ada anomali yang terlewat sehingga lebih aman digunakan dalam konteks pemantauan tanaman. Sistem ini bersifat konservatif dengan memberikan peringatan meskipun dalam beberapa kasus kondisi sebenarnya normal. Hal ini lebih menguntungkan daripada risiko kehilangan deteksi anomali yang sesungguhnya.

Selain itu, integrasi dengan ESP32-CAM meningkatkan tingkat keandalan sistem. Hasil foto yang diambil dapat dijadikan bukti visual pendukung dari deteksi anomali, sehingga memudahkan proses verifikasi kondisi tanaman di lapangan.

Dengan adanya data sensor dan dokumentasi visual, sistem menjadi lebih komprehensif dalam memberikan informasi.

Secara umum, dapat disimpulkan bahwa sistem yang dirancang telah berhasil sesuai dengan tujuan penelitian, yaitu mendeteksi anomali pada tanaman cabai merah secara otomatis berbasis IoT. Meskipun masih terdapat keterbatasan pada precision, hal ini dapat diperbaiki di masa mendatang melalui optimasi parameter model, teknik pra-pemrosesan data yang lebih baik, atau integrasi dengan metode lain untuk menekan jumlah false positive.

BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan uraian-uraian yang telah dijelaskan pada bab-bab sebelunya, maka dapat diambil kesimpulan. Sistem yang dirancang telah berhasil mendeteksi anomali dengan memanfaatkan sensor jarak, suhu, kelembapan, dan gerak yang terhubung pada mikrokontroler ESP32. Data sensor yang diperoleh dikirimkan secara real-time dan diolah menggunakan Google Colab, dengan algoritma Isolation Forest sebagai metode deteksi utama.

Hasil pengujian menunjukkan bahwa algoritma Isolation Forest mampu membedakan data normal dan data anomali dengan tingkat akurasi sebesar 95%. Recall pada kelas anomali mencapai 100%, yang berarti sistem tidak melewatkan satu pun data anomali. Namun demikian, precision pada kelas anomali masih rendah, yaitu 48%, yang menunjukkan masih terdapat data normal yang salah diklasifikasikan sebagai anomali (false positive). Meskipun demikian, hal ini menunjukkan bahwa sistem lebih menekankan pada keamanan deteksi, karena semua anomali berhasil ditangkap.

Distribusi skor anomali yang diperoleh memperlihatkan bahwa data dengan skor negatif lebih cenderung dianggap anomali, sedangkan data dengan skor positif dikategorikan sebagai normal. Hal ini sesuai dengan teori Isolation Forest yang mendasarkan deteksinya pada panjang path isolasi data. Selain itu, integrasi ESP32-CAM memberikan nilai tambah bagi sistem, karena mampu mendokumentasikan kondisi visual tanaman cabai merah pada saat terdeteksi anomali. Dengan adanya

bukti visual ini, sistem tidak hanya mengandalkan data numerik, tetapi juga mampu memberikan informasi yang lebih komprehensif dalam proses monitoring tanaman.

Secara keseluruhan, penelitian ini berhasil mencapai tujuan, yaitu merancang dan mengimplementasikan sistem deteksi anomali tanaman cabai merah berbasis IoT dengan dukungan data sensor dan bukti visual dari ESP32-CAM. Sistem ini dapat dijadikan salah satu solusi untuk mendukung monitoring pertanian secara otomatis, cepat, dan andal.

5.2. Saran

Agar sistem dapat dikembangkan lebih baik pada penelitian selanjutnya, terdapat beberapa saran yang dapat dipertimbangkan. Pertama, precision yang masih rendah perlu ditingkatkan dengan melakukan optimasi model, misalnya mengombinasikan Isolation Forest dengan metode machine lain sehingga hasil deteksi dapat lebih akurat.

Kedua, penelitian ini masih terbatas pada empat sensor utama, yaitu jarak, suhu, kelembapan, dan gerak. Untuk penelitian berikutnya, sistem dapat ditambahkan sensor lain seperti sensor kelembapan tanah, sensor pH tanah, serta sensor intensitas cahaya agar informasi kondisi tanaman cabai merah lebih komprehensif.

Ketiga, penyimpanan data pada penelitian ini masih berbasis file CSV. Sistem dapat ditingkatkan dengan penyimpanan berbasis cloud database seperti Firebase, MySQL, atau ThingsBoard, sehingga memudahkan monitoring secara real-time dari jarak jauh.

Keempat, untuk membuktikan reliabilitas sistem dalam kondisi nyata, perlu dilakukan pengujian langsung di lahan pertanian cabai merah dalam jangka waktu

yang lebih lama. Uji lapangan ini penting untuk melihat kestabilan sensor, ketahanan perangkat, serta performa algoritma dalam menghadapi dinamika lingkungan yang sebenarnya.

Dengan adanya pengembangan lebih lanjut, diharapkan sistem deteksi anomali berbasis IoT ini dapat benar-benar diimplementasikan secara luas di bidang pertanian, khususnya dalam mendukung produktivitas tanaman cabai merah melalui monitoring yang efisien, akurat, dan terpercaya.

DAFTAR PUSTAKA

Aisyah, L., & Febriyanto, F. (2024). Rancang Bangun Sistem Peminjaman Sarana Dan Prasarana Berbasis Website dengan Fitur Push Notification & Bot Telegram. *Jurnal Komputer Dan Aplikasi*, *12*(01).

Alamsyah, M. R., & Kurniawan, H. (2021). Sistem Pakar Menggunakan Metode Certainty Factor untuk Mendiagnosa Hama dan Penyakit pada Tanaman Cabai. *Jurnal Teknologi Informasi*, *XVI*, 38–45.

As'da, A., Suroso, Ciksadan, & Hawayanti, E. (2024). Penerapan Algoritma Yolov3 pada Sistem Cerdas Pendeteksi dan Pengendali Hama Bawang Merah Berbasis IoT. *Building of Informatics, Technology and Science (BITS)*, *6*(2), 930–939. https://doi.org/10.47065/bits.v6i2.5697

Budiani, R. E., Irawan, J. D., & Rudhistiar, D. (2024). SISTEM MONITORING DAN PENYIRAMAN OTOMATIS PADA TANAMAN CABAI BERBASIS INTERNET OF THINGS (IOT). *JATI (Jurnal Mahasiswa Teknik Informatika)*, *8*(2), 1331–1338.

Darusman, L. R. (2023). Pemanfaatan Fitur Bot Telegram Sebagai Pengendali Pada Sistem Penyiraman dan Pemupukan Bibit Kelapa SawiT. *JURNAL AMPERE*, *8*(2), 178–185.

No, V., Hal, O., Santosa, R., Sari, P. A., & Sasongko, A. T. (2023). Sistem Monitoring Suhu dan Kelembaban Berbasis IoT (Internet of Thing) pada Gudang Penyimpanan PT Sakafarma Laboratories. *JTeksis Jurnal Teknologi Dan Sistem Informasi Bisnis*, *5*(4), 391–400.

Putra, R. A., Fatoni, G. A., Rifai, M. H., Ahsani, E. W., & Danendra, R. D. (2024).

Validasi Akurasi Pengukuran Terhadap Benda Menggunakan Sensor Ultrasonik Berbasis NodeMCU 8266. *Jurnal Teknik Mesin, Industri, Elektro Dan Ilmu Komputer*, *2*(3), 188–195.

Riva, L. S., Informatika, J., Komputer, F. I., Pembangunan, U., & Veteran, N. (2023). Deteksi Penyakit Tanaman Cabai Menggunakan Algoritma YOLOv5 Dengan Variasi Pembagian Data. *Jurrnal Informattika: Jurnal Pengembangan IT(JPIT), Vol 8, No. 3, September 2023*, *8*(3), 248–254.

Riyadi, M. (2023). Sistem Cerdas Untuk Monitoring Pengukuran Suhu Dan Kelembapan Tanah Pada Tanaman Cabai Berbasis Internet Of Things (Iot) Menggunakan Aplikasi Telegram. *JTE Jurnal Teknologi Elektro*, *14*(02), 105–109. https://doi.org/10.22441/jte.2023.v14i2.008

Santoso, S. P., & Wijayanto, F. (2022). Rancang Bangun Akses Pintu dengan Sensor Suhu dan Handsanitizer Otomatis Berbasis Arduino. *Jurnal Elektro*, *10*(1).

Sari, I. P., Novita, A., Al-Khowarizmi, Ramadhani, F., & Satria, A. (2024). Pemanfaatan Internet of Things (IoT) pada Bidang Pertanian Menggunakan Arduino UnoR3. *Blend Sains Jurnal Teknik*.

Talli, W. I. S. A., Irawan, J. D., & Ariwibisono, F. X. (2023). RANCANG BANGUN SISTEM MONITORING KUALITAS TANAH UNTUK TANAMAN CABAI BERBASIS IOT (INTERNET OF THINGS). *JATI (Jurnal Mahasiswa Tekknik Informatika)*, *7*(5), 2428–2435.

Wahyudi, Pradana, A. I., & Permatasari, H. (2025). Implementasi Sistem Irigasi Otomatis Berbasis IoT untuk Pertanian Greenhouse Implementation of IoT-Based Automatic Irrigation System for Greenhouse Farming. *Jurnal*

Pendidikan Dan Teknologi Indonesia (JPTI), *5*(2), 435–446.

Wibawa, I. M. S. T., & Karyawati, A. A. I. N. E. (2023). Isolation Forest dengan Exploratory Data Analysis pada Anomaly Detection untuk Data Transaksi. *JNATIA Jurnal Nasional Teknologi Informasi Dan Aplikasinya*, *1*, 803–810.

Zulfikar, A., Rahmani, F. A., & Azizah, N. (2023). DETEKSI ANOMALI MENGGUNAKAN ISOLATION FOREST BELANJA BARANG PERSEDIAAN KONSUMSI PADA SATUAN KERJA KEPOLISIAN REPUBLIK INDONESIA. *Jurnal Manajemen Perbendaharaan*, *4*, 1–15.

LAMPIRAN

1. Lampiran 1 SK Dosen Pembimbing

MAJELIS PENDIDIKAN TINGGI PENELITIAN & PENGEMBANGAN PIMPINAN PUSAT MUHAMMADIYAH
UNIVERSITAS MUHAMMADIYAH SUMATERA UTARA
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UMSU Terakreditasi A Berdasarkan Keputusan Badan Akreditasi Nasional Perguruan Tinggi No. 89/SK/BAN-PT/Akred/PT/III/2019
Pusat Administrasi: Jalan Mukhtar Basri No. 3 Medan 20238 Telp. (061) 6622400 - 66224567 Fax. (061) 6625474 - 6631003
https://fikti.umsu.ac.id fikti@umsu.ac.id umsumedan umsumedan umsumedan umsumedan

PENETAPAN DOSEN PEMBIMBING
PROPOSAL/SKRIPSI MAHASISWA
NOMOR : 486/II.3-AU/UMSU-09/F/2025

Assalamu'alaikum Warahmatullahi Wabarakatuh

Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Muhammadiyah Sumatera Utara, berdasarkan Persetujuan permohonan judul penelitian Proposal / Skripsi dari Ketua / Sekretaris.

Program Studi	**: Teknologi Informasi**
Pada tanggal	**: 12 Maret 2025**

Dengan ini menetapkan Dosen Pembimbing Proposal / Skripsi Mahasiswa.

Nama	**: Surya Darma**
NPM	**: 2109020036**
Semester	**: VIII (Delapan)**
Program studi	**: Teknologi Informasi**
Judul Proposal / Skripsi	**: Implementasi IoT untuk Deteksi dan Pengusiran Hama Pertanian dengan Metode Isolation Forest dan Optimasi Pengendalian Menggunakan Metode Particle Swarm Optimization (PSO)**
Dosen Pembimbing	**: Halim Maulana, S.T.,M.Kom.**

Dengan demikian di izinkan menulis Proposal / Skripsi dengan ketentuan

1. Penulisan berpedoman pada buku panduan penulisan Proposal / Skripsi Fakultas Ilmu Komputer dan Teknologi Informasi UMSU
2. Pelaksanaan Sidang Skripsi harus berjarak 3 bulan setelah dikeluarkannya Surat Penetapan Dosen Pembimbing Skripsi.
3. **Proyek Proposal / Skripsi** dinyatakan " **BATAL** " bila tidak selesai sebelum Masa Kadaluarsa tanggal : **12 Maret 2026**
4. Revisi judul……

Wassalamu'alaikum Warahmatullahi Wabarakatuh.

Ditetapkan di : Medan
Pada Tanggal : 12 Ramadhan 1446 H
12 Maret 2025M

Dekan

Dr. Al-Khowarizmi, M.Kom.
NIDN : 0127099201

Cc. File

2. Lampiran 2 Turnitin

IMPLEMENTASI IOT UNTUK DETEKSI DINI SERANGAN HAMA PADA TANAMAN CABAI MERAH MENGGUNAKAN ALGORITMA ISOLATION FOREST

ORIGINALITY REPORT

18%	17%	6%	11%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Universitas Muhammadiyah Sumatera Utara Student Paper	4%
2	repository.umsu.ac.id Internet Source	1%
3	ejurnal.seminar-id.com Internet Source	1%
4	journal.arteii.or.id Internet Source	1%
5	123dok.com Internet Source	1%
6	text-id.123dok.com Internet Source	<1%
7	jptam.org Internet Source	<1%
8	jmp.kemenkeu.go.id Internet Source	<1%

3. Lampiran 3 ESP32

```
#include <WiFi.h>
 #include <HTTPClient.h>
 #include <DHT.h>

 // Konfigurasi WiFi
 const char* ssid = "Infinixhot50";
 const char* password = "Suryapun825";

 // DHT
#define DHTPIN 4
#define DHTTYPE DHT11  // Ganti ke DHT22 jika perlu
DHT dht(DHTPIN, DHTTYPE);

// PIR
#define PIR_PIN 19

// Ultrasonik
#define TRIG_PIN 5
#define ECHO_PIN 18

// Waktu kirim
unsigned long lastSend = 0;
const long interval = 10000; // 10 detik

void setup() {
  Serial.begin(115200);
  dht.begin();

  pinMode(PIR_PIN, INPUT);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);

  // Hubungkan WiFi
  WiFi.begin(ssid, password);
  Serial.print("Menghubungkan ke WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi terhubung!");
```

```
}

void loop() {
  // Baca DHT
  float suhu = dht.readTemperature();
  float kelembapan = dht.readHumidity();

  // Baca PIR
  int gerakan = digitalRead(PIR_PIN);
  String statusGerak = gerakan == HIGH ? "1" : "0";

  // Baca Ultrasonik
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  long durasi = pulseIn(ECHO_PIN, HIGH);
  float jarak = durasi * 0.034 / 2;

  // Tampilkan ke serial
  Serial.println("=== DATA SENSOR ===");
  Serial.print("Suhu: "); Serial.print(suhu); Serial.println("
°C");
  Serial.print("Kelembapan: "); Serial.print(kelembapan);
Serial.println(" %");
  Serial.print("Gerakan: "); Serial.println(statusGerak == "1"
? "Terdeteksi" : "Tidak ada");
  Serial.print("Jarak: "); Serial.print(jarak);
Serial.println(" cm");

  // Kirim ke server setiap 10 detik
  if (millis() - lastSend >= interval) {
    lastSend = millis();

    if (WiFi.status() == WL_CONNECTED) {
      HTTPClient http;
      http.begin("http://192.168.137.1/iot_db/simpan_data.php"
);
      http.addHeader("Content-Type", "application/x-www-form-
urlencoded");

      String postData = "suhu=" + String(suhu * 10, 1) +
                        "&kelembapan=" + String(kelembapan *
10, 1) +
                        "&jarak=" + String(jarak, 1) +
                        "&gerakan=" + String(gerakan);
```

```

      int httpResponseCode = http.POST(postData);
      Serial.println("Data dikirim: " + postData);
      Serial.println("Response: " + http.getString());
      http.end();
    } else {
      Serial.println("WiFi tidak terhubung saat pengiriman.");
    }
  }

  delay(10000); // loop cepat
}

```

4. Lampiran 4 ESP32 CAM

```
//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> ESP32_CAM_Send_Photo_to_Google_Drive

#include <WiFi.h>
#include <WiFiClientSecure.h>
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include "Base64.h"
#include "esp_camera.h"

//======================================== CAMERA_MODEL_AI_THINKER GPIO (standard mapping)
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27

#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22
//========================================
```

```
30.
31.// LED Flash PIN (GPIO 4)
32.#define FLASH_LED_PIN 4
33.
34.//======================================== Enter your WiFi ssid
   and password.
35.const char* ssid = "Infinixhot50";
36.const char* password = "Suryapun825";
37.//========================================
38.
39.//======================================== Replace with your
   "Deployment ID" and Folder Name.
40.String myDeploymentID =
   "AKfycbyKgKZjWkfzTl7ckhNhu5goP9y5cZt6B9QAqx1YXjZ045haKeJQTU0yW6iC
   ZnGp3xxuIQ";
41.String myMainFolderName = "ESP32-CAM";
42.//========================================
43.
44.//======================================== Variables for
   Timer/Millis.
45.unsigned long previousMillis = 0;
46.const int Interval = 20000; //--> Capture and Send a photo every
   20 seconds.
47.//========================================
48.
49.bool LED_Flash_ON = true;
50.
51.// Initialize WiFiClientSecure.
52.WiFiClientSecure client;
53.
54.//_____________________________________________________________
   ___________________ Test_Con()
55.// This subroutine is to test the connection to
   "script.google.com".
56.void Test_Con() {
57.  const char* host = "script.google.com";
58.  while(1) {
59.    Serial.println("-----------");
60.    Serial.println("Connection Test...");
61.    Serial.println("Connect to " + String(host));
62.
63.    client.setInsecure();
64.
65.    if (client.connect(host, 443)) {
66.      Serial.println("Connection successful.");
67.      Serial.println("-----------");
68.      client.stop();
```

```
69.       break;
70.     } else {
71.       Serial.println("Connected to " + String(host) + "
   failed.");
72.       Serial.println("Wait a moment for reconnecting.");
73.       Serial.println("-----------");
74.       client.stop();
75.     }
76.
77.     delay(1000);
78.   }
79.}
80.//_____________________________________________________________
   _________________
81.
82.//_____________________________________________________________
   _________________ SendCapturedPhotos()
83.// Subroutine for capturing and sending photos to Google Drive.
84.void SendCapturedPhotos() {
85.   const char* host = "script.google.com";
86.   Serial.println();
87.   Serial.println("-----------");
88.   Serial.println("Connect to " + String(host));
89.
90.   client.setInsecure();
91.
92.   //---------------------------------------- The Flash LED blinks
   once to indicate connection start.
93.   digitalWrite(FLASH_LED_PIN, HIGH);
94.   delay(100);
95.   digitalWrite(FLASH_LED_PIN, LOW);
96.   delay(100);
97.   //----------------------------------------
98.
99.   //---------------------------------------- The process of
   connecting, capturing and sending photos to Google Drive.
100.    if (client.connect(host, 443)) {
101.      Serial.println("Connection successful.");
102.
103.      if (LED_Flash_ON == true) {
104.        digitalWrite(FLASH_LED_PIN, HIGH);
105.        delay(100);
106.      }
107.
108.      //.............................. Taking a photo.
109.      Serial.println();
110.      Serial.println("Taking a photo...");
```

```
111.
112.       // warm-up captures
113.       for (int i = 0; i <= 3; i++) {
114.         camera_fb_t * fb = esp_camera_fb_get();
115.         if(!fb) {
116.           Serial.println("Camera capture failed (warm-up).");
117.           esp_camera_fb_return(fb);
118.           return;
119.         }
120.         esp_camera_fb_return(fb);
121.         delay(200);
122.       }
123.
124.       camera_fb_t * fb = esp_camera_fb_get();
125.       if(!fb) {
126.         Serial.println("Camera capture failed (final).");
127.         return;
128.       }
129.
130.       if (LED_Flash_ON == true) digitalWrite(FLASH_LED_PIN,
    LOW);
131.
132.       Serial.println("Taking a photo was successful.");
133.       //.............................
134.
135.       //............................. Sending image to Google
    Drive.
136.       Serial.println();
137.       Serial.println("Sending image to Google Drive.");
138.       Serial.println("Size: " + String(fb->len) + " byte");
139.
140.       String url = "/macros/s/" + myDeploymentID +
    "/exec?folder=" + myMainFolderName;
141.
142.       client.println("POST " + url + " HTTP/1.1");
143.       client.println("Host: " + String(host));
144.       client.println("Transfer-Encoding: chunked");
145.       client.println();
146.
147.       int fbLen = fb->len;
148.       char *input = (char *)fb->buf;
149.       int chunkSize = 3 * 1000; // must be multiple of 3
150.       int chunkBase64Size = base64_enc_len(chunkSize);
151.       // WARNING: chunkBase64Size is compile-time unknown; safe
    stack allocation may fail on low memory boards.
152.       // But we'll try as in original. If compile fails, reduce
    chunkSize.
```

```
153.       char output[chunkBase64Size + 1];
154.
155.       Serial.println();
156.       int chunk = 0;
157.       for (int i = 0; i < fbLen; i += chunkSize) {
158.         int l = base64_encode(output, input, min(fbLen - i,
   chunkSize));
159.         client.print(l, HEX);
160.         client.print("\r\n");
161.         client.print(output);
162.         client.print("\r\n");
163.         delay(100);
164.         input += chunkSize;
165.         Serial.print(".");
166.         chunk++;
167.         if (chunk % 50 == 0) {
168.           Serial.println();
169.         }
170.       }
171.       client.print("0\r\n");
172.       client.print("\r\n");
173.
174.       esp_camera_fb_return(fb);
175.       //...............................
176.
177.       //............................... Waiting for response.
178.       Serial.println("Waiting for response.");
179.       long int StartTime = millis();
180.       while (!client.available()) {
181.         Serial.print(".");
182.         delay(100);
183.         if ((StartTime + 10 * 1000) < millis()) {
184.           Serial.println();
185.           Serial.println("No response.");
186.           break;
187.         }
188.       }
189.       Serial.println();
190.       while (client.available()) {
191.         Serial.print(char(client.read()));
192.       }
193.       //...............................
194.
195.       //............................... Flash LED blinks once as
   an indicator of successfully sending photos to Google Drive.
196.       digitalWrite(FLASH_LED_PIN, HIGH);
197.       delay(500);
```

```
198.       digitalWrite(FLASH_LED_PIN, LOW);
199.       delay(500);
200.       //.............................
201.     }
202.     else {
203.       Serial.println("Connected to " + String(host) + "
   failed.");
204.
205.       //............................. Flash LED blinks twice as
   a failed connection indicator.
206.       digitalWrite(FLASH_LED_PIN, HIGH);
207.       delay(500);
208.       digitalWrite(FLASH_LED_PIN, LOW);
209.       delay(500);
210.       digitalWrite(FLASH_LED_PIN, HIGH);
211.       delay(500);
212.       digitalWrite(FLASH_LED_PIN, LOW);
213.       delay(500);
214.       //.............................
215.     }
216.     //--------------------------------------
217.
218.     Serial.println("-----------");
219.
220.     client.stop();
221.   }
222.   //_____________________________________________________________
   ____________________
223.
224.   //_____________________________________________________________
   ____________________ VOID SETUP()
225.   void setup() {
226.     // Disable brownout detector.
227.     WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
228.
229.     Serial.begin(115200);
230.     Serial.println();
231.     delay(1000);
232.
233.     pinMode(FLASH_LED_PIN, OUTPUT);
234.     digitalWrite(FLASH_LED_PIN, LOW);
235.
236.     // Setting the ESP32 WiFi to station mode.
237.     Serial.println();
238.     Serial.println("Setting the ESP32 WiFi to station mode.");
239.     WiFi.mode(WIFI_STA);
240.
```

```
241.     //------------------------------------ The process of
   connecting ESP32 CAM with WiFi Hotspot / WiFi Router.
242.     Serial.println();
243.     Serial.print("Connecting to : ");
244.     Serial.println(ssid);
245.     WiFi.begin(ssid, password);
246.
247.     unsigned long startAttemptTime = millis();
248.     while (WiFi.status() != WL_CONNECTED && millis() -
   startAttemptTime < 20000) {
249.       Serial.print(".");
250.       digitalWrite(FLASH_LED_PIN, HIGH);
251.       delay(200);
252.       digitalWrite(FLASH_LED_PIN, LOW);
253.       delay(200);
254.     }
255.
256.     Serial.println();
257.     if (WiFi.status() == WL_CONNECTED) {
258.       Serial.println();
259.       Serial.print("Successfully connected to ");
260.       Serial.println(ssid);
261.       Serial.print("ESP32-CAM IP Address: ");
262.       Serial.println(WiFi.localIP());
263.     } else {
264.       int status = WiFi.status();
265.       Serial.println();
266.       Serial.println("❌ Failed to connect to WiFi!");
267.       Serial.print("WiFi.status() = ");
268.       Serial.println(status);
269.       if (status == WL_NO_SSID_AVAIL) Serial.println("⚠ SSID
   not found (cek nama & channel 2.4GHz).");
270.       else if (status == WL_CONNECT_FAILED) Serial.println("⚠
   Connect failed (password salah / WPA3?).");
271.       else Serial.println("⚠ Cek
   SSID/password/frekuensi/channel/power supply.");
272.       // Note: not restarting automatically so you can debug
273.     }
274.     //------------------------------------
275.
276.     //------------------------------------ Set the camera
   ESP32 CAM.
277.     Serial.println();
278.     Serial.println("Set the camera ESP32 CAM...");
279.
280.     camera_config_t config;
281.     config.ledc_channel = LEDC_CHANNEL_0;
```

```
282.     config.ledc_timer = LEDC_TIMER_0;
283.     config.pin_d0 = Y2_GPIO_NUM;
284.     config.pin_d1 = Y3_GPIO_NUM;
285.     config.pin_d2 = Y4_GPIO_NUM;
286.     config.pin_d3 = Y5_GPIO_NUM;
287.     config.pin_d4 = Y6_GPIO_NUM;
288.     config.pin_d5 = Y7_GPIO_NUM;
289.     config.pin_d6 = Y8_GPIO_NUM;
290.     config.pin_d7 = Y9_GPIO_NUM;
291.     config.pin_xclk = XCLK_GPIO_NUM;
292.     config.pin_pclk = PCLK_GPIO_NUM;
293.     config.pin_vsync = VSYNC_GPIO_NUM;
294.     config.pin_href = HREF_GPIO_NUM;
295.     config.pin_sscb_sda = SIOD_GPIO_NUM;
296.     config.pin_sscb_scl = SIOC_GPIO_NUM;
297.     config.pin_pwdn = PWDN_GPIO_NUM;
298.     config.pin_reset = RESET_GPIO_NUM;
299.     config.xclk_freq_hz = 20000000;
300.     config.pixel_format = PIXFORMAT_JPEG;
301.
302.     // init with high specs to pre-allocate larger buffers
303.     if(psramFound()){
304.       config.frame_size = FRAMESIZE_UXGA;
305.       config.jpeg_quality = 20;  //0-63 lower number means
    higher quality
306.       config.fb_count = 2;
307.     } else {
308.       config.frame_size = FRAMESIZE_SVGA;
309.       config.jpeg_quality = 8;  //0-63 lower number means higher
    quality
310.       config.fb_count = 1;
311.     }
312.
313.     // camera init
314.     esp_err_t err = esp_camera_init(&config);
315.     if (err != ESP_OK) {
316.       Serial.printf("Camera init failed with error 0x%x\n",
    err);
317.       Serial.println("⚠ Possible causes: wrong pin mapping,
    incorrect camera module, bad cable/connector, or power issue.");
318.       Serial.println("Please check the camera ribbon cable,
    ensure you selected AI-Thinker board, and try again.");
319.       // Do NOT auto restart so you can read the error.
320.       return;
321.     }
322.
323.     sensor_t * s = esp_camera_sensor_get();
```

```
    if (!s) {
      Serial.println("⚠ Failed to get camera sensor pointer.");
      return;
    }

    // set frame size (you can change)
    s->set_framesize(s, FRAMESIZE_SXGA);

    Serial.println("Setting the camera successfully.");
    Serial.println();

    delay(1000);

    Test_Con();

    Serial.println();
    Serial.println("ESP32-CAM captures and sends photos to the server every 20 seconds.");
    Serial.println();
    delay(2000);
  }
  //_____________________________________________________________________
  //_____________________ VOID LOOP()
  void loop() {
    unsigned long currentMillis = millis();
    if (currentMillis - previousMillis >= Interval) {
      previousMillis = currentMillis;
      SendCapturedPhotos();
    }
  }
  //_____________________________________________________________________
  //<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

5. Lampiran 5 Isolation Forest

```
# ============================================
# DETEKSI ANOMALI TANAMAN CABAI MERAH
# MENGGUNAKAN ISOLATION FOREST
# ============================================
```

```
# 1. IMPORT LIBRARY
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import IsolationForest
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report,
confusion matrix
# 2. LOAD DATA
data = pd.read csv("/content/data cabai.csv", delimiter=";")
print("Preview Data:")
print(data.head())
print("\nInfo Data:")
print(data.info())
# 3. PREPROCESSING
# Ambil fitur numerik
features = ["jarak", "suhu", "kelembapan", "gerak"]
X = data[features]
y_true = data["anomali"]  # label ground-truth
# Isi missing value (kalau ada)
X = X.fillna(X.median())
# Normalisasi
scaler = StandardScaler()
X scaled = scaler.fit transform(X)
# 4. TRAINING ISOLATION FOREST
iso = IsolationForest(
    contamination=0.1,   # asumsi 10% data anomali
    random_state=42
)
y_pred = iso.fit_predict(X_scaled)

# Ubah hasil ke format 0=normal, 1=anomali
y pred = np.where(y pred == -1, 1, 0)

# Simpan hasil ke dataframe
data["predicted_anomaly"] = y_pred

print("\nDistribusi hasil prediksi:")
print(pd.Series(y_pred).value_counts())

# 4b. DAPATKAN SKOR ANOMALI
scores = iso.decision function(X scaled)
data["anomaly_score"] = scores

# Distribusi skor anomali
plt.figure(figsize=(10,6))
```

```
sns.histplot(scores, bins=50, kde=True, color="blue")
plt.title("Distribusi Skor Anomali (Isolation Forest)")
plt.xlabel("Skor Anomali")
plt.ylabel("Frekuensi")
plt.axvline(x=np.median(scores), color="red", linestyle="--",
label="Median Skor")
plt.legend()
plt.show()
# Scatter plot skor vs index data
plt.figure(figsize=(12,6))
plt.plot(data.index, scores, label="Skor Anomali",
color="blue")
plt.scatter(
    data.index[data["predicted_anomaly"]==1],
    scores[data["predicted_anomaly"]==1],
    color="red", label="Anomali"
)
plt.title("Skor Anomali per Data")
plt.xlabel("Index Data")
plt.ylabel("Skor Anomali")
plt.legend()
plt.show()
# 5. VISUALISASI HASIL (contoh sensor Jarak)
# Visualisasi hasil prediksi untuk semua sensor
sensors = ["jarak", "suhu", "kelembapan", "gerak"]
plt.figure(figsize=(16, 12))
for i, sensor in enumerate(sensors, 1):
    plt.subplot(len(sensors), 1, i)
    plt.plot(data.index, data[sensor], label=f"{sensor}",
color="blue")
    plt.scatter(
        data.index[data["predicted anomaly"] == 1],
        data[sensor][data["predicted_anomaly"] == 1],
        color="red", label="Anomali"
    )
    plt.title(f"Deteksi Anomali pada Sensor
{sensor.capitalize()}")
    plt.xlabel("Index Data")
    plt.ylabel(sensor.capitalize())
    plt.legend()
plt.tight layout()
plt.show()
# 6. EVALUASI (Precision, Recall, F1)
print("\nConfusion Matrix (angka):")
print(confusion matrix(y true, y pred))
print("\nClassification Report:")
```

```
print(classification_report(y_true, y_pred,
target_names=["Normal", "Anomali"]))
# 7. VISUALISASI PERBANDINGAN (Ground Truth vs Prediksi)
plt.figure(figsize=(14,6))
# Grafik Label Asli
plt.subplot(1,2,1)
plt.plot(data.index, data["jarak"], label="Jarak Sensor",
color="blue")
plt.scatter(
    data.index[data["anomali"]==1],
    data["jarak"][data["anomali"]==1],
    color="red", label="Anomali (Ground Truth)"
)
plt.title("Label Asli (Ground Truth)")
plt.xlabel("Index Data")
plt.ylabel("Jarak (cm)")
plt.legend()

# Grafik Prediksi Model
plt.subplot(1,2,2)
plt.plot(data.index, data["jarak"], label="Jarak Sensor",
color="blue")
plt.scatter(
    data.index[data["predicted_anomaly"]==1],
    data["jarak"][data["predicted anomaly"]==1],
    color="orange", label="Anomali (Prediksi Model)"
)
plt.title("Prediksi Isolation Forest")
plt.xlabel("Index Data")
plt.ylabel("Jarak (cm)")
plt.legend()

plt.tight_layout()
plt.show()
# 8. CONFUSION MATRIX HEATMAP
cm = confusion matrix(y true, y pred)

plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
            xticklabels=["Normal", "Anomali"],
            yticklabels=["Normal", "Anomali"])
plt.title("Confusion Matrix Heatmap")
plt.xlabel("Prediksi")
plt.ylabel("Label Asli")
plt.show()
# 9. VISUALISASI DETEKSI ANOMALI (SEMUA SENSOR DALAM 1 FIGURE)
sensors = ["jarak", "suhu", "kelembapan"]
```

```
plt.figure(figsize=(16,10))

for i, sensor in enumerate(sensors, 1):
    plt.subplot(len(sensors), 1, i)
    plt.plot(data.index, data[sensor], label=f"Sensor
{sensor}", color="blue")
    plt.scatter(
        data.index[data["predicted anomaly"]==1],
        data[sensor][data["predicted_anomaly"]==1],
        color="red", label="Anomali"
    )
    plt.title(f"Deteksi Anomali pada Sensor
{sensor.capitalize()} Tanaman Cabai Merah")
    plt.xlabel("Index Data")
    plt.ylabel(sensor.capitalize())
    plt.legend()

plt.tight_layout()
plt.show()
# ========================================
# 10. SIMPAN HASIL PREDIKSI KE CSV
# ========================================
output_file = "/content/hasil_prediksi.csv"
data.to csv(output file, index=False)
print(f"Hasil prediksi berhasil disimpan ke: {output_file}")
# 11. TABEL HASIL DETEKSI ANOMALI
hasil_tabel = data[["jarak", "suhu", "kelembapan", "gerak",
                    "anomali", "predicted_anomaly",
"anomaly_score"]]
print("\nTabel Hasil Deteksi Anomali (5 data teratas):")
print(hasil tabel.head(10))  # tampilkan 10 baris pertama
# Jika mau lihat tabel lengkap di Google Colab
import pandas as pd
from IPython.display import display
display(hasil tabel)
```

6. Lampiran 6 YOLOv5

```
1. import os
2. import io
3. import re
4. import time
5. import json
6. import pickle
```

```
import torch
from datetime import datetime
from googleapiclient.discovery import build
from googleapiclient.errors import HttpError
from googleapiclient.http import MediaIoBaseDownload
from google_auth_oauthlib.flow import InstalledAppFlow
from google.auth.transport.requests import Request
import requests

# ========================
# KONFIGURASI
# ========================
SCOPES = ['https://www.googleapis.com/auth/drive.readonly']
CREDENTIALS_FILE = 'credentials.json'    # OAuth Client (bukan service account)
TOKEN_PICKLE = 'token.pickle'

# >>>> GANTI DENGAN FOLDER YANG MAU DIBACA (TETAP) <<<<
FOLDER_ID = '1aiVS6wjlnqSCFCtRU4fOyr0cilD3wPvm'  # folder yang akan kamu baca isinya

# Jika folder ini berisi subfolder (misal 20250818, 20250819, dst), set True
# Jika folder ini langsung berisi gambar, set False
RECURSIVE = True

# YOLOv5
model_path = r"C:\Users\User\OneDrive\Desktop\ProjectSkripsi\yolov5\runs\train\exp11\weights\best.pt"
yolov5_repo = r"C:\Users\User\OneDrive\Desktop\ProjectSkripsi\yolov5"

# Endpoint PHP/MySQL
php_url = "http://192.168.137.1/iot_db/insert_data.php"

# Lama jalan (detik) - 24 jam
run_duration = 24 * 60 * 60
poll_interval_sec = 10  # jangan terlalu kecil agar tidak kena rate limit

# ========================
# AUTH & SERVICE
# ========================
def get_drive_service():
    """
    OAuth dengan token refresh otomatis.
```

```
    - Jika token.pickle ada → pakai & refresh saat expired.
    - Jika tidak ada → buka login (run_local_server).
    """
    creds = None
    if os.path.exists(TOKEN_PICKLE):
        with open(TOKEN_PICKLE, 'rb') as f:
            creds = pickle.load(f)
    if not creds or not creds.valid:
        if creds and creds.expired and creds.refresh_token:
            try:
                creds.refresh(Request())
            except Exception:
                creds = None
        if not creds:
            # Pastikan credentials.json adalah OAuth client, bukan service account
            flow = InstalledAppFlow.from_client_secrets_file(CREDENTIALS_FILE, SCOPES)
            # Akan buka browser sekali untuk login
            creds = flow.run_local_server(port=0)
        with open(TOKEN_PICKLE, 'wb') as f:
            pickle.dump(creds, f)
    return build('drive', 'v3', credentials=creds)

service = get_drive_service()

# ========================
# UTIL DRIVE
# ========================
def list_children(service, folder_id):
    """List semua child (file & folder) langsung di bawah folder_id (non-rekursif)."""
    files = []
    page_token = None
    while True:
        resp = service.files().list(
            q=f"'{folder_id}' in parents and trashed=false",
            fields="nextPageToken, files(id, name, mimeType, createdTime)",
            pageToken=page_token,
            pageSize=1000
        ).execute()
        files.extend(resp.get('files', []))
        page_token = resp.get('nextPageToken', None)
        if not page_token:
            break
```

```
    return files

def list_images_non_recursive(service, folder_id):
    """Ambil file image/* langsung di folder_id (tanpa menelusuri subfolder)."""
    results = []
    for f in list_children(service, folder_id):
        if f.get('mimeType', '').startswith('image/'):
            results.append(f)
    return results

def list_images_recursive(service, folder_id):
        """
        Telusuri seluruh subfolder (BFS) dan kumpulkan semua file image/*.
        Cocok bila struktur: ESP32-CAM > 20250818 > gambar.jpg, dst.
        """
        images = []
        queue = [folder_id]
        visited = set()
        while queue:
            fid = queue.pop(0)
            if fid in visited:
                continue
            visited.add(fid)
            for f in list_children(service, fid):
                mt = f.get('mimeType', '')
                if mt == 'application/vnd.google-apps.folder':
                    queue.append(f['id'])
                elif mt.startswith('image/'):
                    images.append(f)
        return images

    def safe_drive_call(func, *args, **kwargs):
        """
        Bungkus pemanggilan Drive API dengan retry & re-auth sederhana.
        """
        max_attempts = 3
        delay = 5
        global service
        for attempt in range(1, max_attempts + 1):
            try:
                return func(*args, **kwargs)
            except HttpError as e:
```

```
131.                 code = getattr(e.resp, 'status', None)
132.                 print(f"⚠ HttpError (try
    {attempt}/{max_attempts}) status={code}: {e}")
133.                 # 401 → token invalid/expired; 403/429/5xx →
    rate limit/transient
134.                 if code in (401,):
135.                     service = get_drive_service()
136.                 time.sleep(delay * attempt)
137.             except Exception as e:
138.                 print(f"⚠ Error (try
    {attempt}/{max_attempts}): {e}")
139.                 time.sleep(delay * attempt)
140.         # Jika tetap gagal, lempar lagi
141.         return func(*args, **kwargs)
142.
143.     def download_file(service, file_id, local_path):
144.         """Unduh file via Drive API (bukan link publik)."""
145.         request = service.files().get_media(fileId=file_id)
146.         with open(local_path, 'wb') as fh:
147.             downloader = MediaIoBaseDownload(fh, request)
148.             done = False
149.             while not done:
150.                 status, done = downloader.next_chunk()
151.
152.     # ========================
153.     # YOLO SETUP
154.     # ========================
155.     model = torch.hub.load(
156.         yolov5_repo,
157.         'custom',
158.         path=model_path,
159.         source='local'
160.     )
161.     model.conf = 0.6
162.     model.iou = 0.45
163.     allowed_classes = list(model.names.values())
164.
165.     # ========================
166.     # MAIN LOOP
167.     # ========================
168.     start_time = time.time()
169.     processed_files = set()  # simpan ID file yang sudah
    diproses
170.
171.     print("Memantau folder Google Drive selama 24 jam...")
172.
173.     while True:
```

```
174.         if time.time() - start_time > run_duration:
175.             print("Waktu 24 jam habis.")
176.             break
177.
178.         try:
179.             # Ambil daftar gambar
180.             if RECURSIVE:
181.                 files =
    safe_drive_call(list_images_recursive, service, FOLDER_ID)
182.             else:
183.                 files =
    safe_drive_call(list_images_non_recursive, service, FOLDER_ID)
184.
185.             if not files:
186.                 print("Belum ada file gambar yang
    terdeteksi...")
187.             else:
188.                 # Urutkan berdasarkan createdTime agar rapi
189.                 files.sort(key=lambda x:
    x.get('createdTime', ''))
190.
191.                 for f in files:
192.                     fid = f['id']
193.                     fname = f['name']
194.
195.                     if fid in processed_files:
196.                         continue
197.
198.                     # Unduh temporer
199.                     local_path = f"temp_{fid}_{fname}"
200.                     try:
201.                         safe_drive_call(download_file,
    service, fid, local_path)
202.                     except Exception as e:
203.                         print(f"⚠ Gagal mengunduh {fname}:
    {e}")
204.                         continue
205.
206.                     # Deteksi dengan YOLO
207.                     try:
208.                         results = model(local_path)
209.                         detections =
    results.pandas().xyxy[0]
210.                         detections =
    detections[detections['name'].isin(allowed_classes)]
211.                         status = "Anomali" if
    len(detections) > 0 else "Tidak Anomali"
```

```
212.                         print(f"{fname} → {status}")
213.                     except Exception as e:
214.                         print(f"⚠ Gagal deteksi {fname}:
    {e}")
215.                         status = "Error"
216.
217.                     # Kirim ke PHP/MySQL
218.                     data = {
219.                         'nama_file': fname,
220.                         'waktu':
    datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
221.                         'keterangan': status
222.                     }
223.                     try:
224.                         r = requests.post(php_url,
    data=data, timeout=5)
225.                         print("Respon server:",
    r.text.strip())
226.                     except Exception as e:
227.                         print("⚠ Gagal kirim ke server:",
    e)
228.
229.                     processed_files.add(fid)
230.                     try:
231.                         os.remove(local_path)
232.                     except Exception:
233.                         pass
234.
235.             time.sleep(poll_interval_sec)
236.
237.         except KeyboardInterrupt:
238.             print("Dihentikan oleh pengguna.")
239.             break
240.         except Exception as e:
241.             # Guard terakhir agar loop tidak mati
242.             print("⚠ Error di loop utama:", e)
243.             # Rebuild service jika perlu
244.             try:
245.                 service = get_drive_service()
246.             except Exception:
247.                 pass
248.             time.sleep(10)
249.
```