Jurnal Publikasi Ilmu Komputer dan Multimedia

E-ISSN: 2808-8999
P-ISSN: 2808-9375

Sistem Absensi Otomatis Berbasis Face Recognition Menggunakan ESP32-CAM dan Web Dashboard

Muhammad As'ad Fadhil Yahya [1], Agusma Wajiansyah [2], Dwi Titi Maesaroh [3,*]

[123]Politeknik Negeri Samarinda; Jl. Dr. Cipto Mangunkusumo, Samarinda 75131
Email : fadilboy295@gmail.com[1], agusma.wajiansyah@gmail.com[2], dwititi.maesaroh@polnes.ac.id[3]
***Penulis Korespondensi**: Dwi Titi Maesaroh

Abstract: This study presents the development of an automatic attendance system based on face recognition using ESP32-CAM and a web-based dashboard. The object of this research is an integrated attendance system that combines an IoT-based camera module, computer vision processing, database storage, and a web interface. Conventional attendance systems are often inefficient and prone to manipulation, such as proxy attendance. Therefore, this research aims to design and implement an attendance system that can detect faces, recognize registered users, and record attendance data automatically. The proposed method uses Haar Cascade for face detection and Local Binary Pattern Histogram (LBPH) for face recognition, implemented on a Python server using OpenCV and Flask. ESP32-CAM is used to capture video frames and send them to the server through a Wi-Fi connection. The recognized attendance data are stored in an SQLite database and displayed through a web dashboard. The dataset was collected from registered users with several variations in facial position, expression, distance, and lighting conditions. The experimental results show that the system successfully detected faces in all testing scenarios and achieved the best recognition performance at a distance of 30-50 cm with lighting conditions of 150-300 lux. Computational performance testing showed an average response time of 8.98 ms, an average streaming rate of 22.59 FPS, an average network latency of 7 ms, an average CPU usage of 5.44%, and an average RAM usage of 143.69 MB on the server. In conclusion, the proposed system can be used as a low-cost attendance solution, although its recognition performance is still affected by distance and lighting conditions.

Keywords: *face recognition; ESP32-CAM; LBPH; attendance system; Haar Cascade; OpenCV; web dashboard; SQLite*

Abstrak: Penelitian ini membahas pengembangan sistem absensi otomatis berbasis pengenalan wajah menggunakan ESP32-CAM dan dashboard berbasis web. Objek penelitian ini adalah sistem absensi terintegrasi yang menggabungkan modul kamera berbasis IoT, pemrosesan citra, penyimpanan database, dan antarmuka web. Sistem absensi konvensional masih memiliki kelemahan, seperti kurang efisien dan rentan terhadap manipulasi, misalnya praktik titip absen. Oleh karena itu, penelitian ini bertujuan untuk merancang dan mengimplementasikan sistem absensi yang mampu mendeteksi wajah, mengenali pengguna terdaftar, serta mencatat data kehadiran secara otomatis. Metode yang digunakan adalah Haar Cascade untuk deteksi wajah dan Local Binary Pattern Histogram (LBPH) untuk pengenalan wajah, yang diimplementasikan pada server Python menggunakan OpenCV dan Flask. ESP32-CAM digunakan untuk menangkap frame video dan mengirimkannya ke server melalui koneksi Wi-Fi. Data absensi hasil pengenalan disimpan ke dalam database SQLite dan ditampilkan melalui dashboard web. Dataset diperoleh dari pengguna terdaftar dengan beberapa variasi posisi wajah, ekspresi, jarak, dan kondisi pencahayaan. Hasil pengujian menunjukkan bahwa sistem berhasil mendeteksi wajah pada seluruh skenario pengujian dan memperoleh performa pengenalan terbaik pada jarak 30-50 cm dengan kondisi pencahayaan 150-300 lux. Pengujian performa komputasi menunjukkan waktu respons rata-rata 8,98 ms, FPS streaming rata-rata 22,59 FPS, latensi jaringan rata-rata 7 ms, penggunaan CPU rata-rata 5,44%, dan penggunaan RAM rata-rata 143,69 MB pada server. Kesimpulannya, sistem yang diusulkan dapat digunakan sebagai solusi absensi berbiaya rendah, meskipun performa pengenalan masih dipengaruhi oleh jarak dan kondisi pencahayaan.

Diterima: 18 Mei 2026
Direvisi: 22 Mei 2026
Diterima: 25 Mei 2026
Diterbitkan: 31 Mei 2026
Versi sekarang: Mei 2026

Kata kunci: *pengenalan wajah; ESP32-CAM; LBPH; sistem absensi; Haar Cascade; OpenCV; dashboard web; SQLite*

1. Pendahuluan

Perkembangan teknologi computer vision dan Internet of Things (IoT) telah mendorong penerapan sistem otomatis dalam berbagai bidang, termasuk sistem pencatatan kehadiran. Salah satu penerapannya adalah sistem absensi berbasis pengenalan wajah yang memanfaatkan citra wajah sebagai identitas biometrik pengguna. Teknologi ini dinilai dapat mengurangi kelemahan metode absensi konvensional seperti tanda tangan manual atau kartu identitas, yang masih berpotensi mengalami kesalahan pencatatan maupun praktik titip absen [1], [2].

Sistem absensi berbasis pengenalan wajah umumnya bekerja melalui beberapa tahapan, yaitu akuisisi citra wajah, deteksi wajah, ekstraksi ciri, pengenalan wajah, dan pencatatan data kehadiran [1], [2]. Dalam penerapannya, keberhasilan sistem sangat dipengaruhi oleh kualitas citra yang diperoleh, metode pengolahan citra yang digunakan, serta kondisi lingkungan seperti pencahayaan dan jarak pengambilan wajah [2], [4], [11]. Oleh karena itu, diperlukan perancangan sistem yang tidak hanya mampu mengenali wajah, tetapi juga mampu mencatat data absensi secara terstruktur dan mudah dipantau.

Sementara itu, Zhang et al. [6] mengembangkan sistem pengenalan wajah berbasis deep learning dengan akurasi tinggi, sedangkan kajian terbaru oleh Wang dan Deng [18] menunjukkan bahwa pendekatan deep face recognition memiliki kemampuan representasi fitur yang kuat untuk variasi wajah yang kompleks. Namun, pendekatan tersebut umumnya membutuhkan dataset dan sumber daya komputasi yang lebih besar untuk proses pelatihan maupun pengenalan wajah.

ESP32-CAM dipilih karena telah dilengkapi modul kamera dan konektivitas Wi-Fi yang mendukung pengembangan sistem berbasis IoT serta banyak digunakan pada implementasi kamera berbasis mikrokontroler dan sistem absensi berbasis pengenalan wajah [8], [20], [24].

Berdasarkan kajian tersebut, masih terdapat kebutuhan untuk mengembangkan sistem absensi berbasis pengenalan wajah yang mengintegrasikan perangkat akuisisi citra, proses deteksi dan pengenalan wajah, penyimpanan database, serta dashboard web dalam satu sistem. Penelitian ini mengusulkan sistem absensi berbasis ESP32-CAM dan server Python yang menggunakan OpenCV sebagai pustaka pengolahan citra [9]. Metode Haar Cascade digunakan untuk mendeteksi lokasi wajah pada citra, sedangkan metode LBPH digunakan untuk mengenali wajah berdasarkan model hasil training dari dataset pengguna. Data absensi yang dihasilkan kemudian disimpan ke dalam database SQLite dan ditampilkan melalui web dashboard berbasis Flask.

Tujuan penelitian ini adalah: (1) merancang dan mengimplementasikan sistem absensi berbasis pengenalan wajah menggunakan ESP32-CAM dan server Python; (2) menjelaskan proses pengambilan dataset, training model LBPH, deteksi wajah, dan pengenalan wajah pada sistem; (3) menguji keberhasilan deteksi dan pengenalan wajah pada variasi intensitas cahaya dan jarak pengambilan wajah; serta (4) menampilkan data absensi melalui database dan dashboard web. Kontribusi penelitian ini terletak pada integrasi ESP32-CAM sebagai perangkat akuisisi citra, server Python sebagai pusat pemrosesan, SQLite sebagai penyimpanan data, dan dashboard web sebagai antarmuka monitoring absensi.

Kebaruan penelitian ini terletak pada integrasi ESP32-CAM sebagai perangkat akuisisi citra berbasis Wi-Fi dengan server Python sebagai pusat pemrosesan, metode Haar Cascade untuk deteksi wajah, metode LBPH untuk pengenalan wajah, database SQLite sebagai media penyimpanan, dan web dashboard sebagai antarmuka monitoring dalam satu alur sistem. Berbeda dengan penelitian sebelumnya yang umumnya berfokus pada deteksi wajah, penggunaan webcam lokal, atau metode pengenalan tertentu, penelitian ini menambahkan evaluasi kondisi penggunaan berdasarkan kombinasi intensitas cahaya dan jarak wajah terhadap kamera. Dengan demikian, kontribusi penelitian ini tidak hanya terletak pada implementasi sistem absensi, tetapi juga pada pengujian kondisi operasional sistem berbasis ESP32-CAM secara terukur.

Artikel ini terorganisasi sebagai berikut: Bagian 2 membahas tinjauan literatur terkait; Bagian 3 menjelaskan metodologi penelitian; Bagian 4 menyajikan hasil dan pembahasan;

Bagian 5 menyajikan perbandingan, novelty, dan keterbatasan penelitian; dan Bagian 6 menyajikan kesimpulan serta saran pengembangan lebih lanjut.

2. Tinjauan Literatur

2.1. Sistem Absensi Berbasis Face Recognition

Sistem absensi berbasis pengenalan wajah merupakan salah satu penerapan teknologi biometrik yang digunakan untuk mengidentifikasi individu berdasarkan karakteristik wajah. Berbeda dengan metode absensi konvensional seperti tanda tangan manual atau kartu identitas, sistem berbasis wajah memanfaatkan citra wajah sebagai data identifikasi sehingga dapat mengurangi potensi manipulasi seperti praktik titip absen. Secara umum, sistem pengenalan wajah terdiri dari beberapa tahap, yaitu akuisisi citra, deteksi wajah, ekstraksi ciri, pencocokan data, dan pengambilan keputusan identitas [2].

Dalam konteks sistem absensi, kamera berperan sebagai perangkat akuisisi citra untuk menangkap wajah pengguna [1], [2]. Citra wajah yang diperoleh kemudian diproses untuk mendeteksi keberadaan wajah dan mengenali identitas pengguna berdasarkan data yang telah dilatih sebelumnya [2], [3], [4]. Jika identitas berhasil dikenali, sistem akan mencatat data kehadiran seperti ID pengguna, nama, dan waktu absensi ke dalam basis data.

Meskipun sistem absensi berbasis pengenalan wajah memiliki keunggulan dalam hal otomatisasi dan kemudahan penggunaan, performanya masih dipengaruhi oleh beberapa faktor. Kualitas citra, kondisi pencahayaan, jarak pengambilan wajah, posisi wajah, serta keragaman dataset training dapat mempengaruhi tingkat keberhasilan deteksi dan pengenalan wajah [2], [4], [11]. Oleh karena itu, sistem perlu dirancang dengan memperhatikan proses akuisisi citra dan kondisi pengujian agar hasil yang diperoleh lebih valid [1], [2], [11].

2.2. Metode Haar Cascade untuk Deteksi Wajah

Haar Cascade merupakan metode deteksi objek yang diperkenalkan oleh Viola dan Jones [3]. Metode ini bekerja dengan memanfaatkan fitur Haar-like untuk mendeteksi pola tertentu pada citra, seperti perbedaan intensitas antara area terang dan gelap. Dalam deteksi wajah, pola tersebut digunakan untuk mengenali karakteristik wajah seperti area mata, hidung, dan mulut. Proses klasifikasi dilakukan menggunakan cascade classifier, yaitu rangkaian classifier bertingkat yang menyaring area citra secara bertahap hingga ditemukan area yang memiliki karakteristik wajah.

Input dari metode Haar Cascade adalah citra, umumnya dalam format grayscale, karena proses deteksi didasarkan pada intensitas piksel [3]. Pada sistem absensi ini, frame video dari ESP32-CAM diterima oleh server Python, kemudian dikonversi menjadi citra grayscale sebelum diproses oleh Haar Cascade. Output dari metode Haar Cascade bukan berupa identitas pengguna, melainkan lokasi wajah pada citra dalam bentuk koordinat bounding box, yaitu nilai x, y, w, dan h [3]. Koordinat tersebut menunjukkan posisi awal wajah serta lebar dan tinggi area wajah yang terdeteksi.

Keunggulan Haar Cascade adalah proses deteksinya relatif cepat dan dapat diterapkan pada sistem berbasis OpenCV [9]. Hal ini membuat metode ini banyak digunakan pada aplikasi deteksi wajah sederhana. Namun, metode ini memiliki keterbatasan, terutama pada kondisi pencahayaan rendah, wajah tidak frontal, atau latar belakang yang kompleks. Oleh karena itu, Haar Cascade pada penelitian ini digunakan sebagai tahap awal untuk menemukan area wajah, sedangkan proses pengenalan identitas dilakukan oleh metode lain, yaitu LBPH.

2.3. Metode LBPH untuk Pengenalan Wajah

Local Binary Pattern Histogram (LBPH) merupakan metode pengenalan wajah yang dikembangkan dari konsep Local Binary Pattern (LBP) [4]. Metode ini bekerja dengan menganalisis tekstur lokal pada citra wajah. Setiap piksel dibandingkan dengan piksel di sekitarnya untuk membentuk pola biner. Pola-pola biner tersebut kemudian direpresentasikan dalam bentuk histogram yang digunakan sebagai ciri wajah.

Pada tahap training, input LBPH berupa kumpulan citra wajah grayscale dan label numerik dari masing-masing pengguna [4]. Citra wajah tersebut diperoleh dari dataset yang telah dikumpulkan sebelumnya. Setiap pengguna memiliki folder dataset tersendiri, misalnya folder 1, 2, dan seterusnya, yang menunjukkan label identitas pengguna. Dalam proses training, citra wajah disimpan sebagai data wajah, sedangkan nama folder atau label numerik

disimpan sebagai label pengguna. Hasil dari proses training adalah model pengenalan wajah yang disimpan dalam file trainer.yml [4].

Pada tahap pengenalan, input LBPH adalah citra wajah hasil cropping dari proses deteksi Haar Cascade. Citra wajah tersebut kemudian dibandingkan dengan model hasil training. Output dari LBPH bukan langsung berupa nama pengguna, melainkan berupa label prediksi dan nilai confidence [4]. Label prediksi menunjukkan label pengguna yang paling mendekati data training, sedangkan nilai confidence menunjukkan jarak atau tingkat perbedaan antara citra uji dan data pada model. Semakin kecil nilai confidence, maka tingkat kemiripan wajah semakin tinggi [4].

Dengan demikian, identitas seperti ID dan nama pengguna bukan merupakan output langsung dari LBPH. ID dan nama diperoleh melalui proses pemetaan label prediksi ke data pengguna yang telah didefinisikan dalam sistem atau database [4]. Penjelasan ini penting agar proses pengenalan wajah tidak disalahartikan sebagai proses yang langsung menghasilkan nama atau ID pengguna tanpa tahap pemetaan.

LBPH dipilih karena memiliki proses yang sederhana dan dapat diterapkan pada sistem dengan kebutuhan komputasi yang tidak terlalu tinggi [4], [17]. Namun, metode ini masih memiliki keterbatasan pada kondisi citra yang kurang baik, seperti wajah terlalu jauh dari kamera, pencahayaan tidak merata, atau sudut wajah yang terlalu berbeda dari dataset training [2], [11].

2.4. ESP32-CAM sebagai Perangkat Akuisisi Citra

ESP32-CAM merupakan modul mikrokontroler yang dilengkapi dengan kamera dan konektivitas Wi-Fi. Perangkat ini banyak digunakan pada sistem berbasis Internet of Things (IoT) karena ukurannya kecil, biaya relatif rendah, dan mampu mengirim data melalui jaringan nirkabel [8]. Dalam penelitian ini, ESP32-CAM digunakan sebagai perangkat akuisisi citra wajah, bukan sebagai perangkat utama untuk menjalankan proses pengenalan wajah.

Pemisahan fungsi ini dilakukan karena ESP32-CAM memiliki keterbatasan sumber daya komputasi untuk menjalankan proses pengolahan citra yang lebih kompleks, sehingga pemrosesan pada server lokal menjadi pendekatan yang lebih sesuai untuk sistem ini [8], [19], [20].

Dengan pendekatan tersebut, ESP32-CAM berfungsi sebagai node kamera berbasis jaringan, sedangkan server Python berfungsi sebagai pusat pemrosesan [8]. Hal ini membedakan sistem ini dari sistem berbasis webcam USB yang harus terhubung langsung ke komputer pemrosesan [7]. Namun, penggunaan ESP32-CAM tetap memiliki keterbatasan, terutama pada kualitas citra dan resolusi kamera, sehingga jarak pengambilan wajah dan intensitas cahaya tetap menjadi faktor penting dalam pengujian sistem [8], [11].

Gambar 1. Mikrokontroler ESP32-CAM

2.5. Integrasi Sistem dengan Database dan Web Dashboard

Integrasi database dan web dashboard diperlukan agar sistem absensi tidak hanya mampu mengenali wajah, tetapi juga mampu mencatat, menyimpan, dan menampilkan data kehadiran secara terstruktur [10], [13]. Database digunakan untuk menyimpan data absensi, seperti ID pengguna, nama, dan waktu kehadiran. Pada penelitian ini, database yang

digunakan adalah SQLite, yang berjalan pada laptop/PC server yang sama dengan aplikasi Python [13].

Server Python dikembangkan menggunakan framework Flask. Flask berfungsi sebagai penghubung antara proses pengolahan citra, database, dan antarmuka web [10]. Dengan demikian, server Python, database SQLite, dan web dashboard bukan merupakan perangkat fisik yang berbeda, melainkan bagian dari satu sistem perangkat lunak yang berjalan pada laptop/PC server.

Web dashboard digunakan sebagai antarmuka pengguna untuk menampilkan informasi absensi [10]. Dashboard menampilkan hasil live streaming kamera, data kehadiran, fitur filter berdasarkan tanggal, dan fitur ekspor data. Dengan adanya dashboard, proses monitoring absensi menjadi lebih mudah karena data yang telah disimpan dalam SQLite dapat ditampilkan kembali melalui browser [10], [13].

3. Metode

Penelitian ini menggunakan metode rekayasa eksperimen (engineering design) dengan pendekatan perancangan, implementasi, dan pengujian sistem berbasis computer vision untuk absensi otomatis [1], [12]. Sistem yang dikembangkan memanfaatkan ESP32-CAM sebagai perangkat akuisisi citra wajah, sedangkan proses deteksi wajah, pengenalan wajah, penyimpanan data, dan tampilan dashboard dilakukan pada laptop/PC server menggunakan Python, OpenCV, Flask, dan SQLite [8], [9], [10], [13].

Metode penelitian dilakukan melalui beberapa tahapan utama, yaitu perancangan perangkat keras dan perangkat lunak, pengambilan dataset wajah, training model pengenalan wajah menggunakan LBPH, proses deteksi wajah menggunakan Haar Cascade, pencatatan data absensi ke database, serta pengujian sistem berdasarkan variasi intensitas cahaya dan jarak wajah terhadap kamera [2], [3], [4], [11], [12].

3.1. Alat dan Bahan Penelitian

Perancangan perangkat keras (hardware) difokuskan pada integrasi yang bersifat modular agar mudah dipasang dan digunakan. Komponen utama yang digunakan dalam penelitian ini adalah:

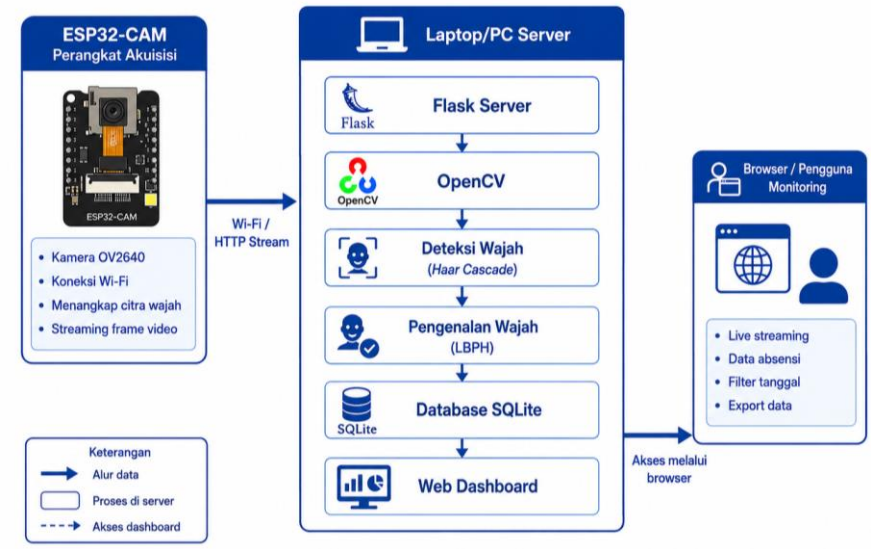
- ESP32-CAM: Sebagai perangkat utama untuk menangkap citra wajah secara langsung dan mengirimkan frame video ke server melalui jaringan Wi-Fi [8].
- Laptop/PC Server: Sebagai pusat pemrosesan data yang menjalankan aplikasi Python, OpenCV, Flask, serta database SQLite[9], [10], [13], [16].
- Lux Meter: Digunakan sebagai alat ukur intensitas cahaya untuk mengetahui pengaruh pencahayaan terhadap keberhasilan deteksi dan pengenalan wajah [11].
- Laser Distance Meter: Digunakan untuk mengukur jarak antara wajah pengguna dan kamera ESP32-CAM pada saat pengujian.

Perangkat lunak (software) yang digunakan dalam penelitian ini adalah:

- Python: Digunakan sebagai bahasa pemrograman utama untuk membangun sistem pemrosesan citra dan absensi.
- OpenCV: Digunakan sebagai pustaka pengolahan citra untuk proses deteksi wajah menggunakan Haar Cascade dan pengenalan wajah menggunakan LBPH [3], [4], [9], [16].
- Flask: Digunakan sebagai framework web server untuk menghubungkan proses pemrosesan citra, database, dan dashboard web [10].
- SQLite: Digunakan sebagai database untuk menyimpan data absensi yang meliputi ID pengguna, nama, dan waktu kehadiran [13].
- Web Browser (Dashboard Monitoring): Digunakan sebagai antarmuka untuk mengakses dashboard monitoring absensi.

3.2. Perancangan Sistem

Perancangan perangkat keras digambarkan melalui diagram blok yang menunjukkan hubungan antar komponen utama sistem. Secara keseluruhan, sistem terdiri dari tiga blok utama yaitu blok input, blok proses, dan blok output [12].



Gambar 2. Blok diagram sistem absensi wajah berbasis ESP32-CAM.

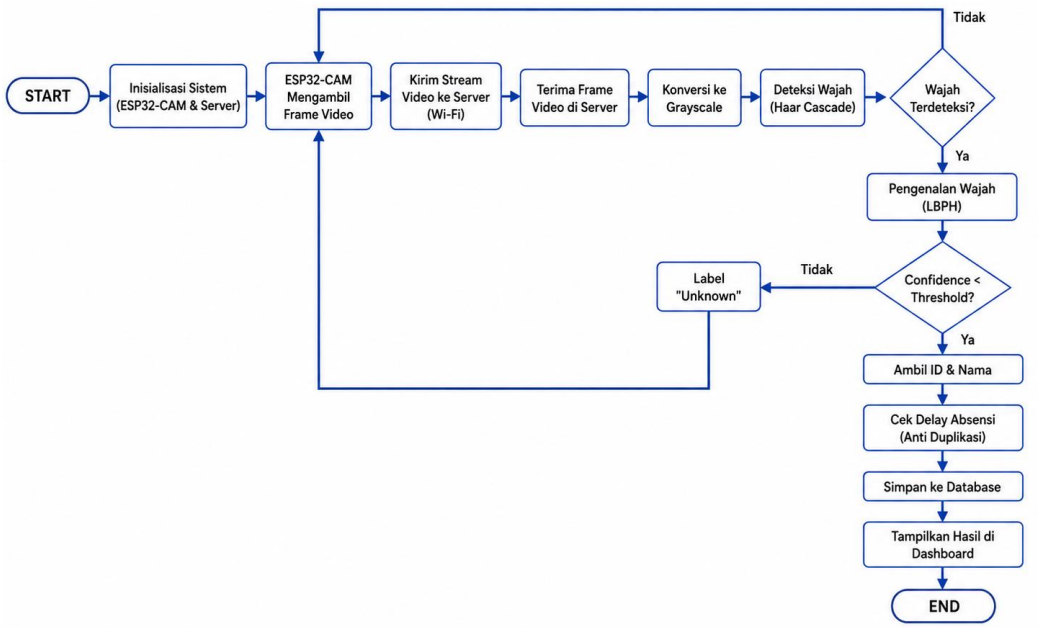
Blok input terdiri dari ESP32-CAM yang berfungsi sebagai perangkat akuisisi citra wajah. ESP32-CAM menangkap frame video wajah pengguna, kemudian mengirimkannya ke laptop/PC server melalui jaringan Wi-Fi dalam bentuk streaming [8]. Selain itu, lux meter dan laser distance meter digunakan sebagai alat bantu pengujian untuk mengukur intensitas cahaya dan jarak wajah terhadap kamera. Kedua alat ukur tersebut tidak terintegrasi langsung dengan sistem, tetapi digunakan sebagai instrumen pengujian untuk memperoleh data kondisi lingkungan secara terukur [11], [12].

Blok proses berada pada laptop/PC server. Pada server ini dijalankan aplikasi Python berbasis Flask dan OpenCV [9], [10], [16]. Server menerima frame video dari ESP32-CAM, kemudian mengubah citra menjadi grayscale untuk diproses menggunakan metode Haar Cascade. Metode Haar Cascade atau Viola-Jones menerima input berupa citra grayscale dan menghasilkan output berupa lokasi wajah dalam bentuk koordinat bounding box, yaitu x, y, w, dan h [3]. Koordinat tersebut menunjukkan posisi area wajah pada citra.

Area wajah yang berhasil dideteksi kemudian dipotong atau di-crop berdasarkan koordinat bounding box. Citra wajah hasil cropping selanjutnya digunakan sebagai input pada proses pengenalan wajah menggunakan metode Local Binary Pattern Histogram (LBPH). Pada tahap pengenalan, LBPH membandingkan citra wajah masukan dengan model hasil training. Output dari LBPH bukan langsung berupa nama pengguna, melainkan berupa label prediksi dan nilai confidence [4]. Label prediksi kemudian dipetakan ke data pengguna untuk memperoleh ID dan nama.

Blok output terdiri dari database SQLite dan web dashboard. Database SQLite berada pada laptop/PC server yang sama dengan aplikasi Python. Database digunakan untuk menyimpan data absensi berupa ID, nama, dan waktu kehadiran [13]. Web dashboard merupakan antarmuka berbasis browser yang disediakan oleh Flask untuk menampilkan live streaming, data absensi, filter tanggal, dan fitur ekspor data [10].

Perancangan perangkat lunak dikembangkan menggunakan bahasa pemrograman Python dengan dukungan OpenCV untuk pengolahan citra dan Flask sebagai web server [9], [10], [16]. Alur kerja sistem secara keseluruhan ditunjukkan pada Gambar 3.



Gambar 3. Flowchart cara kerja sistem absensi wajah berbasis ESP32-CAM.

3.3. Proses Pengambilan Dataset, Training, dan Pengenalan Wajah

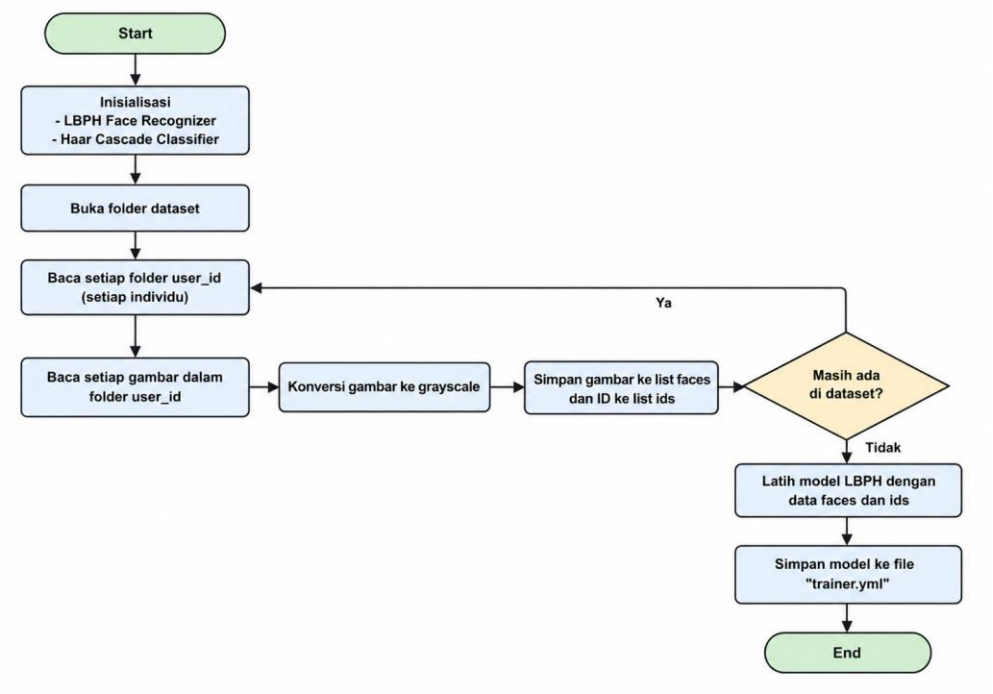
Sistem absensi berbasis pengenalan wajah pada penelitian ini terdiri dari tiga tahapan utama, yaitu pengambilan dataset wajah, proses training model, dan proses pengenalan wajah pada saat sistem berjalan. Tahapan ini dilakukan agar sistem tidak hanya mampu mendeteksi keberadaan wajah, tetapi juga mampu mengenali identitas pengguna yang telah terdaftar pada dataset. Sistem pengenalan wajah secara umum bekerja dengan membandingkan citra wajah masukan dengan data wajah yang telah tersimpan sebelumnya dalam model atau basis data [2].

Tahap pertama adalah pengambilan dataset wajah. Dataset diperoleh dengan cara mengambil citra wajah pengguna menggunakan kamera, kemudian menyimpan hasil citra tersebut ke dalam folder berdasarkan label identitas masing-masing pengguna. Setiap pengguna direpresentasikan menggunakan satu label numerik, misalnya folder 1 untuk pengguna pertama, folder 2 untuk pengguna kedua, dan seterusnya. Pada implementasi sistem, setiap pengguna diambil sebanyak 50 citra wajah. Dataset pada penelitian ini terdiri dari 3 pengguna, dengan masing-masing pengguna memiliki 50 citra wajah, sehingga total dataset yang digunakan adalah 150 citra wajah. Dataset diambil dengan beberapa variasi sederhana, seperti perbedaan posisi wajah, ekspresi wajah, jarak pengambilan, dan kondisi pencahayaan. Variasi dataset diperlukan karena kualitas citra, posisi wajah, dan pencahayaan dapat mempengaruhi keberhasilan sistem dalam mendeteksi dan mengenali wajah [2], [11].

Tahap kedua adalah proses training model. Pada tahap ini, seluruh citra wajah yang tersimpan dalam folder dataset dibaca oleh sistem menggunakan OpenCV [9], [16]. Citra wajah kemudian dikonversi menjadi grayscale karena Haar Cascade dan LBPH bekerja berdasarkan informasi intensitas piksel [3], [4]. Setelah itu, setiap citra wajah dipasangkan dengan label numerik sesuai folder pengguna. Citra wajah disimpan dalam variabel faces, sedangkan label identitas disimpan dalam variabel ids.

Proses training dilakukan menggunakan algoritma Local Binary Pattern Histogram (LBPH). Input pada proses training LBPH adalah kumpulan citra wajah grayscale dan label numerik dari masing-masing pengguna. Algoritma LBPH dikembangkan dari konsep Local Binary Pattern yang bekerja dengan membandingkan nilai intensitas piksel pusat terhadap piksel di sekitarnya untuk membentuk pola biner lokal, kemudian pola tersebut direpresentasikan dalam bentuk histogram sebagai ciri wajah [4]. Hasil dari proses training adalah model pengenalan wajah yang disimpan dalam file trainer.yml. File model ini kemudian digunakan pada tahap pengenalan wajah.

Secara keseluruhan, alur proses pengambilan dataset, pembacaan citra, pemberian label, pelatihan model LBPH, hingga penyimpanan model ditunjukkan pada Gambar 4 [4], [17].



Gambar 4. Flowchart proses training pengenalan wajah

Tahap ketiga adalah proses pengenalan wajah. Pada saat sistem berjalan, ESP32-CAM menangkap frame video dan mengirimkannya ke server Python melalui jaringan Wi-Fi. ESP32-CAM digunakan sebagai perangkat akuisisi citra karena memiliki modul kamera dan konektivitas Wi-Fi yang sesuai untuk sistem berbasis IoT [8]. Frame yang diterima server dikonversi menjadi grayscale, kemudian diproses menggunakan metode Haar Cascade. Input Haar Cascade berupa citra grayscale, sedangkan outputnya berupa koordinat area wajah dalam bentuk bounding box x, y, w, dan h [3].

Area wajah yang telah terdeteksi kemudian dipotong berdasarkan koordinat bounding box. Citra wajah hasil cropping menjadi input untuk algoritma LBPH pada tahap pengenalan [3], [4]. Pada tahap ini, LBPH membandingkan citra wajah masukan dengan model hasil training yang tersimpan dalam file trainer.yml [4].

Output dari algoritma LBPH bukan langsung berupa nama pengguna, melainkan berupa label prediksi dan nilai confidence. Label prediksi adalah angka yang menunjukkan label pengguna yang paling mendekati data training, sedangkan nilai confidence menunjukkan jarak atau tingkat perbedaan antara citra uji dengan data wajah pada model. Semakin kecil nilai confidence, maka wajah yang diuji semakin mirip dengan data wajah pada model [4]. Label prediksi kemudian dipetakan ke data pengguna untuk memperoleh ID dan nama pengguna. Dengan demikian, ID dan nama yang ditampilkan pada sistem merupakan hasil pemetaan dari label prediksi LBPH, bukan output langsung dari algoritma LBPH.

Apabila nilai confidence berada di bawah threshold yang telah ditentukan, maka sistem menganggap wajah berhasil dikenali [4]. Selanjutnya, sistem mengambil ID dan nama pengguna, memeriksa kemungkinan duplikasi absensi dalam waktu tertentu, lalu menyimpan data kehadiran ke database SQLite [13]. Data yang disimpan meliputi ID pengguna, nama, dan waktu kehadiran. Sebaliknya, apabila nilai confidence melebihi threshold, wajah diberi label "Unknown" dan tidak disimpan sebagai data absensi.

3.4. Tahapan Pengujian

Pengujian sistem dilakukan untuk memvalidasi kinerja sistem dalam mendeteksi wajah, mengenali pengguna, dan mencatat data absensi [12]. Tahapan pengujian yang dilakukan adalah sebagai berikut:

- Pengujian Fungsional: Memastikan seluruh komponen sistem, yaitu ESP32-CAM, server Python, OpenCV, SQLite, dan web dashboard, berfungsi sesuai dengan perancangan [8], [9], [10], [13].
- Pengujian Deteksi Wajah: Menguji kemampuan Haar Cascade dalam mendeteksi area wajah pada frame video yang diterima dari ESP32-CAM [3].
- Pengujian Pengenalan Wajah: Menguji kemampuan LBPH dalam mengenali identitas pengguna berdasarkan model training yang telah dibuat [4].
- Pengujian Intensitas Cahaya: Menguji performa sistem pada variasi pencahayaan 50 lux, 150 lux, dan 300 lux yang diukur menggunakan lux meter [11].
- Pengujian Jarak Wajah ke Kamera: Menguji performa sistem pada variasi jarak 30 cm, 50 cm, dan 100 cm yang diukur menggunakan laser distance meter.
- Pengujian Tingkat Keberhasilan Deteksi dan Pengenalan Wajah: Menghitung persentase keberhasilan sistem dalam mendeteksi dan mengenali wajah berdasarkan jumlah percobaan yang dilakukan [12].
- Pengujian Performa Komputasi: Mengukur waktu respons sistem, FPS streaming, latensi jaringan, serta penggunaan CPU dan RAM pada laptop/PC server.

3.5. Analisis dan Evaluasi Data

Data hasil pengujian dianalisis secara deskriptif kuantitatif [12]. Analisis dilakukan dengan membandingkan tingkat keberhasilan sistem pada setiap kombinasi intensitas cahaya dan jarak wajah terhadap kamera [11], [12]. Parameter yang diamati meliputi keberhasilan deteksi wajah menggunakan Haar Cascade dan keberhasilan pengenalan wajah menggunakan LBPH [3], [4]. Parameter utama yang digunakan dalam evaluasi adalah tingkat keberhasilan deteksi wajah dan pengenalan wajah yang dihitung menggunakan persamaan berikut [12]:

$$\boldsymbol{A = \frac{n}{N}\, X\, 100\%}$$

Keterangan:
A = tingkat keberhasilan (%)
n = jumlah percobaan berhasil
N = total percobaan

Pada penelitian ini, setiap kombinasi intensitas cahaya dan jarak diuji sebanyak 10 kali percobaan. Variasi intensitas cahaya yang digunakan adalah 50 lux, 150 lux, dan 300 lux, sedangkan variasi jarak yang digunakan adalah 30 cm, 50 cm, dan 100 cm. Dengan demikian, terdapat 9 skenario pengujian dengan total 90 percobaan. Hasil pengujian kemudian digunakan untuk menentukan kondisi optimal sistem dalam mendeteksi dan mengenali wajah.

Selain tingkat keberhasilan, data performa komputasi dianalisis berdasarkan waktu respons sistem, FPS streaming, latensi jaringan, penggunaan CPU, dan penggunaan RAM. Waktu respons dihitung dari saat frame diterima oleh server hingga proses deteksi, pengenalan, dan encoding frame selesai dilakukan. FPS dihitung berdasarkan jumlah frame yang berhasil diproses setiap detik, latensi jaringan diukur melalui pengujian ping terhadap alamat IP ESP32-CAM, sedangkan penggunaan CPU dan RAM dicatat menggunakan pustaka psutil dengan interval pencatatan setiap 1 detik.

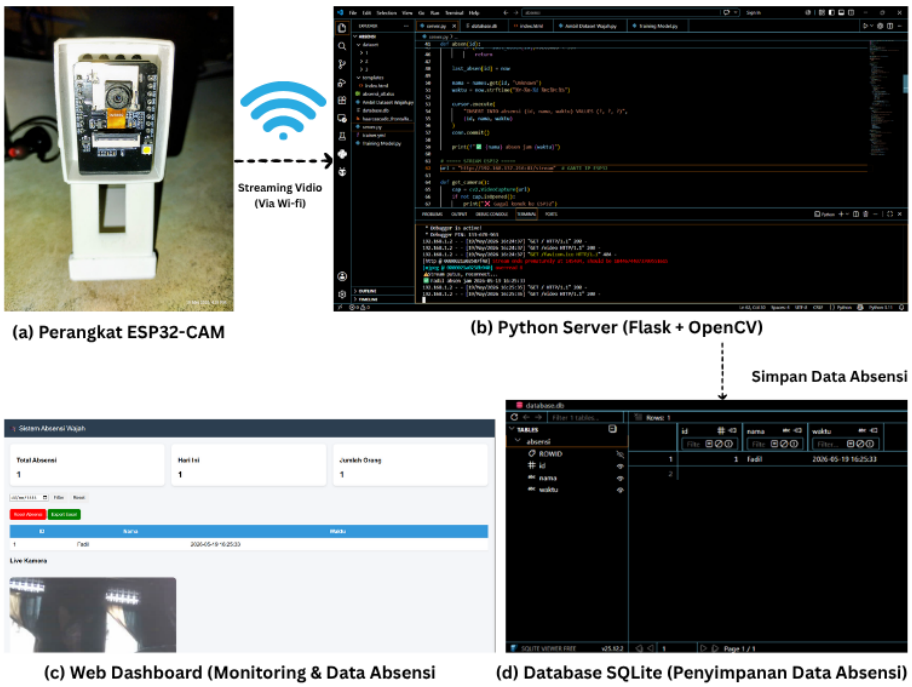
4. Hasil dan Pembahasan

4.1. Implementasi Sistem

Sistem absensi berbasis pengenalan wajah pada penelitian ini diimplementasikan menggunakan ESP32-CAM sebagai perangkat akuisisi citra wajah dan laptop/PC server

sebagai pusat pemrosesan data [8]. ESP32-CAM berfungsi menangkap frame video wajah pengguna, kemudian mengirimkan frame tersebut ke server melalui jaringan Wi-Fi. Pada sisi server, sistem dikembangkan menggunakan bahasa pemrograman Python dengan dukungan Flask, OpenCV, dan SQLite [9], [10], [13], [16].

Server Python berperan untuk menerima frame video dari ESP32-CAM, melakukan konversi citra ke grayscale, mendeteksi wajah menggunakan metode Haar Cascade, serta mengenali wajah menggunakan metode Local Binary Pattern Histogram (LBPH) [3], [4], [9], [16]. Hasil pengenalan wajah kemudian dipetakan ke data pengguna untuk memperoleh ID dan nama pengguna. Setelah identitas pengguna dikenali, sistem menyimpan data absensi ke dalam database SQLite berupa ID, nama, dan waktu kehadiran [13].



Gambar 5. Implementasi sistem absensi berbasis ESP32-CAM: (a) perangkat ESP32-CAM, (b) server Python, (c) web dashboard, dan (d) database SQLite.

Dari sisi perangkat lunak, sistem terdiri dari beberapa modul utama, yaitu modul pengambilan dataset wajah, modul training model LBPH, modul deteksi dan pengenalan wajah, modul penyimpanan database, serta modul web dashboard [4], [10], [13], [17]. Web dashboard dikembangkan menggunakan Flask untuk menampilkan streaming kamera, data absensi, fitur filter tanggal, serta ekspor data ke dalam format Excel [10]. Dengan integrasi tersebut, sistem tidak hanya melakukan deteksi dan pengenalan wajah, tetapi juga mencatat dan menampilkan data absensi secara terstruktur.

4.2. Pengujian Fungsional

Pengujian fungsional dilakukan untuk memastikan setiap komponen sistem bekerja sesuai dengan rancangan [12]. Pengujian ini mencakup fungsi akuisisi citra oleh ESP32-CAM, deteksi wajah menggunakan Haar Cascade, pengenalan wajah menggunakan LBPH, penyimpanan data ke SQLite, serta tampilan data pada web dashboard [3], [4], [8], [10], [13]. Hasil pengujian fungsional disajikan pada Tabel 1.

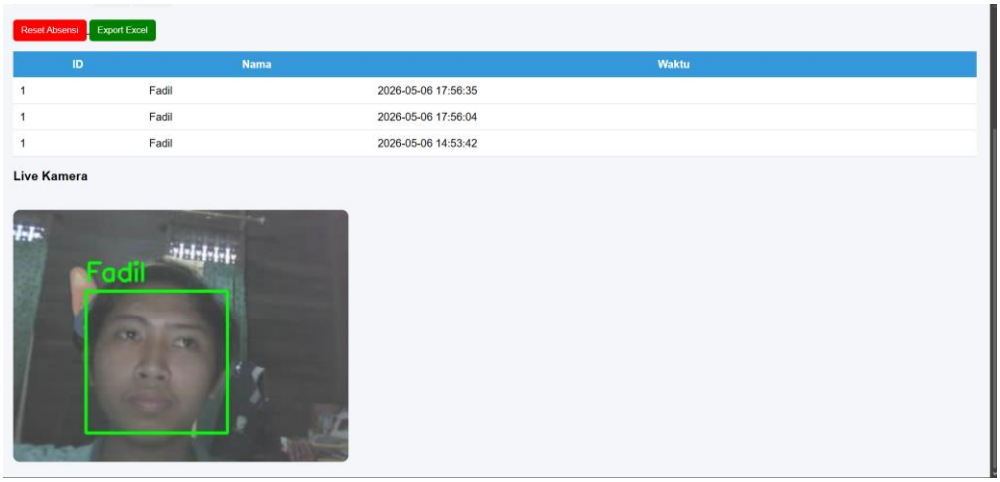
Tabel 1. Hasil Pengujian Fungsional Sistem Absensi.

NO	Komponen	Fungsi yang diuji	Hasil	Status
1	ESP32-CAM	Mengirim frame video ke server	Video streaming berhasil ditampilkan pada dashboard	Berhasil

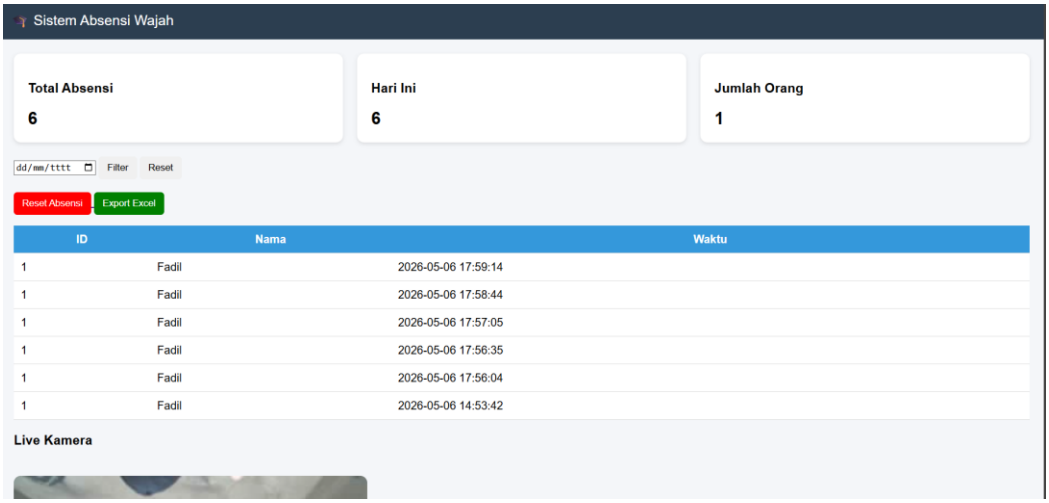
2	OpenCV (Haar Cascade)	Deteksi wajah	Area wajah berhasil terdeteksi pada frame video	Berhasil
3	OpenCV (LBPH)	Pengenalan wajah	Wajah dikenali sesuai dataset training	Berhasil
4	Database (SQLite)	Penyimpanan data absensi	Data ID, nama, dan waktu tersimpan dengan benar	Berhasil
5	Flask Server	Integrasi kamera, pengenalan wajah, dan database	Sistem berjalan sesuai alur yang dirancang	Berhasil
6	Web Dashboard	Menampilkan data absensi	Data absensi tampil setelah tersimpan di database	Berhasil
7	Sistem Keseluruhan	Integrasi seluruh komponen	Semua komponen dapat bekerja secara terpadu	Berhasil

Berdasarkan Tabel 1, seluruh komponen sistem berhasil menjalankan fungsi utamanya. ESP32-CAM mampu mengirimkan frame video ke server, sedangkan server Python berhasil memproses citra menggunakan OpenCV. Metode Haar Cascade mampu mendeteksi area wajah pada frame video, kemudian citra wajah hasil deteksi digunakan oleh LBPH untuk proses pengenalan wajah berdasarkan model training yang telah dibuat sebelumnya.

Database SQLite berhasil menyimpan data absensi ketika wajah dikenali oleh sistem. Data yang tersimpan kemudian ditampilkan pada web dashboard. Hasil ini menunjukkan bahwa integrasi antara ESP32-CAM, server Python, OpenCV, SQLite, dan web dashboard telah berjalan sesuai dengan perancangan sistem.



Gambar 6. Tampilan deteksi dan pengenalan wajah secara real-time.



Gambar 7. Tampilan web dashboard sistem absensi berbasis pengenalan wajah.

4.3. Pengujian Intensitas Cahaya dan Jarak

Pengujian intensitas cahaya dan jarak dilakukan untuk mengetahui pengaruh kondisi lingkungan terhadap keberhasilan sistem dalam mendeteksi dan mengenali wajah [11], [12]. Pengujian dilakukan pada tiga tingkat pencahayaan, yaitu 50 lux, 150 lux, dan 300 lux. Variasi jarak yang digunakan adalah 30 cm, 50 cm, dan 100 cm.

Setiap kombinasi intensitas cahaya dan jarak diuji sebanyak 10 kali percobaan, sehingga total pengujian yang dilakukan adalah 90 percobaan [12]. Intensitas cahaya diukur menggunakan lux meter, sedangkan jarak wajah terhadap ESP32-CAM diukur menggunakan laser distance meter. Parameter yang diamati meliputi keberhasilan deteksi wajah menggunakan Haar Cascade dan keberhasilan pengenalan wajah menggunakan LBPH [3], [4]. Hasil pengujian disajikan pada Tabel 2.

Tabel 2. Hasil Pengujian Intensitas Cahaya dan Jarak.

NO	Intensitas (Lux)	Jarak (cm)	Jumlah Percobaan	Deteksi wajah	Pengenalan Wajah
1	50	30	10	10/10	10/10
2	50	50	10	10/10	0/10
3	50	100	10	10/10	0/10
4	150	30	10	10/10	10/10
5	150	50	10	10/10	10/10
6	150	100	10	10/10	0/10
7	300	30	10	10/10	10/10
8	300	50	10	10/10	10/10
9	300	100	10	10/10	0/10

Berdasarkan Tabel 2, sistem berhasil mendeteksi wajah pada seluruh skenario pengujian dengan hasil deteksi 10/10 pada setiap kombinasi intensitas cahaya dan jarak. Hal ini menunjukkan bahwa metode Haar Cascade masih mampu menemukan area wajah pada seluruh kondisi pengujian [3].

Namun, hasil pengenalan wajah menunjukkan perbedaan pada setiap skenario. Pada jarak 30 cm, sistem berhasil mengenali wajah pada seluruh tingkat pencahayaan, yaitu 50 lux, 150 lux, dan 300 lux. Pada jarak 50 cm, sistem berhasil mengenali wajah pada intensitas cahaya 150 lux dan 300 lux, tetapi gagal mengenali wajah pada intensitas cahaya 50 lux. Sementara itu, pada jarak 100 cm, sistem tidak berhasil mengenali wajah pada seluruh kondisi pencahayaan meskipun wajah tetap berhasil terdeteksi.

Hasil tersebut menunjukkan bahwa proses deteksi wajah dan pengenalan wajah memiliki karakteristik yang berbeda. Deteksi wajah hanya menentukan lokasi wajah pada citra,

sedangkan pengenalan wajah membutuhkan kualitas citra yang cukup baik agar fitur tekstur wajah dapat dibandingkan dengan model training [3], [4]. Dokumentasi pengujian intensitas cahaya dan jarak ditunjukkan pada Gambar 8.

Gambar 8. Alat pengujian intensitas cahaya dan jarak.

4.4. Analisis Tingkat Keberhasilan Pengenalan Wajah

Tingkat keberhasilan pengenalan wajah dihitung berdasarkan jumlah percobaan yang berhasil mengenali wajah dibandingkan dengan total percobaan pada setiap variasi jarak. Persamaan yang digunakan adalah sebagai berikut [12]:

$$\boldsymbol{A = \frac{n}{N}\ X\ 100\%}$$

Keterangan:
A = tingkat keberhasilan (%)
n = jumlah percobaan berhasil
N = total percobaan

Pada setiap variasi jarak terdapat tiga kondisi intensitas cahaya, yaitu 50 lux, 150 lux, dan 300 lux. Masing-masing kondisi diuji sebanyak 10 kali, sehingga total percobaan pada setiap jarak adalah 30 percobaan. Selain itu, total seluruh skenario pengujian adalah 90 percobaan.

Perhitungan tingkat keberhasilan deteksi dan pengenalan wajah secara keseluruhan adalah sebagai berikut:

$$\boldsymbol{A_{deteksi} = \frac{90}{90}\ X\ 100\% = 100\%}$$

$$\boldsymbol{A_{pengenalan} = \frac{50}{90}\ X\ 100\% = 55,56\%}$$

Perhitungan tingkat keberhasilan pengenalan wajah berdasarkan jarak adalah sebagai berikut:

$$A_{30cm} = \frac{30}{30} X\ 100\% = 100\%$$

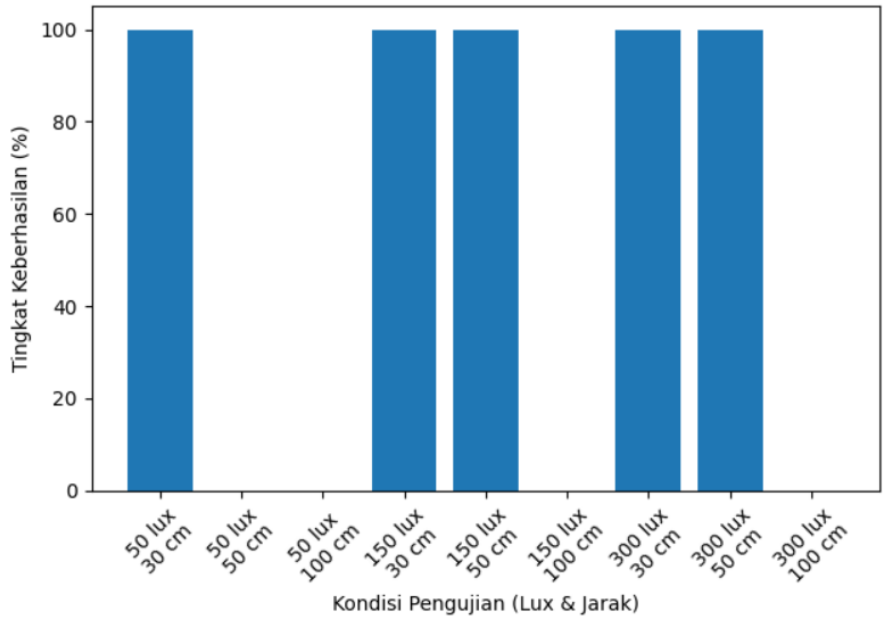
$$A_{50cm} = \frac{20}{30} X\ 100\% = 66,7\%$$

$$A_{100cm} = \frac{0}{30} X\ 100\% = 0\%$$

Jika dilihat berdasarkan intensitas cahaya, tingkat keberhasilan pengenalan wajah adalah 33,3% pada 50 lux, 66,7% pada 150 lux, dan 66,7% pada 300 lux. Hasil ini menunjukkan bahwa keberhasilan pengenalan wajah tidak hanya dipengaruhi oleh pencahayaan, tetapi juga sangat dipengaruhi oleh jarak wajah terhadap kamera.

Pada penelitian ini, istilah akurasi digunakan sebagai tingkat keberhasilan sederhana berdasarkan jumlah percobaan berhasil terhadap total percobaan. Evaluasi precision, recall, dan confusion matrix belum dilakukan karena data pengujian yang tersedia belum mencatat label aktual dan label prediksi untuk setiap kelas pengguna pada setiap percobaan. Perhitungan metrik tersebut memerlukan data prediksi per pengguna, misalnya apakah pengguna A dikenali sebagai pengguna A, pengguna lain, atau Unknown.

Berdasarkan hasil perhitungan tersebut, tingkat keberhasilan pengenalan wajah pada jarak 30 cm adalah 100%, pada jarak 50 cm sebesar 66,7%, dan pada jarak 100 cm sebesar 0%. Hasil ini menunjukkan bahwa semakin jauh jarak wajah terhadap kamera, semakin rendah tingkat keberhasilan sistem dalam mengenali wajah [2], [4].



Gambar 9. Grafik tingkat keberhasilan pengenalan wajah terhadap variasi intensitas cahaya dan jarak.

Gambar 9 menunjukkan bahwa performa pengenalan wajah mengalami penurunan seiring bertambahnya jarak. Kondisi terbaik diperoleh pada jarak 30 cm karena wajah masih tertangkap dengan ukuran yang cukup besar dan detail tekstur wajah masih dapat dikenali [4]. Pada jarak 50 cm, sistem masih mampu mengenali wajah pada intensitas cahaya 150 lux dan 300 lux, tetapi gagal pada 50 lux. Pada jarak 100 cm, sistem gagal mengenali wajah pada seluruh kondisi pencahayaan.

4.5. Pengujian Performa Komputasi

Pengujian performa komputasi dilakukan untuk mengetahui kemampuan sistem dalam menjalankan proses streaming, deteksi wajah, pengenalan wajah, dan penyimpanan data absensi. Parameter yang diuji meliputi waktu respons sistem, FPS streaming, latensi jaringan, serta penggunaan CPU dan RAM pada laptop/PC server. Waktu respons dihitung dari saat frame berhasil diterima oleh server hingga proses deteksi, pengenalan, dan encoding frame selesai dilakukan. FPS dihitung berdasarkan jumlah frame yang berhasil diproses setiap detik. Latensi jaringan diperoleh melalui pengujian ping terhadap alamat IP ESP32-CAM, sedangkan penggunaan CPU dan RAM dicatat dengan interval 1 detik pada proses server Python.

Tabel 3. Hasil Pengujian Performa Komputasi Sistem.

No	Parameter	Minimum	Rata-rata	Maksimum
1	Waktu respons	3,36 ms	8,98 ms	318,03 ms
2	FPS streaming	-	22,59 FPS	34,78 FPS
3	Latensi jaringan	2 ms	7 ms	38 ms
4	CPU server	0,00%	5,44%	13,86%
5	RAM server	142,61 MB	143,69 MB	144,75 MB

Berdasarkan Tabel 3, sistem memperoleh waktu respons rata-rata sebesar 8,98 ms dengan FPS streaming rata-rata 22,59 FPS. Nilai latensi jaringan rata-rata sebesar 7 ms menunjukkan bahwa komunikasi antara ESP32-CAM dan server berada pada kondisi yang cukup rendah untuk kebutuhan monitoring absensi pada jaringan lokal. Penggunaan CPU rata-rata sebesar 5,44% dan RAM rata-rata sebesar 143,69 MB menunjukkan bahwa beban komputasi pada laptop/PC server relatif rendah selama proses pengujian berlangsung.

Hasil pengujian performa tersebut menunjukkan bahwa sistem mampu menjalankan proses streaming dan pengenalan wajah secara cukup responsif pada server lokal. Namun, nilai waktu respons maksimum sebesar 318,03 ms menunjukkan adanya lonjakan waktu proses pada kondisi tertentu, misalnya saat proses pembacaan frame, deteksi wajah, encoding citra, atau komunikasi jaringan mengalami keterlambatan. Oleh karena itu, optimasi koneksi Wi-Fi, resolusi frame, dan proses pemrosesan citra masih diperlukan apabila sistem dikembangkan untuk jumlah pengguna atau beban akses yang lebih besar.

4.6. Pembahasan

Hasil pengujian menunjukkan bahwa jarak wajah terhadap kamera menjadi faktor utama yang mempengaruhi keberhasilan pengenalan wajah. Pada jarak yang lebih dekat, wajah terlihat lebih besar pada citra sehingga informasi tekstur lokal yang digunakan oleh metode LBPH dapat diekstraksi dengan lebih baik. Sebaliknya, pada jarak yang lebih jauh, ukuran wajah dalam citra menjadi lebih kecil sehingga detail tekstur wajah berkurang. Kondisi ini menyebabkan proses pencocokan wajah dengan model training menjadi kurang optimal [4].

Intensitas cahaya juga berpengaruh terhadap keberhasilan pengenalan wajah, tetapi pengaruhnya terlihat lebih jelas ketika jarak wajah terhadap kamera bertambah [11]. Pada intensitas cahaya 50 lux, sistem masih mampu mengenali wajah pada jarak 30 cm, tetapi gagal pada jarak 50 cm dan 100 cm. Hal ini menunjukkan bahwa pencahayaan rendah masih dapat ditoleransi selama wajah berada dekat dengan kamera. Namun, ketika jarak bertambah, kualitas citra menjadi kurang memadai untuk proses pengenalan wajah [2], [11].

Pada intensitas cahaya 150 lux dan 300 lux, sistem mampu mengenali wajah pada jarak 30 cm dan 50 cm. Kondisi ini menunjukkan bahwa pencahayaan yang cukup membantu kamera menangkap detail wajah dengan lebih baik [11]. Namun, pada jarak 100 cm, sistem tetap gagal mengenali wajah meskipun intensitas cahaya meningkat. Hal ini menunjukkan bahwa peningkatan pencahayaan tidak sepenuhnya dapat mengatasi penurunan kualitas citra akibat jarak yang terlalu jauh [2], [8].

Metode Haar Cascade menunjukkan hasil deteksi yang konsisten pada seluruh kondisi pengujian karena metode ini hanya mencari pola wajah dan menghasilkan lokasi wajah dalam bentuk bounding box. Namun, keberhasilan LBPH dalam mengenali identitas pengguna sangat bergantung pada kualitas citra wajah hasil cropping. Hal ini sesuai dengan karakteristik LBPH yang menggunakan tekstur lokal sebagai ciri utama dalam proses pengenalan wajah [4].

Selain itu, keterbatasan kamera pada ESP32-CAM juga mempengaruhi hasil pengenalan wajah, terutama pada jarak yang lebih jauh. Resolusi dan kualitas citra yang dihasilkan ESP32-CAM membatasi detail wajah yang dapat digunakan pada proses pengenalan. Oleh karena itu, berdasarkan hasil pengujian, sistem bekerja paling baik pada jarak 30–50 cm dengan intensitas cahaya 150–300 lux. Hasil ini juga menunjukkan bahwa sistem masih memerlukan optimasi apabila ingin digunakan pada jarak lebih dari 50 cm atau pada kondisi pencahayaan rendah [8], [11].

5. Perbandingan, Novelty, dan Keterbatasan Penelitian

Dibandingkan dengan penelitian terdahulu, sistem yang dikembangkan dalam penelitian ini memiliki perbedaan pada aspek integrasi perangkat, metode pengenalan wajah, media akuisisi citra, serta ruang lingkup pengujian. Perbandingan pada penelitian ini bersifat berbasis literatur dan rancangan sistem, bukan perbandingan eksperimental langsung dengan metode lain, karena penelitian ini tidak menjalankan CNN, FaceNet, atau YOLO Face Detection pada dataset yang sama. Perbandingan tersebut tetap penting untuk memperjelas posisi penelitian dan kontribusi sistem yang dikembangkan.

Tabel 4. Perbandingan penelitian terdahulu dan metode terkait dengan penelitian ini.

Aspek	Penelitian/Metode Terdahulu	Penelitian Ini	Novelty/Keterangan
Akuisisi citra	Webcam lokal atau kamera umum [5], [7]	ESP32-CAM berbasis Wi-Fi [8], [20], [24]	Kamera dapat ditempatkan terpisah dari server selama masih dalam jaringan yang sama.
Deteksi wajah	OpenCV/Haar Cascade [3], [5] atau YOLO-Face modern [21], [23]	Haar Cascade [3]	Area wajah berhasil Metode sederhana dan ringan, tetapi belum sekuat YOLO untuk skala/occlusion kompleks.
Pengenalan wajah	CNN/FaceNet/deep learning [6], [18], [22]	LBPH [4]	Fokus pada metode sederhana untuk dataset kecil dan server lokal, bukan model deep learning.
Integrasi sistem	Sebagian penelitian fokus pada deteksi atau webcam lokal [5], [7]	SQLite + Flask web dashboard [10], [13]	Menggabungkan akuisisi, deteksi, pengenalan, penyimpanan, dan monitoring dalam satu alur.
Pengujian	Cahaya atau skenario terbatas [11]	50, 150, 300 lux dan 30, 50, 100 cm	Mengevaluasi kombinasi intensitas cahaya dan jarak terhadap deteksi serta pengenalan wajah.

Berdasarkan Tabel 4, novelty penelitian ini terletak pada integrasi ESP32-CAM sebagai perangkat akuisisi citra berbasis Wi-Fi, server Python sebagai pusat pemrosesan, metode Haar Cascade untuk deteksi wajah, metode LBPH untuk pengenalan wajah, SQLite sebagai media penyimpanan data absensi, serta web dashboard sebagai antarmuka monitoring. Selain itu, penelitian ini juga mengevaluasi pengaruh kombinasi intensitas cahaya dan jarak terhadap keberhasilan deteksi dan pengenalan wajah. Kombinasi integrasi sistem dan pengujian kondisi lingkungan tersebut menjadi pembeda utama dibandingkan penelitian terdahulu.

Penelitian ini tidak bertujuan untuk mengungguli metode berbasis deep learning seperti CNN, FaceNet, atau YOLO Face Detection, melainkan mengembangkan sistem absensi terintegrasi yang dapat dijalankan menggunakan ESP32-CAM sebagai perangkat akuisisi citra dan server lokal sebagai pusat pemrosesan. Metode deep learning tetap menjadi pembanding

penting karena penelitian terbaru menunjukkan bahwa pendekatan deep face recognition dan FaceNet memiliki kemampuan representasi fitur yang lebih kuat, terutama untuk variasi wajah kompleks, sedangkan YOLO-FaceV2 dan YOLO5Face dikembangkan untuk deteksi wajah pada variasi skala wajah dan kondisi occlusion [18], [21], [22], [23].

Meskipun demikian, penelitian ini masih memiliki beberapa keterbatasan. Pertama, dataset yang digunakan masih terbatas, yaitu 3 pengguna dengan masing-masing 50 citra wajah, sehingga hasil pengujian belum dapat digeneralisasikan untuk jumlah pengguna yang lebih besar. Kedua, pengujian hanya dilakukan pada tiga variasi intensitas cahaya dan tiga variasi jarak, sehingga belum mencakup variasi sudut wajah, penggunaan masker atau kacamata, latar belakang kompleks, dan deteksi beberapa pengguna secara bersamaan. Kondisi wajah tertutup sebagian atau occlusion, seperti penggunaan masker, dapat memengaruhi proses deteksi dan pengenalan wajah sehingga perlu diuji secara khusus pada penelitian berikutnya [21], [25]. Ketiga, penelitian ini belum melakukan evaluasi precision, recall, dan confusion matrix karena data pengujian belum mencatat label aktual dan label prediksi setiap pengguna pada setiap percobaan. Keempat, penelitian ini belum melakukan perbandingan eksperimental dengan metode lain seperti CNN, FaceNet, atau YOLO Face Detection pada dataset yang sama.

Berdasarkan keterbatasan tersebut, penelitian selanjutnya disarankan untuk menambah jumlah responden dan variasi dataset, memperluas skenario pengujian termasuk sudut wajah, background kompleks, multi-user detection, dan occlusion [21], [25], mencatat label aktual dan label prediksi setiap pengguna agar precision, recall, dan confusion matrix dapat dihitung, serta membandingkan performa LBPH dengan metode berbasis deep learning seperti FaceNet dan YOLO5Face pada kondisi pengujian yang sama [18], [22], [23]. Selain itu, pengujian performa komputasi dapat diperluas pada beberapa perangkat server dan kondisi jaringan yang berbeda agar kestabilan waktu respons, FPS, latensi, serta penggunaan CPU/RAM dapat dinilai secara lebih komprehensif.

6. Kesimpulan

Penelitian ini berhasil merancang dan mengimplementasikan sistem absensi berbasis pengenalan wajah menggunakan ESP32-CAM yang terhubung dengan laptop/PC server melalui jaringan Wi-Fi. Sistem ini mengintegrasikan beberapa komponen utama, yaitu ESP32-CAM sebagai perangkat akuisisi citra, server Python sebagai pusat pemrosesan, OpenCV sebagai pustaka pengolahan citra, SQLite sebagai media penyimpanan data absensi, serta web dashboard sebagai antarmuka monitoring.

Proses sistem dimulai dari pengambilan frame video oleh ESP32-CAM, kemudian frame dikirimkan ke server untuk diproses. Deteksi wajah dilakukan menggunakan metode Haar Cascade, sedangkan pengenalan wajah dilakukan menggunakan metode Local Binary Pattern Histogram (LBPH). Hasil pengenalan wajah berupa label prediksi dan nilai confidence kemudian dipetakan ke data pengguna untuk memperoleh ID dan nama. Apabila wajah berhasil dikenali, sistem menyimpan data absensi berupa ID, nama, dan waktu kehadiran ke dalam database SQLite.

Hasil pengujian menunjukkan bahwa sistem berhasil melakukan deteksi wajah pada seluruh skenario pengujian dengan hasil 10/10 pada setiap kombinasi intensitas cahaya dan jarak. Namun, keberhasilan pengenalan wajah dipengaruhi oleh jarak wajah terhadap kamera dan intensitas cahaya. Pada jarak 30 cm, sistem memperoleh tingkat keberhasilan pengenalan sebesar 100%. Pada jarak 50 cm, tingkat keberhasilan pengenalan menurun menjadi 66,7%, sedangkan pada jarak 100 cm sistem tidak berhasil mengenali wajah pada seluruh kondisi pencahayaan. Kondisi terbaik diperoleh pada jarak 30–50 cm dengan intensitas cahaya 150–300 lux.

Pengujian performa komputasi menunjukkan bahwa sistem memperoleh waktu respons rata-rata 8,98 ms, FPS streaming rata-rata 22,59 FPS, latensi jaringan rata-rata 7 ms, penggunaan CPU rata-rata 5,44%, dan penggunaan RAM rata-rata 143,69 MB pada laptop/PC server. Hasil ini menunjukkan bahwa sistem dapat menjalankan proses streaming, deteksi, dan pengenalan wajah dengan beban komputasi yang relatif rendah pada kondisi pengujian server lokal.

Berdasarkan hasil tersebut, dapat disimpulkan bahwa metode Haar Cascade mampu mendeteksi area wajah secara konsisten pada skenario pengujian yang dilakukan, sedangkan metode LBPH membutuhkan kualitas citra wajah yang cukup baik agar proses pengenalan

dapat berjalan optimal [3], [4]. Jarak pengambilan wajah menjadi faktor yang paling berpengaruh karena semakin jauh jarak wajah terhadap kamera, ukuran wajah pada citra semakin kecil sehingga detail tekstur wajah sulit diekstraksi [2], [4]. Selain itu, pencahayaan rendah juga dapat menurunkan keberhasilan pengenalan terutama pada jarak yang lebih jauh [11].

Untuk pengembangan lebih lanjut, disarankan: (1) menambah jumlah dan keragaman dataset wajah agar model lebih mampu mengenali wajah pada berbagai kondisi [2]; (2) menambahkan metode pembanding seperti algoritma berbasis deep learning, FaceNet, atau YOLO Face Detection untuk mengevaluasi perbedaan performa pada dataset yang sama [18], [21], [22], [23]; (3) meningkatkan kualitas kamera atau resolusi citra agar pengenalan wajah pada jarak lebih jauh dapat lebih optimal [8], [20], [24]; (4) menambahkan fitur keamanan seperti autentikasi pengguna dan perlindungan data absensi; serta (5) mengembangkan sistem penyimpanan berbasis cloud agar data absensi dapat diakses secara lebih luas [13].

Kontribusi Penulis: Konseptualisasi: MAFY dan AW; Metodologi: MAFY; Perangkat Lunak: MAFY; Validasi: MAFY, AW, dan DTM; Analisis formal: MAFY; Investigasi: MAFY; Sumber daya: AW; Kurasi data: MAFY; Penulisan—persiapan draf asli: MAFY; Penulisan—peninjauan dan penyuntingan: AW dan DTM; Supervisi: AW.

Pendanaan: Penelitian ini tidak menerima pendanaan eksternal.

Pernyataan Ketersediaan Data: Data hasil pengujian sistem tersedia atas permintaan kepada penulis yang sesuai.

Ucapan Terima Kasih: Penulis mengucapkan terima kasih kepada Bapak Agusma Wajiansyah dan Ibu Dwi Titi Maesaroh selaku dosen pembimbing di Politeknik Negeri Samarinda yang telah memberikan bimbingan, arahan, serta masukan selama proses penelitian dan penulisan makalah ini. Penulis juga menyatakan bahwa dalam proses penelitian ini, perangkat kecerdasan buatan (Artificial Intelligence/AI), yaitu ChatGPT (OpenAI) dan Claude (Anthropic), digunakan sebagai alat bantu dalam pengembangan perangkat lunak, khususnya untuk pembangkitan kode program (code generation) pada implementasi sistem berbasis Python, OpenCV, dan Flask. Seluruh keluaran yang dihasilkan oleh perangkat AI tersebut telah melalui proses peninjauan, modifikasi, dan validasi oleh penulis agar sesuai dengan kebutuhan sistem yang dikembangkan. Adapun seluruh ide penelitian, perancangan metodologi, analisis data, serta interpretasi hasil penelitian sepenuhnya merupakan hasil pemikiran dan tanggung jawab penulis.

Konflik Kepentingan: Penulis menyatakan tidak ada konflik kepentingan.

Referensi

[1] R. Szeliski, Computer Vision: Algorithms and Applications. New York, NY, USA: Springer, 2010.
[2] S. Z. Li and A. K. Jain, Handbook of Face Recognition, 2nd ed. London, U.K.: Springer, 2011.
[3] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR), 2001, pp. 511–518.
[4] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in Proc. European Conf. Computer Vision (ECCV), 2004, pp. 469–481.
[5] A. Saputra, R. Kurniawan, and D. Setiawan, "Implementasi OpenCV untuk deteksi wajah pada sistem berbasis Python," Jurnal Teknologi Informasi dan Komputer, vol. 5, no. 2, pp. 120–126, 2021.
[6] Y. Zhang, L. Wang, and H. Chen, "Real-time face recognition based on deep learning for embedded systems," IEEE Access, vol. 8, pp. 108234–108245, 2020.
[7] M. I. Putra, A. Pratama, and S. Wibowo, "Sistem absensi berbasis pengenalan wajah menggunakan webcam dan OpenCV," Jurnal Informatika, vol. 7, no. 1, pp. 45–52, 2022.
[8] Espressif Systems, "ESP32-CAM technical reference manual," 2022. [Online]. Available: https://www.espressif.com
[9] G. Bradski, "The OpenCV library," Dr. Dobb's Journal of Software Tools, 2000.
[10] M. Grinberg, Flask Web Development: Developing Web Applications with Python, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2018.
[11] F. Rahman, D. Nugroho, and E. Saputri, "Pengaruh intensitas cahaya terhadap akurasi deteksi wajah menggunakan metode Haar Cascade," Jurnal Ilmu Komputer dan Sistem Informasi, vol. 6, no. 3, pp. 210–217, 2021.
[12] Sugiyono, Metode Penelitian Kuantitatif, Kualitatif, dan R&D. Bandung, Indonesia: Alfabeta, 2017.
[13] F. Chollet, Deep Learning with Python. Shelter Island, NY, USA: Manning Publications, 2017.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems (NeurIPS), 2012, pp. 1097–1105.
[15] A. Rosebrock, "Face recognition with OpenCV and Python," PyImageSearch, 2018. [Online]. Available: https://pyimagesearch.com/2018/09/24/opencv-face-recognition/
[16] OpenCV, "Open source computer vision library documentation," 2023. [Online]. Available: https://opencv.org
[17] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," IEEE Signal Processing Letters, vol. 23, no. 10, pp. 1499–1503, 2016.
[18] M. Wang and W. Deng, "Deep face recognition: A survey," Neurocomputing, vol. 429, pp. 215–244, 2021, doi: 10.1016/j.neucom.2020.10.081.
[19] R. Singh and S. S. Gill, "Edge AI: A survey," Internet of Things and Cyber-Physical Systems, vol. 3, pp. 71–92, 2023, doi: 10.1016/j.iotcps.2023.02.004.
[20] P. D. P. Adi and Y. Wahyu, "A performance evaluation of ESP32 Camera Face Recognition for various projects," Internet of Things and Artificial Intelligence Journal, vol. 2, no. 1, pp. 10–21, 2022, doi: 10.31763/iota.v2i1.512.
[21] Z. Yu, H. Huang, W. Chen, Y. Su, Y. Liu, and X. Wang, "YOLO-FaceV2: A scale and occlusion aware face detector," Pattern Recognition, vol. 155, Art. no. 110714, 2024, doi: 10.1016/j.patcog.2024.110714.
[22] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815–823, doi: 10.1109/CVPR.2015.7298682.
[23] D. Qi, W. Tan, Q. Yao, and J. Liu, "YOLO5Face: Why reinventing a face detector," arXiv preprint arXiv:2105.12931, 2021.
[24] A. Hamidah, "Penerapan ESP32CAM untuk sistem absensi karyawan dengan face recognition," Jurnal Digit, 2024.
[25] F. Kurniawan et al., "Facemask detection using the YOLO-v5 algorithm," 2023.