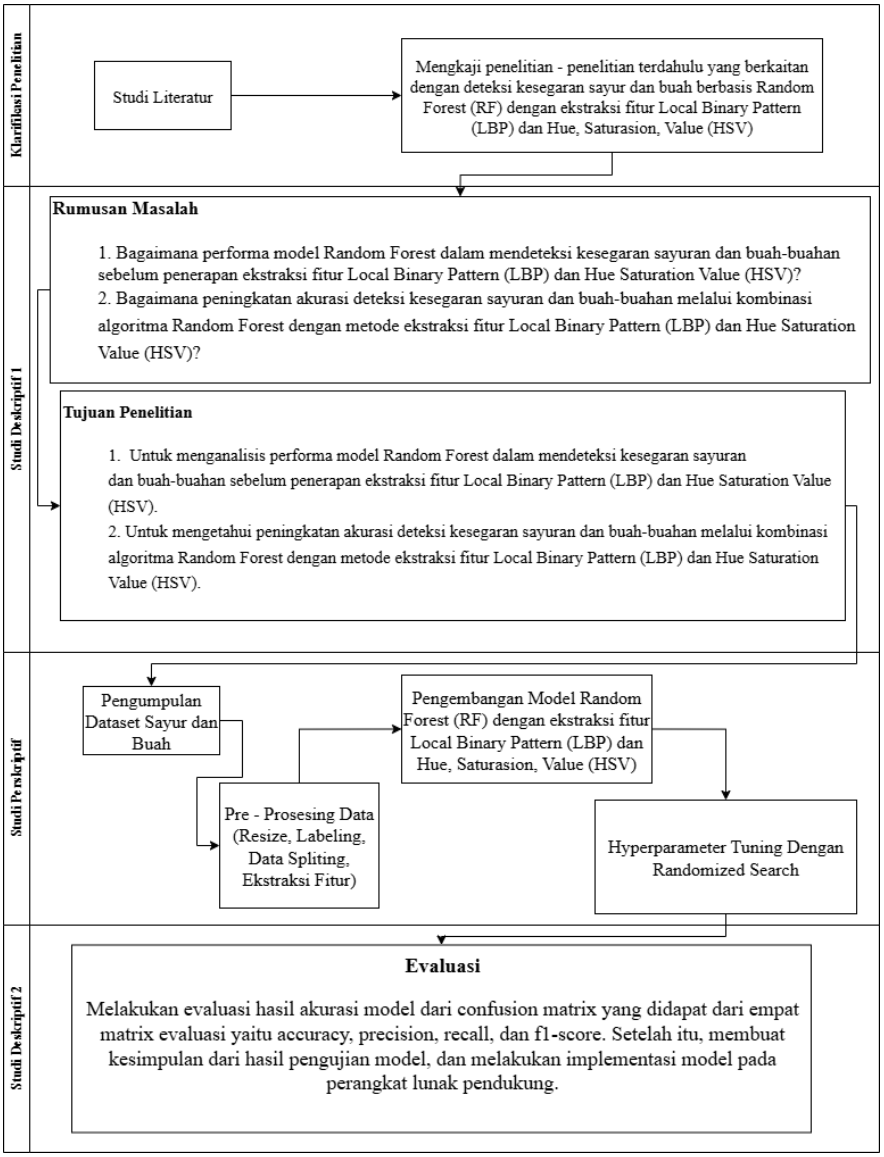
BAB III
METODOLOGI PENELITIAN

3.1 Desain Penelitian

Penelitian ini menggunakan metode *Design Research Methodology* (DRM), sebuah kerangka kerja sistematis untuk mengembangkan dan mengevaluasi solusi berbasis desain (Blessing & Chakrabarti, 2009). Metode *Design Research Methodology* (DRM) dipilih karena pendekatannya yang iteratif dan berbasis data sangat relevan untuk tujuan penelitian ini, yakni menguji serta meningkatkan efektivitas model *machine learning* (Rijal dkk., 2024)



Gambar 3.1 Alur *Desain Research Methodology* (DRM)

Penelitian ini dilakukan dengan metode penelitian *Desain Research Methodology* (DRM). Metode ini mempunyai empat tahapan utama diantaranya klarifikasi penelitian, studi deskriptif 1, studi perskriptif, sampai studi deskriptif 2. Untuk menjelasakan lebih lanjut berikut ini adalah penjabaran dari empat tahapan dari *Desain Research Methodology* (DRM) diatas.

3.1.1 Klarifikasi Penelitian

Pada tahap ini, merupakan tahapan untuk melakukan kajian literatur dari penelitian-penelitian sebelumnya yang memiliki keterkaitan tehadap *sub*-bidang *mechine learning* yaitu *object classification* dengan penggunaan untuk klasifikasi terhadap sayuran dan buah. Serta tujuan untuk melakukan studi literatur ini adalah menjadi batasan agar penelitian yang dilakukan masih sejalan dengan peneliti-peneliti sebelumnya (Dwi Lestari & Merthayasa, 2023), agar dalam pelaksaannya teknik yang digunakan bisa efektif dan memberikan pembaruan dari penelitian sebelumnya.

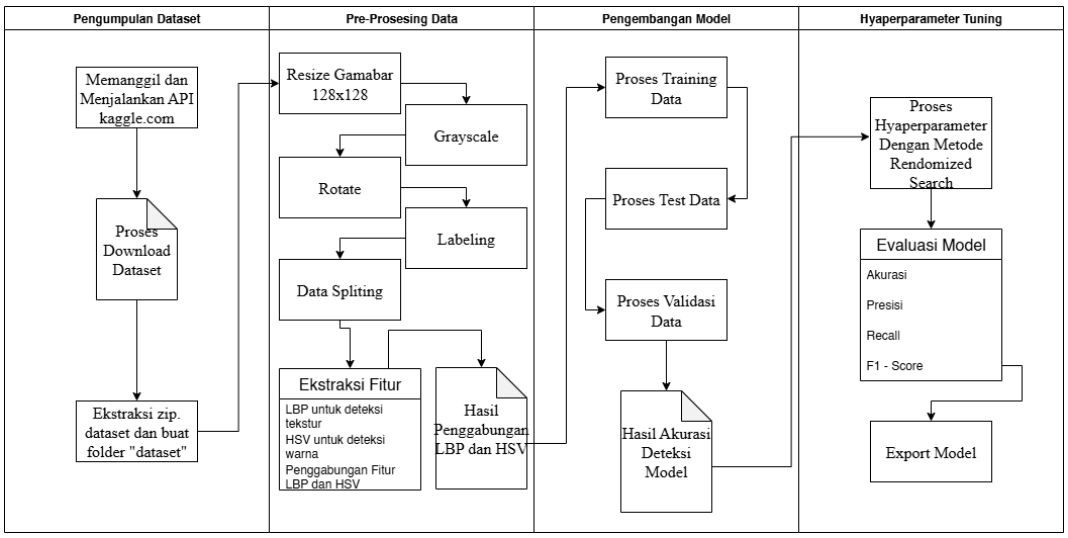
3.1.2 Studi Deskriptif 1

Tahapan ini bertujuan untuk memetakan permasalahan dan menetapkan tujuan setelah melakukan studi literatur yang mendalam pada tahap klarifikasi penelitian (Pebriani dkk., 2020). Dengan melakukan ini, penelitian akan tetap berada dalam jalur yang sesuai dengan konteks dan kondisi penelitian yang telah ada. Identifikasi permasalahan dan tujuan dirumuskan berdasarkan hasil kajian literatur yang telah dilakukan pada tahap klarifikasi. Selanjutnya, tahap studi deskriptif 1 dilakukan untuk menetapkan indikator keberhasilan penelitian. Tahap ini berfungsi untuk memberikan pemahaman lebih mendalam mengenai konteks penelitian, termasuk variabel-variabel yang akan digunakan dalam proses pengujian dan evaluasi (Dewi, 2018).

3.1.3 Studi Perskriptif

Mengambil penjelasan tentang studi perskriptif dari Gambar 3.1 diatas, kita akan mengelompokannya menjadi tiga bagian untuk memudahkan kita dalam memahami isi dari proses studi perskriptif ini: Pengumpulan dataset, pra-

pemrosesan model, dan pengembangan model. Selanjutnya penjelasan pengembangan model secara menyeluruh akan dijelaskan pada gambar berikut:



Gambar 3.2 Alur Proses pada Tahapan Studi Perskriptif

3.1.3.1 Pengumpulan Data

Tahapan pertama pada penelitian ini adalah pengumpulan data citra sayuran dan buah. Data sekunder diperoleh dari Kaggle dengan cara menghubungkan API Kaggle ke dalam Google Colab. Setelah berhasil mengunduh file ZIP, langkah selanjutnya adalah melakukan ekstraki menggunakan modul zipfile atau perintah !unzip ke dalam folder kerja, sehingga di dalam direktori dataset muncul *sub*-folder berisi kumpulan gambar sayuran dan buah. Masing-masing *sub*-folder merepresentasikan kelas label, dan setiap gambar di dalamnya siap diproses lebih lanjut untuk pelatihan model. Dataset ini bersifat *open source*, sehingga mudah diakses dan direproduksi oleh peneliti lain.

3.1.3.2 Pre-pemrosesan Data

Pada penelitian ini, dataset yang digunakan terdiri dari 14.700 gambar yang dibagi menjadi enam data objek diantaranya apel, pisang, pare, paprika hijau, jeruk, dan tomat yang dibuat dengan versi segar dan busuk jadi total nya menjadi 12 folder. Pada proses ini dilakukan dalam beberapa tahapan utama diantaranya sebagai berikut:

a) Resize

Proses pre-pemrosesan dimulai dengan membaca setiap citra menggunakan *OpenCV* dan mengubah ukurannya menjadi 128×128 piksel agar semua gambar memiliki dimensi seragam. Langkah resize memastikan bahwa model tidak dipaksa menangani variasi ukuran input, sehingga arsitektur dan bobotnya bisa lebih stabil (Nugroho dkk., 2020). Proses tersebut memudahkan algoritma pembelajaran mesin mengakses dan memproses data secara konsisten.

i) Membaca gambar dengan cv2.imread() untuk memuat file citra ke dalam array numpy.
ii) Resize gambar agar semua input memiliki ukuran 128x128 pixsel.

b) Grayscale

Setelah dilakukan proses resize maka nanti gambar akan dilakukan proses konversi ruang warna menggunakan opencv kembali dengan parameter khusus yang disesuaikan agar gambar bisa diubah kedalam format grayscale.

i) Merubah format dengan cv2.cvtColor() untuk mengubah warna gambar.
ii) cv2.COLOR_BGR2GRAY perintah spesifik untuk mengubah format RGB menjadi grayscale.

c) Rotate

Setelah gambar menjadi grayscale, tahap selanjutnya adalah rotasi. Gambar tersebut akan diputar searah atau berlawanan arah jarum jam dengan sudut tertentu (misalnya 45°, 90°, atau sudut acak). Dimana proses ini bertujuan untuk:

i) Menambah variasi dan memperkaya dataset dengan data baru.
ii) Meningkatkan ketangguhan model agar mampu mengenali objek dalam berbagai orientasi.

d) Labeling

Setelah data dilakukan resize, langkah berikutnya adalah merubah lebel yang bersifat kategorial harus dilakukan konversi menjadi bentuk numerik. Dimana

dalam dataset ini yang memerlukan transformasi adalah label kesegaran dari sayur dan buah, yaitu:

i) Fresh sama dengan 0
ii) Stale sama dengan 1

e) Data Spliting

Setelah seluruh citra ter-preproses dengan benar, langkah berikutnya adalah membagi dataset menjadi tiga bagian utama dengan proporsi 80%–10%–10% (Faoziatun Khusna dkk., t.t.), yaitu:

i) 80% Data latih untuk digunakan untuk melatih model pembelajaran mesin.
ii) 10% Data uji digunakan untuk mengevaluasi performa akhir model pada data yang sama sekali belum pernah dilihat.
iii) 10% Data validasi igunakan untuk memilih dan menyetel hyperparameter, serta memantau potensi overfitting selama proses pelatihan.

Pemisahan ini dilakukan menggunakan Stratified Sampling, sehingga proporsi kelas fresh dan stale pada masing-masing subset tetap seimbang.

f) Ekstraksi Fitur

Ekstraksi fitur pada buah dan sayuran dilakukan dengan dua metode komplementer: *Local Binary Pattern* (LBP) untuk menangkap pola tekstur permukaan dan ruang warna HSV untuk mendeteksi perubahan warna seiring kematangan atau kerusakan. Pertama, *Local Binary Pattern* (LBP) membandingkan tiap piksel dengan tetangga sekitarnya untuk membentuk pola biner yang merangkum kontur mikro buah segar cenderung memiliki permukaan halus dan seragam, sedangkan buah yang mulai menua atau membusuk menunjukkan bercak dan retakan yang menghasilkan pola *Local Binary Pattern* (LBP) berbeda; histogram *Local Binary Pattern* (LBP) yang dinormalisasi kemudian memberikan sinyal tekstur yang kaya kepada model.

Kedua, dengan mengonversi citra ke ruang *Hue Saturation Value* (HSV), kita memisahkan komponen hue (tonalitas warna), saturation (kejenuhan), dan value

(intensitas), sehingga lebih tahan terhadap variasi pencahayaan; seiring buah berubah warna misalnya dari hijau segar menjadi kuning pucat atau kecokelatan rata-rata tiap channel *Hue Saturation Value* (HSV) akan bergeser, yang dapat diukur sebagai indikator kesegaran. Kedua vektor fitur ini digabung menjadi satu vektor akhir, memungkinkan algoritma seperti *Random Forest* memanfaatkan informasi tekstur dan warna secara bersamaan untuk meningkatkan akurasi deteksi tingkat kesegaran buah.

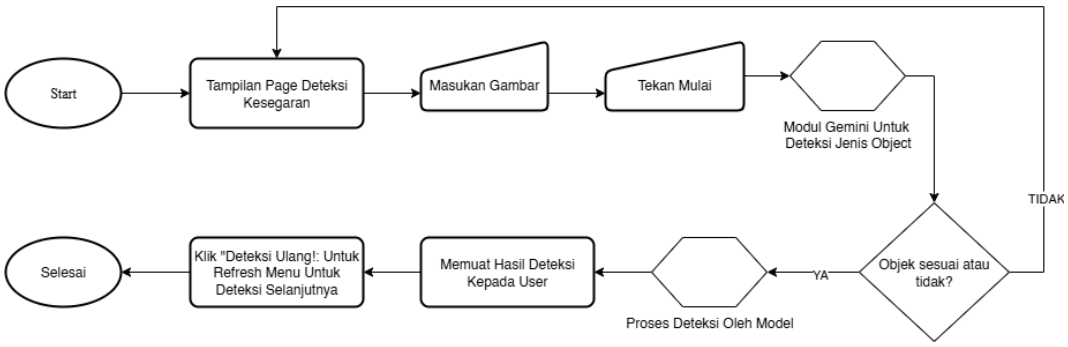
3.1.3.3 Pengembangan Model

Pada tahap pengembangan model, data yang telah diproses dibagi menjadi tiga bagian, yaitu data latih (*training*), data validasi, dan data uji (*testing*) dengan rasio 80:10:10. Data latih digunakan untuk membangun dan melatih model awal. Setelah model dilatih, data validasi digunakan untuk mengevaluasi performa model secara berkala dan menyesuaikan parameter agar model tidak *overfitting*. Proses *tuning* ini dikenal sebagai *hyperparameter tuning*, di mana kombinasi parameter terbaik dicari untuk meningkatkan kinerja model. Setelah parameter optimal ditemukan, model dilatih ulang kembali dan dilakukan validasi serta test ulang.

3.1.3.4 Implementasi Model Pada Website

Implementasi aplikasi web deteksi kesegaran ini dirancang menggunakan arsitektur *full-stack modern* yang memisahkan secara jelas antara logika antarmuka pengguna di sisi klien dan pemrosesan data di sisi server, memberikan keunggulan dalam hal skalabilitas dan kemudahan pemeliharaan. *Frontend*, sebagai titik interaksi utama bagi pengguna, dibangun sebagai *Single-Page Application* (SPA) yang dinamis menggunakan *library React* dan *toolchain Vite*, memungkinkan pembuatan komponen antarmuka yang interaktif dan dapat digunakan kembali secara efisien. Untuk menciptakan desain yang konsisten dan responsif, *framework Tailwind* CSS dengan pendekatan *utility-first* diimplementasikan untuk mempercepat proses styling. Di sisi *backend*, sebuah *Application Programming Interface* (API) server yang terfokus dan ringan dibangun menggunakan Flask, sebuah *micro-framework* Python yang ideal untuk menangani permintaan spesifik dari *frontend* dengan gabungan gemini *artificial intelligence*. Dimana gemini

artificial intelligence akan digunakan untuk pemberi batasan dalam deteksi object. Alur kerja website mulai dari *input* gambar sampai mendapatkan hasil deteksi dijabarkan secara sistematis pada Gambar 3. 3.



Gambar 3.3 Diagram Alur Kerja Website

Alur kerja deteksi dimulai saat pengguna mengunggah file gambar melalui antarmuka *React*. Gambar tersebut kemudian dikirimkan secara asinkron menggunakan Axios ke *endpoint* / *predict* di server Flask. Setibanya di *backend*, server menjalankan serangkaian pre-pemrosesan untuk mengekstrak fitur-fitur. Proses ini mencakup ekstraksi *Local Binary Pattern* (LBP) untuk menganalisis karakteristik tekstur yang menandakan kondisi fisik sayuran, serta pembuatan histogram warna *Hue Saturation Value* (HSV) untuk menangkap informasi warna yang lebih robust terhadap variasi pencahayaan. Fitur-fitur gabungan yang kaya informasi ini kemudian dimasukkan ke dalam inti dari sistem, yaitu model *machine learning Random Forest* (RF). Model ini, yang telah dilatih secara *offline* menggunakan Scikit-learn dan disimpan dalam format .pkl melalui Joblib, dipilih secara spesifik karena keseimbangannya antara akurasi yang tinggi dan kebutuhan komputasi yang rendah. Dengan memuat model yang sudah terlatih ini saat dimasukan kedalam *backend*, server Flask dapat melakukan inferensi secara instan, menghasilkan prediksi yang kemudian dikemas dalam format JSON terstruktur dan dikirim kembali ke *frontend*, di mana React akan menampilkannya secara dinamis kepada pengguna.

3.1.4 Studi Deskriptif 2

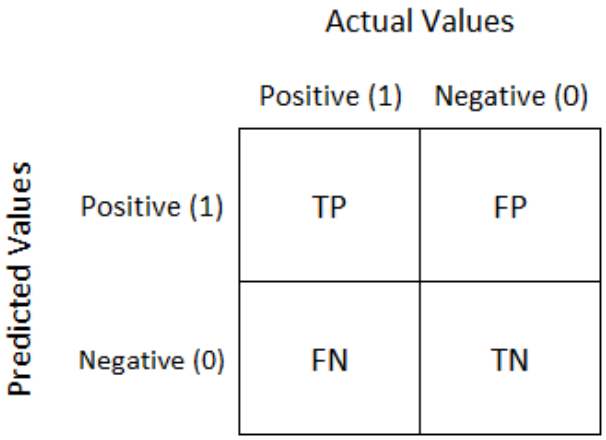
Ini adalah proses terakhir dari tahapan *Desain Research Methodology* (DRM), yang mana isinya membahas hasil pengujian dari model algoritma yang telah dibuat akan dievaluasi menggunakan berbagai metrik evaluasi yang telah ditetapkan (Fadilla dkk., 2020). Nanti setelah dilaksanakannya proses *training, test,* dan *validation* maka akan dihitung akumulasi nilai keseluruhan untuk mengetahui berapa persen tingkat akurasi dari algoritma *Random Forest* (*RF*) dengan ekstraksi fitur *Local Binary Pattern* (LBP) dan *Hue Saturation Value* (HSV). Berdasarkan hasil pengujian, kesimpulan, dan hipotesis akan disusun untuk memberikan pemahaman yang jelas tentang hasil penelitian yang telah dilakukan. Proses ini memungkinkan peneliti untuk menilai efektivitas model dalam mendeteksi kesegaran sayuran dan buah – buahan untuk menentukan berapa persen akurasi yang didapat. Model dievaluasi dengan empat matrix evaluasi yaitu *accuracy, precision, recall, dan F1-score* jika performa belum optimal maka dilakukan hyperparameter tuning dengan randomized search untuk menemukan parameter terbaik dari lima parameter *n_estimators, max_depth, min_samples_split, min_samples_leaf, max_features*.

3.2 *Dataset* Sayuran

Dataset sayuran yang digunakan dalam penelitian ini berasal dari situs Kaggle dengan judul " *Fresh and Stale Images of Fruits and Vegetables*" Dataset ini berisi sekitar 14.700 gambar, yang dibuat pada tahun 2024 dengan dataset yang digunakan diambil dari update terbaru ditahun 2025. Dataset mencakup enam jenis sayuran dan buah berbeda, dan gambar-gambar tersebut dikumpulkan oleh pembuat dataset untuk tujuan klasifikasi kesegaran sayuran dan buah sebab itu didalamnya dibagi menjadi 12 folder terpisah, yaitu fresh dan stale pada setiap jenis sayuran dan buah. Setiap folder memiliki peran penting dalam proses pengembangan model, di mana folder itu digunakan untuk labeling dan pembeda antara jenis dan kelayakan dari sayuran dan buah pada model, Gambar-gambar dalam dataset ini akan menjadi referensi utama dalam proses klasifikasi dan penentuan kelayakan sayuran dan sayuran yang dilakukan oleh peneliti.

3.3 Instrumen Penelitian

Terdapat sekitar empat instrumen yang menunjang penelitian ini, yaitu *accuracy, precision, recall, dan F1-score* sebagai matrix evaluasi dari model yang dibuat. Dimana ke empat instrumen ini digunakan oleh peneliti untuk memberikan hasil akurat jumlah persenan dari kinerja algoritma *Random Forest* (*RF*) dengan ekstraksi fitur *Local Binary Pattern* (LBP) dan *Hue Saturation Value* (HSV), dalam melakukan deteksi kesegaran sayur dan buah.



Gambar 3.4 Logika *Confusion Matrix*

3.3.1 *Accuracy*

Accuracy mengukur proporsi dari prediksi yang benar terhadap keseluruhan prediksi. Rumusnya adalah:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (10)$$

Dimana:

TP (*True Positives*) = Jumlah kasus positif yang terdeteksi dengan benar

TN (*True Negatives*) = Jumlah kasus negatif yang terdeteksi dengan benar

FP (*False Positives*) = Jumlah kasus negatif yang salah terdeteksi sebagai positif

FN (*False Negatives*) = Jumlah kasus positif yang terdeteksi sebagai negatif

3.3.2 *Precision*

Precision mengukur seberapa banyak dari hasil deteksi yang benar-benar positif. Rumusnya adalah:

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

Dimana:

TP (*True Positives*) = Jumlah kasus positif yang terdeteksi dengan benar

FP (*False Positives*) = Jumlah kasus negatif yang salah terdeteksi sebagai positif

3.3.3 *Recall*

Recall mengukur seberapa baik model mendeteksi semua kasus positif yang sebenarnya. Rumusnya adalah:

$$Recall\ = \frac{TP}{TP + FN} \tag{12}$$

Dimana:

TP (*True Positives*) = Jumlah kasus positif yang terdeteksi dengan benar

FN (*False Negatives*) = Jumlah kasus positif yang terdeteksi sebagai negatif

3.3.4 *F1-Score*

F1-Score adalah rata-rata harmonis dari *precision* dan *recall*. Nilai ini memberikan gambaran seimbang antara *precision* dan *recall*. Rumusnya adalah:

$$F1 - Score\ = 2 \times \frac{Presisi \times RecaLL}{Presisi + RecaLL} \tag{13}$$

F1-Score berguna terutama ketika terdapat ketidakseimbangan antara jumlah kasus positif dan negatif. Nilai *F1-Score* yang lebih tinggi menunjukkan keseimbangan antara *precision* dan *recall* yang baik.

3.4 Alat dan Bahan Penelitian

Pada penelitian ini perangkat atau alat digunakan untuk keberlangsungan dalam penelitian dan pengembangan model deteksi sayuran dan buah yang akan dibuat adalah sebagai berikut:

Tabel 3.1 Spesifikasi Perangkat yang Digunakan

Komponen	Spesifikasi
Laptop	
Sistem operasi	Windows 10
***Random Access Memory* (RAM)**	16 GB
Processor	Intel Core i7 generasi sepuluh
Perangkat penyimpanan	SSD dengan kapasitas 512 GB
***Graphic Processing Unit* (GPU)**	NVIDIA GPU
Cloud Environment	
Platform	Google Colab
Prosesor	Intel Xeon
***Graphic Processing Unit* (GPU)**	NVIDIA Tesla T4
***Random Access Memory* (RAM)**	13 GB

Adapun perangakat lunak beserta *library* yang digunakan untuk menunjang pengembangan model pada penelitian ini dicantumkan pada Tabel 3.2

Tabel 3.2 Perangkat lunak dan *library* yang digunakan

Nama	Jenis	Deskripsi
Python	Bahasa Pemrograman	Digunakan sebagai bahasa pemrograman utama untuk pengembangan *Machine Learning* (ML) dan *computer vision*.
Google Colab	IDE	Digunakan untuk eksperimen interaktif, *prototyping* model

		Machine Learning (ML), dan dokumentasi proses penelitian.
OS	Modul	Digunakan untuk berinteraksi dengan sistem operasi, seperti manipulasi path dan file.
Shutil	Modul	Digunakan untuk operasi file tingkat lanjut seperti menyalin dan memindahkan file atau direktori.
OpenCV	*Library*	Berfungsi untuk memuat dan memproses gambar, seperti mengubah ruang warna dan mengekstraksi fitur histogram *Hue Saturation Value* (HSV), agar data visual tersebut siap diolah oleh model *Random Forest* (RF).
Numpy	*Library* Numerik	Digunakan untuk operasi *array* dan perhitungan numerik tingkat tinggi.
Matplotlib.pyplot	*Library* Visualisasi	Digunakan untuk membuat grafik dan visualisasi data secara statis.
Scikit-learn	*Library*	Digunakan untuk implementasi model klasifikasi, seperti SVM,

		Random Forest (RF), dan evaluasi model dengan metrik seperti *accuracy, precision, recall, dan F1-score*.
Pandas	*Library*	Digunakan untuk pengelolaan dan analisis *dataset*, termasuk pengolahan hasil evaluasi model.
Matplotlib, Seaborn	*Library*	Digunakan untuk visualisasi hasil evaluasi model seperti confusion matrix, grafik evaluasi, dan visualisasi fitur *Hue Saturation Value* (HSV)
Learning_curve	Fungsi dari *sklearn*	Digunakan untuk menampilkan grafik kurva pembelajaran model *machine learning*.
TensorFlow/PyTorch (Opsional)	*Library*	Digunakan untuk eksperimen lebih lanjut dengan model pembelajaran mendalam (*Deep Learning*) jika diperlukan.
Local_binary_pattern	Fungsi dari *skimage*	Digunakan untuk mengekstraksi fitur tekstur dari gambar.

Train_test_split	Fungsi dari *sklearn*	Digunakan untuk membagi dataset menjadi data latih dan data uji.
GridSearchCV	Fungsi dari *sklearn*	Digunakan untuk mencari kombinasi parameter terbaik melalui pencarian grid.
Random Forest Classifier	Algoritma ML	Digunakan untuk klasifikasi dengan membangun banyak pohon keputusan.
Accuracy_score	Fungsi Evaluasi	Digunakan untuk menghitung akurasi prediksi model.
Classification_report	Fungsi Evaluasi	Memberikan laporan lengkap metrik evaluasi seperti *accuracy, precision, recall, dan F1-score*.
Confusion_matrix	Fungsi Evaluasi	Digunakan untuk menampilkan matriks kesalahan dari hasil prediksi.
Joblib	*Library* Utilitas	Digunakan untuk menyimpan dan memuat model *machine learning* dalam bentuk file .pkl.
VS *Code*	IDE/*Text Editor*	Alat untuk menulis, mengedit, dan menjalankan code. Mendukung *debugging* dan

		pengembangan skala penuh.
Kaggle API	Data *Access Tool*	Digunakan untuk mengunduh dataset gambar sayuran dari Kaggle, memungkinkan akses mudah ke data penelitian.

3.5 Analisis Data

Penelitian ini diawali dengan praproses citra yang mencakup validasi gambar, perubahan ukuran menjadi 128×128 piksel, dan konversi citra ke dalam bentuk vektor numerik menggunakan metode flatten. Ekstraksi fitur dilakukan dengan menggabungkan *Local Binary Pattern* (LBP) untuk menangkap tekstur permukaan dan *Hue, Saturation, Value* (HSV) untuk mendeteksi perubahan warna pada buah dan sayur. Dataset dibagi menjadi tiga bagian: 80% untuk pelatihan, 10% validasi, dan 10% pengujian agar distribusi kelas seimbang.

Model klasifikasi menggunakan *Random Forest* (RF), yang membangun banyak pohon keputusan melalui *Bootstrap* Sampling dan pemilihan fitur secara acak guna meningkatkan akurasi dan mengurangi *overfitting*. Prediksi akhir ditentukan melalui majority voting dari seluruh pohon. Evaluasi model dilakukan menggunakan metrik *accuracy, precision, recall, dan F1-score*. Jika diperlukan, *Hyperparameter Tuning* dengan *Grid Search* dilakukan untuk mengoptimalkan parameter seperti *n_estimators, max_depth, dan max_features*. Proses ini bertujuan untuk menilai efektivitas *Random Forest* (RF) dalam mengklasifikasikan tingkat kesegaran buah dan sayur berdasarkan ciri visualnya.