# BAB II
# LANDASAN TEORI

## 2.1 Penelitian Terkait

Penelitian mengenai deteksi kondisi mental dan tingkat kewaspadaan berbasis citra wajah telah banyak dilakukan, khususnya dalam bidang transportasi dan industri. Salah satu penelitian oleh Essel (2024) mengembangkan sistem deteksi kantuk pada pengemudi menggunakan metode Convolutional Neural Network (CNN) berbasis analisis fitur wajah. Penelitian tersebut menunjukkan bahwa pendekatan computer vision mampu mendeteksi kondisi kantuk dengan tingkat akurasi yang tinggi, sehingga efektif digunakan sebagai sistem peringatan dini.

Selanjutnya, penelitian oleh Ayatuna Lazuardy (2023) berfokus pada pengenalan ekspresi wajah untuk mendeteksi kelelahan menggunakan teknik computer vision. Hasil penelitian menunjukkan bahwa analisis ekspresi mikro (micro-expression) dapat dimanfaatkan untuk mengidentifikasi kondisi kelelahan secara objektif. Hal ini memperkuat dasar bahwa citra wajah merupakan indikator yang valid dalam mendeteksi perubahan kondisi fisik maupun mental seseorang.

Dalam konteks pemilihan algoritma, penelitian yang dilakukan oleh Gubbala (2023) membandingkan beberapa metode klasifikasi untuk pengenalan emosi berbasis citra wajah. Hasilnya menunjukkan bahwa *Random Forest* memiliki performa yang stabil dan akurat dibandingkan beberapa algoritma lain seperti Support Vector Machine (SVM) dan K-Nearest Neighbor (KNN). *Random Forest* dinilai unggul dalam menangani data berdimensi tinggi dan mampu meminimalkan risiko overfitting.

Berdasarkan penelitian-penelitian tersebut, dapat disimpulkan bahwa teknologi computer vision dan algoritma machine learning efektif dalam mendeteksi kondisi seperti kelelahan dan kantuk. Namun, sebagian besar penelitian masih berfokus pada sektor transportasi atau industri. Penelitian yang secara khusus mengkaji klasifikasi kondisi belajar mahasiswa (fokus, lelah, dan mengantuk) berbasis algoritma *Random Forest* dalam konteks akademik masih terbatas. Oleh karena itu, penelitian ini diarahkan untuk mengisi celah tersebut dengan menerapkan metode klasifikasi berbasis citra wajah pada lingkungan pendidikan.

## 2.2 Pengolahan Citra Wajah (*Facial Image Processing*)

Pengolahan citra wajah (facial image processing) merupakan bagian dari bidang computer vision yang berfokus pada analisis, interpretasi, serta pemodelan informasi visual dari wajah manusia menggunakan algoritma komputasi. Wajah manusia mengandung berbagai informasi penting, seperti ekspresi, arah pandangan, tingkat kewaspadaan, hingga kondisi emosional. Dengan memanfaatkan teknik pengolahan citra dan pembelajaran mesin (machine learning), sistem komputer dapat mengenali pola-pola tertentu yang tidak selalu mudah diamati secara langsung oleh manusia (Nugroho, 2025).

Secara konseptual, pengolahan citra wajah bertujuan untuk mengubah data visual mentah (gambar atau video) menjadi representasi numerik yang dapat diproses oleh algoritma klasifikasi. Proses ini melibatkan transformasi citra menjadi bentuk data terstruktur melalui serangkaian tahapan sistematis. Dalam implementasinya, sistem harus mampu menangani variasi pencahayaan, sudut pengambilan gambar, resolusi kamera, serta perbedaan karakteristik wajah antar

individu. Oleh karena itu, diperlukan tahapan pemrosesan yang terstruktur agar informasi penting dari wajah dapat diekstraksi secara optimal. Secara umum, tahapan dalam pengolahan citra wajah meliputi:

1. Akuisisi Citra

   Tahap awal adalah proses pengambilan gambar menggunakan kamera atau webcam. Kualitas citra yang diperoleh sangat memengaruhi performa sistem secara keseluruhan. Faktor seperti resolusi kamera, pencahayaan ruangan, jarak subjek terhadap kamera, dan posisi wajah harus diperhatikan untuk menghasilkan citra yang jelas dan representative (Dong et al., 2009). Pada penelitian ini, citra wajah mahasiswa diambil dalam kondisi alami saat belajar untuk merepresentasikan situasi yang realistis.

2. Deteksi Wajah (Face Detection)

   Tahap ini bertujuan untuk mengidentifikasi dan memisahkan area wajah dari latar belakang. Proses deteksi wajah penting agar sistem hanya memproses bagian citra yang relevan. Algoritma deteksi wajah bekerja dengan mencari pola tertentu yang menyerupai struktur wajah manusia, seperti keberadaan mata, hidung, dan mulut. Hasil dari tahap ini berupa koordinat area wajah yang kemudian akan dipotong (cropping) untuk diproses lebih lanjut.

3. Pra-pemrosesan (Preprocessing)

   Pra-pemrosesan dilakukan untuk meningkatkan kualitas citra serta menyederhanakan proses analisis. Tahapan ini umumnya meliputi:

   - Konversi citra berwarna (RGB) menjadi grayscale untuk mengurangi kompleksitas komputasi.

- Normalisasi ukuran gambar agar seluruh dataset memiliki dimensi yang seragam.
- Pengurangan noise untuk meminimalkan gangguan visual yang dapat memengaruhi ekstraksi fitur.
- Penyesuaian kontras atau pencahayaan apabila diperlukan.
- Tahap ini sangat penting karena data yang bersih dan konsisten akan meningkatkan akurasi model klasifikasi.

4. Ekstraksi Fitur (Feature Extraction)

   Ekstraksi fitur merupakan inti dari pengolahan citra wajah. Pada tahap ini, sistem mengambil karakteristik numerik yang merepresentasikan kondisi wajah. Fitur yang umum digunakan dalam analisis kondisi belajar antara lain:

   - Posisi dan tingkat keterbukaan mata (indikator kantuk).
   - Frekuensi dan durasi kedipan mata.
   - Posisi kepala (tegak atau menunduk).
   - Bentuk dan pergerakan mulut (misalnya menguap).
   - Pola perubahan ekspresi wajah.

   Fitur-fitur tersebut diubah menjadi nilai numerik yang kemudian menjadi input bagi algoritma klasifikasi. Pemilihan fitur yang tepat sangat menentukan kemampuan sistem dalam membedakan kondisi fokus, lelah, dan mengantuk.

5. Klasifikasi

Tahap terakhir adalah mengelompokkan citra ke dalam kategori tertentu berdasarkan fitur yang telah diperoleh. Proses ini dilakukan menggunakan algoritma machine learning yang telah dilatih sebelumnya. Model klasifikasi akan mempelajari pola hubungan antara fitur wajah dan label kondisi belajar, kemudian memprediksi kategori pada data baru (Winanti et al., 2023).

## 2.3 Algoritma Random Forest

*Random Forest* merupakan salah satu algoritma machine learning berbasis metode ensemble learning yang широко digunakan untuk tugas klasifikasi maupun regresi. Konsep dasar ensemble learning adalah menggabungkan beberapa model sederhana untuk menghasilkan satu model yang lebih kuat dan memiliki performa lebih baik dibandingkan model tunggal. Dalam Random Forest, model dasar yang digunakan adalah decision tree

Algoritma ini bekerja dengan membangun sejumlah decision tree secara acak pada data latih, kemudian menggabungkan hasil prediksi dari masing-masing pohon untuk menentukan keputusan akhir. Pada kasus klasifikasi, hasil akhir diperoleh melalui mekanisme majority voting, yaitu kelas yang paling banyak diprediksi oleh seluruh pohon akan menjadi output akhir. Sementara itu, pada kasus regresi, hasil prediksi diperoleh melalui rata-rata nilai dari seluruh pohon.

Secara umum, proses kerja *Random Forest* dapat dijelaskan melalui beberapa tahapan berikut:

- Menentukan jumlah pohon (n_estimators) yang akan dibangun.

- Melakukan pengambilan sampel data latih secara acak dengan metode bootstrap sampling.
- Membangun decision tree untuk setiap sampel data.
- Pada setiap node dalam pohon, hanya sebagian fitur yang dipilih secara acak untuk menentukan pemisahan (split).
- Menggabungkan seluruh hasil prediksi pohon untuk memperoleh keputusan akhir.

Konsep dasar *Random Forest* melibatkan dua teknik utama, yaitu:

1. Bootstrap Sampling (Bagging)

   Metode ini mengambil data latih secara acak dengan pengembalian (sampling with replacement) untuk membangun setiap decision tree. Artinya, satu data dapat terpilih lebih dari satu kali, sementara data lain mungkin tidak terpilih dalam satu pohon tertentu. Teknik ini bertujuan untuk menciptakan variasi antar pohon sehingga model menjadi lebih robust dan tidak bergantung pada satu subset data tertentu.

2. Random Feature Selection

   Pada setiap proses pemisahan node dalam decision tree, algoritma tidak menggunakan seluruh fitur yang tersedia, melainkan hanya sebagian fitur yang dipilih secara acak. Teknik ini membantu mengurangi korelasi antar pohon dan meningkatkan keberagaman model. Dengan demikian, risiko overfitting dapat ditekan secara signifikan.

Dalam implementasinya, terdapat beberapa parameter yang memengaruhi performa model, antara lain:

- n_estimators: jumlah pohon yang dibangun dalam hutan. Semakin banyak pohon, umumnya semakin stabil hasil prediksi, namun membutuhkan waktu komputasi lebih besar.
- max_depth: kedalaman maksimum setiap pohon. Pembatasan kedalaman dapat membantu mencegah overfitting.
- min_samples_split: jumlah minimum sampel yang diperlukan untuk melakukan pemisahan node.
- max_features: jumlah fitur yang dipertimbangkan dalam setiap proses pemisahan.
- Penyesuaian parameter (hyperparameter tuning) dapat dilakukan untuk memperoleh performa model yang optimal.

Beberapa keunggulan *Random Forest* yang menjadikannya populer dalam berbagai penelitian antara lain:

- Mampu menangani data berdimensi tinggi dan kompleks, termasuk data citra yang memiliki banyak fitur.
- Mengurangi risiko overfitting dibandingkan decision tree tunggal.
- Stabil terhadap noise dan variasi data.
- Mendukung klasifikasi multi-kelas tanpa memerlukan modifikasi algoritma.

- Dapat mengukur tingkat kepentingan fitur (feature importance), sehingga membantu dalam proses analisis dan interpretasi model.
- Tidak memerlukan proses normalisasi data yang terlalu ketat seperti beberapa algoritma lain (misalnya SVM atau KNN).

Meskipun memiliki banyak keunggulan, *Random Forest* juga memiliki beberapa keterbatasan, antara lain:

- Membutuhkan sumber daya komputasi yang lebih besar dibandingkan decision tree tunggal.
- Model yang dihasilkan cenderung sulit diinterpretasikan secara sederhana karena terdiri dari banyak pohon.
- Waktu pelatihan bisa meningkat seiring bertambahnya jumlah pohon dan ukuran dataset.

Namun demikian, dalam banyak kasus klasifikasi citra, keunggulan yang dimiliki *Random Forest* jauh lebih signifikan dibandingkan keterbatasannya.

Dalam konteks pengolahan citra digital untuk klasifikasi ekspresi wajah, data masukan (*input*) yang digunakan oleh Random Forest tidak berbentuk tabel atribut kategorikal, melainkan berupa matriks piksel 2D. Setiap citra berukuran $48 \times 48$ piksel yang telah diubah ke skala kelabu (*grayscale*) diratakan (*flattening*) menjadi sebuah vektor fitur satu dimensi $X = [x_1, x_2, \dots, x_{2.304}]$, di mana setiap elemen $x_i \in [0{,}255]$ mewakili nilai intensitas derajat keabuan pada piksel ke-$i$. Vektor numerik berdimensi tinggi (2.304 fitur) inilah yang bertindak sebagai variabel prediktor untuk setiap *Decision Tree* di dalam *Random Forest*.

Mekanisme penting yang membedakan Random Forest dari Decision Tree biasa adalah pembatasan jumlah fitur yang dipertimbangkan pada setiap percabangan (node split). Jika total fitur citra adalah $p = 2.304$, maka pada setiap node, Random Forest hanya memilih subset fitur secara acak sejumlah $m$, di mana secara standar (default klasifikasi pada Scikit-learn):

$$m = \sqrt{p} = \sqrt{2.304} = 48 \text{ fitur}$$

Dengan hanya mengevaluasi 48 fitur piksel acak dari total 2.304 fitur pada setiap node, algoritma berhasil mematahkan korelasi antar-pohon (de-correlating trees). Hal ini memastikan bahwa pohon-pohon keputusan tidak selalu memilih piksel yang sama (misalnya area mata saja) sebagai root node, sehingga meningkatkan keberagaman pola ekspresi yang dipelajari.

Untuk menentukan fitur piksel mana yang paling optimal dalam memisahkan kondisi Fokus, Lelah, dan Mengantuk pada suatu node, digunakan kriteria Gini Impurity atau Entropy. Gini Impurity mengukur tingkat ketidakmurnian suatu node dan dirumuskan sebagai:

$$\text{Gini}(D) = 1 - \sum_{i=1}^{C} p_i^2$$

Keterangan:

- $C$ = Jumlah kelas target ($C = 3$, yaitu Fokus, Lelah, Mengantuk).
- $p_i$ = Proporsi sampel dari kelas i pada node tersebut.

Pemisahan terbaik (best split) dipilih berdasarkan nilai Information Gain atau penurunan Gini Impurity terbesar. Artinya, piksel yang paling efektif dalam membedakan antara mata terbuka (Fokus) dan mata tertutup (Mengantuk) akan diprioritaskan menjadi pemisah pada node tersebut.

Selain menghasilkan label prediksi akhir melalui mekanisme majority voting (suara terbanyak), Random Forest juga mampu menghitung nilai tingkat keyakinan (confidence level) dalam bentuk probabilitas untuk setiap kelas $c \in$ {Fokus,Lelah,Mengantuk}. Probabilitas kelas $P(y = c|X)$ dihitung berdasarkan rasio jumlah pohon yang memprediksi kelas c terhadap total seluruh pohon ($N_{trees}$):

$$P(y = c|X) = \frac{1}{T}\sum_{t=1}^{T} I\left(\hat{y}_t(X) = c\right)$$

Keterangan:

- T = Total estimator / pohon keputusan (n_estimators).
- $\hat{y}_t(X)$ = Prediksi kelas dari pohon ke-t.
- $I(\cdot)$ = Fungsi indikator yang bernilai 1 jika prediksi sesuai kelas c, dan 0 jika tidak.

Sebagai contoh, jika dibangun 100 pohon keputusan dan 80 pohon memprediksi citra tersebut sebagai 'Fokus', maka nilai probabilitas untuk kelas 'Fokus' adalah $\frac{80}{100} = 0{,}80$.

Meskipun Random Forest sangat handal dalam menangani data numerik berdimensi tinggi, penggunaan fitur berupa nilai piksel mentah (*raw pixel*

*intensities*) memiliki keterbatasan spasial. *Decision Tree* memperlakukan setiap piksel ke-$i$ secara independen tanpa memahami hubungan posisi geometris antar-piksel sekitarnya (seperti jarak antara kelopak mata atas dan bawah). Akibatnya, ketika dua kondisi memiliki nilai intensitas piksel yang mirip seperti mata yang menyipit pada kondisi Lelah dan mata tertutup pada kondisi Mengantuk algoritma Random Forest berbasis piksel mentah dapat mengalami sedikit penurunan akurasi klasifikasi.

## 2.4 Produktivitas Mahasiswa

Produktivitas mahasiswa dapat diartikan sebagai kemampuan mahasiswa dalam mengelola waktu, energi, perhatian, serta sumber daya yang dimiliki untuk mencapai hasil belajar yang optimal. Produktivitas tidak hanya diukur dari nilai akademik semata, tetapi juga dari tingkat partisipasi aktif dalam perkuliahan, konsistensi dalam menyelesaikan tugas, kemampuan memahami materi secara mendalam, serta keterlibatan dalam kegiatan akademik maupun non-akademik yang menunjang kompetensi diri. Dengan kata lain, produktivitas mencerminkan efektivitas proses belajar, bukan sekadar output akhir berupa nilai (Putri Hati Siregar et al., 2025).

Secara teoritis, produktivitas belajar berkaitan erat dengan manajemen diri (self-management), regulasi diri (self-regulated learning), serta motivasi intrinsik mahasiswa. Mahasiswa yang mampu mengatur jadwal belajar, menetapkan prioritas, dan menjaga konsistensi cenderung memiliki tingkat produktivitas yang lebih tinggi. Namun, dalam praktiknya, berbagai faktor internal dan eksternal dapat memengaruhi stabilitas produktivitas tersebut.

Dalam konteks pembelajaran modern, produktivitas mahasiswa sangat dipengaruhi oleh kondisi fisik dan mental. Faktor seperti tingkat fokus, kelelahan, stres akademik, beban tugas, serta kualitas tidur memiliki hubungan langsung dengan performa belajar. Kurangnya waktu istirahat dan tingginya tekanan akademik dapat menyebabkan penurunan konsentrasi, lambatnya pemrosesan informasi, serta berkurangnya daya ingat jangka pendek. Kondisi ini secara langsung berdampak pada kemampuan mahasiswa dalam memahami materi dan berpartisipasi secara aktif (Muhammad Yusuf Gunawan et al., 2026).

Model pembelajaran daring dan hybrid yang semakin dominan dalam beberapa tahun terakhir turut memberikan tantangan tersendiri. Mahasiswa kini menghabiskan waktu yang cukup lama di depan layar perangkat digital untuk mengikuti perkuliahan, diskusi, maupun menyelesaikan tugas. Intensitas penggunaan perangkat digital yang tinggi dapat meningkatkan risiko digital fatigue, yaitu kondisi kelelahan mental akibat paparan layar dalam waktu lama. Digital fatigue ditandai dengan gejala seperti mata lelah, sakit kepala ringan, kesulitan berkonsentrasi, serta rasa kantuk yang muncul meskipun waktu belajar belum terlalu lama. Kondisi ini dapat menyebabkan menurunnya daya serap informasi dan berkurangnya efektivitas Pembelajaran (Lingkungan Sekolah Bisnis et al., 2025).

Selain faktor teknologi, lingkungan belajar juga berperan penting dalam menentukan produktivitas mahasiswa. Gangguan dari lingkungan sekitar, seperti kebisingan, notifikasi media sosial, atau kurangnya ruang belajar yang kondusif, dapat mengurangi fokus dan meningkatkan distraksi. Dalam pembelajaran daring,

pengawasan yang lebih longgar dibandingkan pembelajaran tatap muka juga dapat memicu penurunan disiplin belajar.

Produktivitas yang menurun sering kali ditandai dengan gejala fisik dan perilaku tertentu. Beberapa indikator umum meliputi:

- Sering menguap atau menggosok mata.
- Menundukkan kepala dalam waktu lama.
- Tatapan kosong atau tidak terarah.
- Frekuensi kedipan mata yang meningkat.
- Respon yang lambat terhadap pertanyaan atau instruksi.
- Penurunan interaksi verbal maupun non-verbal selama perkuliahan.

Indikator-indikator tersebut sebenarnya dapat diamati melalui ekspresi wajah dan gerakan mikro (micro-movement). Perubahan kecil pada ekspresi wajah sering kali menjadi sinyal awal terjadinya penurunan fokus atau munculnya kelelahan. Namun, pengamatan manual oleh dosen atau pengajar memiliki keterbatasan, terutama dalam kelas dengan jumlah mahasiswa yang banyak atau dalam sistem pembelajaran daring yang mengandalkan tampilan video kecil di layar.

## 2.5 Python dan Library Pendukung

Python merupakan bahasa pemrograman tingkat tinggi (high-level programming language) yang pertama kali dikembangkan oleh Guido van Rossum dan dirilis pada tahun 1991. Bahasa ini dirancang dengan filosofi yang menekankan keterbacaan kode (code readability) dan kesederhanaan sintaksis, sehingga mudah dipahami baik oleh pemula maupun pengembang berpengalaman. Struktur

penulisan kode Python yang ringkas dan jelas memungkinkan proses pengembangan perangkat lunak menjadi lebih efisien dan minim kesalahan sintaks.

Sebagai bahasa yang bersifat open-source, Python didukung oleh komunitas global yang sangat besar dan aktif. Dukungan komunitas ini berkontribusi pada perkembangan ekosistem pustaka (library) yang luas dan terus diperbarui. Python juga bersifat dinamis, multiguna, serta kompatibel di berbagai sistem operasi, sehingga banyak digunakan dalam beragam bidang seperti pengembangan web, otomatisasi sistem, analisis data, kecerdasan buatan (Artificial Intelligence), Machine Learning, hingga Deep Learning (Inggit Agustina & Raditya Danar Dana, 2024).

Popularitas Python dalam dunia penelitian dan industri tidak terlepas dari kelengkapan pustaka standar yang dimilikinya. Bahasa ini menyediakan modul bawaan yang komprehensif untuk pengolahan data, manipulasi file, komputasi numerik, hingga visualisasi. Selain itu, Python mendukung integrasi dengan berbagai bahasa pemrograman lain seperti C dan C++, sehingga mampu menangani kebutuhan komputasi yang kompleks dengan performa yang tetap optimal (Ljubomir Perkovic, 2011).

Dalam praktik pengembangan, Python dapat ditulis dan dijalankan melalui berbagai Integrated Development Environment (IDE), seperti Visual Studio Code (VS Code), PyCharm, dan Sublime Text. Selain itu, tersedia pula lingkungan berbasis cloud seperti Jupyter Notebook dan Google Colab yang memungkinkan eksekusi kode secara langsung melalui peramban web tanpa instalasi lokal.

Platform ini sangat mendukung penelitian berbasis data karena memudahkan dokumentasi, visualisasi, serta kolaborasi secara daring.

Python dianggap sebagai bahasa pemrograman yang paling banyak digunakan dalam bidang Machine Learning dan Deep Learning. Keunggulan utamanya terletak pada kemudahan implementasi algoritma, fleksibilitas dalam eksplorasi data, serta ketersediaan berbagai library yang mendukung pengembangan model prediktif. Kombinasi antara sintaks yang sederhana, dokumentasi yang lengkap, dan dukungan komunitas yang luas menjadikan Python sebagai standar de facto dalam pengembangan sistem kecerdasan buatan (Riziq sirfatullah Alfarizi et al., 2023).

Dalam penelitian ini, Python digunakan sebagai bahasa utama untuk membangun sistem klasifikasi kondisi belajar mahasiswa berbasis citra wajah. Beberapa library utama yang digunakan antara lain:

1. OpenCV (Open Source Computer Vision Library)

   OpenCV merupakan pustaka yang dirancang khusus untuk pengolahan citra dan visi komputer (computer vision). Library ini menyediakan berbagai fungsi siap pakai untuk mendeteksi wajah (face detection), melakukan konversi warna (RGB ke grayscale), melakukan cropping, normalisasi ukuran gambar, serta pengurangan noise. OpenCV sangat efisien dalam memproses citra secara real-time dan banyak digunakan dalam pengembangan sistem berbasis pengenalan wajah. Dalam penelitian ini,

OpenCV berperan penting pada tahap pra-pemrosesan citra sebelum data dimasukkan ke model klasifikasi (Zulkhaidi et al., 2020).

2. Scikit-learn

   Scikit-learn adalah library Python yang menyediakan berbagai algoritma machine learning, termasuk Random Forest, Support Vector Machine (SVM), K-Nearest Neighbor (KNN), regresi, serta teknik evaluasi model. Library ini dirancang dengan antarmuka yang sederhana namun kuat, sehingga memudahkan proses pelatihan (training), pengujian (testing), serta validasi model. Dalam penelitian ini, Scikit-learn digunakan untuk membangun model Random Forest, melakukan pembagian data latih dan data uji, serta menghitung metrik evaluasi seperti confusion matrix, akurasi, presisi, dan recall (Nasien et al., 2024).

3. NumPy dan Pandas

   NumPy dan pandas merupakan pustaka yang digunakan untuk pengolahan data numerik dan manajemen dataset. NumPy menyediakan struktur data array multidimensi dan berbagai operasi matematika yang efisien, sedangkan Pandas digunakan untuk memanipulasi dan mengelola dataset dalam bentuk tabel (dataframe). Kedua library ini sangat membantu dalam proses persiapan data sebelum dimasukkan ke dalam model klasifikasi.

Secara keseluruhan, kombinasi Python dengan berbagai library pendukung tersebut memungkinkan pengembangan sistem yang terstruktur, efisien, dan mudah direplikasi. Ekosistem Python yang kuat memberikan fleksibilitas dalam pengembangan lanjutan, seperti integrasi dengan antarmuka grafis (Graphical User

Interface), aplikasi berbasis web, maupun sistem monitoring real-time. Dengan demikian, penggunaan Python dalam penelitian ini tidak hanya relevan dari sisi teknis, tetapi juga mendukung keberlanjutan dan pengembangan sistem di masa mendatang.