

BAB III

ANALISIS DAN PERANCANGAN

3.1 Analisis

3.1.1 Identifikasi Masalah

Saat ini, pemilihan makanan bagi individu yang memiliki alergi sering kali dilakukan secara manual dan hanya mengandalkan informasi yang terbatas. Banyak orang memilih makanan hanya untuk mengatasi rasa lapar tanpa mempertimbangkan potensi kandungan alergen yang dapat memicu reaksi alergi serius. Padahal, reaksi alergi terhadap makanan tertentu bisa berakibat fatal bagi mereka yang sensitif. Kurangnya informasi visual mengenai bahan penyusun makanan, terutama pada makanan yang diolah atau dijual tanpa label, menjadi salah satu kendala utama dalam memastikan keamanan bagi individu dengan alergi.

Di sisi lain, meskipun teknologi kecerdasan buatan semakin berkembang, belum banyak aplikasi yang memanfaatkan metode otomatis seperti *Convolutional Neural Network* (CNN) untuk mendeteksi potensi alergen dalam makanan. Teknologi (CNN) memiliki potensi untuk menganalisis makanan secara visual dan mengidentifikasi kemungkinan kandungan alergen dengan lebih objektif dan akurat. Aplikasi ini diharapkan dapat memberikan pilihan yang lebih aman dan praktis bagi pengguna dengan alergi, sehingga mereka dapat membuat keputusan yang lebih informasional dan mengurangi risiko alergi.

Tabel 3.1 SWOT

<i>Strengths</i>	<i>Weaknesses</i>
Penggunaan (CNN) memungkinkan deteksi potensi alergen secara otomatis dan akurat, memberikan keamanan lebih bagi pengguna.	Implementasi (CNN) membutuhkan data pelatihan yang besar dan spesifik, serta keterampilan teknis dalam <i>image processing</i> dan <i>Deep Learning</i> .
<i>Opportunities</i>	<i>Threats</i>

Teknologi AI dalam pendeteksian makanan memiliki potensi besar untuk diterapkan secara luas, terutama dengan meningkatnya kesadaran akan keamanan makanan bagi penderita alergi.	Ketersediaan data gambar makanan yang cukup relevan bisa menjadi tantangan, serta resistensi masyarakat terhadap teknologi baru untuk identifikasi alergen.
--	---

3.1.2 Pemecahan Masalah

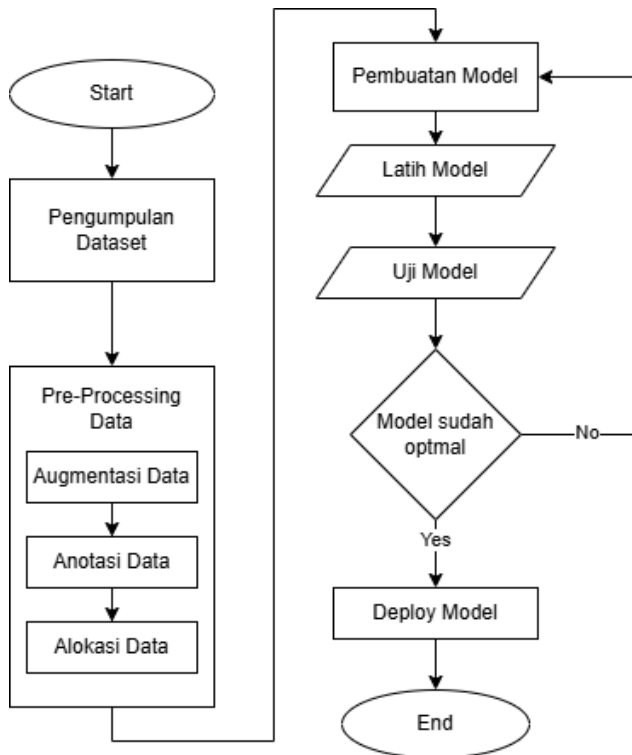
Pemecahan masalah terhadap tantangan dalam pemilihan makanan yang aman bagi individu dengan alergi dapat diatasi dengan pengembangan aplikasi pendeteksi potensi alergen dalam makanan menggunakan metode *Convolutional Neural Network* (CNN). Aplikasi ini akan memanfaatkan teknologi *image processing* dan machine learning untuk mengidentifikasi kandungan makanan secara otomatis dan akurat melalui analisis visual, sehingga memberikan informasi yang lebih objektif dan membantu pengguna menghindari alergen yang dapat memicu reaksi alergi.

Selain itu, untuk mengatasi potensi resistensi pengguna terhadap penerapan teknologi AI dalam identifikasi allergen makanan, diperlukan pendekatan komunikasi dan edukasi kepada masyarakat. Edukasi yang tepat dapat membangun pemahaman dan kepercayaan pengguna terhadap aplikasi ini, yang tidak hanya meningkatkan kesadaran akan keamanan makanan tetapi juga memberikan solusi praktis bagi mereka yang memiliki alergi.

3.2 Perancangan

3.2.1 Perancangan Sistem

Pada perancangan sistem klasifikasi menggunakan *Convolutional Neural Network* (CNN) Sebagai berikut:



Gambar 3.1 Diagram Alir (CNN)

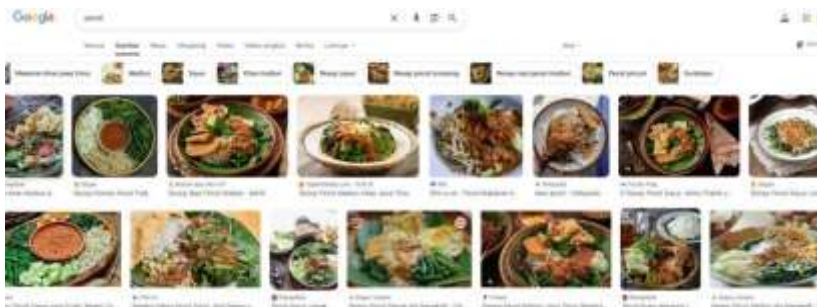
Dalam perancangan sistem (CNN) pada penelitian ini, terdapat beberapa tahapan utama: pengumpulan dataset, pemrosesan awal (pre-processing) data, pembuatan model, pelatihan, pengujian, dan akhirnya, penerapan model. Alur perancangan dimulai dengan pengumpulan dataset, kemudian data tersebut akan di augmentasi agar dataset yang digunakan dalam pelatihan (*training*) lebih bervariasi serta diubah ukurannya sesuai dengan model pra-latih (*pre-trained*) yang akan diterapkan. Setelah itu, data yang telah dikumpulkan dilabeli sesuai kategori.

Tahap berikutnya adalah pembagian data menjadi bagian pelatihan (*training*) dan validasi (*validation*). Selanjutnya, dilakukan tahap pemodelan,

yaitu merancang arsitektur yang akan diterapkan dalam proses pelatihan, dimulai dengan memuat model dari pustaka TensorFlow dan Keras, menambahkan beberapa lapisan tersembunyi (*hidden layers*), serta mengatur hyperparameter yang akan digunakan dalam penelitian ini. Pelatihan model kemudian dilakukan menggunakan model pra-latih *Mobilenetv2* hingga mencapai akurasi yang optimal. Setelah itu, model diuji dengan data uji (*test*) untuk mengukur performanya. Jika hasilnya belum memuaskan, pemodelan ulang akan dilakukan. Apabila hasilnya sudah optimal, model akan dikonversi ke format tflite agar dapat disimpan di Firebase.

3.2.1.1 Pengumpulan Dataset

Pengumpulan dataset adalah kegiatan mengumpulkan informasi dari berbagai sumber untuk digunakan dalam berbagai keperluan. Dataset merupakan kumpulan data terstruktur yang disimpan dan dianalisis oleh komputer untuk menghasilkan informasi. Dalam penelitian ini, dataset yang digunakan berasal dari situs www.kaggle.com dengan nama *Food Image* dari pengguna bernama Scoot Mader dan menambahkan dengan mencari gambar melalui www.google.com dengan penyesuaian dari jenis makanan yang ada diberbagai daerah.



Gambar 3.2 Pengumpulan Dataset

3.2.1.2 Augmentasi Data

Augmentasi data merupakan proses pembuatan data baru dengan mentransformasikan citra asli. Pada penelitian ini diperlukan ada nya proses augmentasi dengan mengubah ukuran, penyesuaian kecerahan dan kontras, dan rotasi. Jadi proses pengubahan ukuran ini dilakukan dengan mengubah gambar dengan Panjang dan lebar nya 224x224 pixel dengan penyesuaian *input layer* dari arsitektur (CNN) *Mobilenetv2*.

3.2.1.3 Anotasi Data

Pada anotasi data, dilakukannya pelabelan dari jenis makanan pada gambar untuk mendeteksi klasifikasi makanan. Dalam proses ini mempermudah sistem untuk melakukan *training* dalam mengenali jenis makanan.

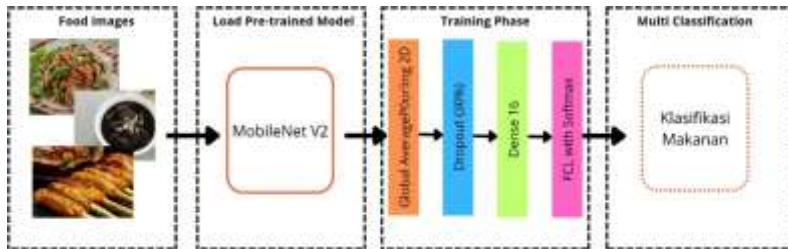
3.2.1.4 Alokasi Data

Pada alokasi data, dilakukannya persiapan dalam pengolahan dan klasifikasi data. Setiap data tentang makanan dibagi menjadi data pelatihan dan pengujian. Data pelatihan digunakan dalam mencari hasil terbaik, sedangkan data pengujian digunakan untuk menguji model yang telah dihasilkan selama proses pelatihan. Dari 100 citra dari setiap 16 jenis makanan data dibagi proporsi 70% untuk data pelatihan dan 30% untuk data pengujian.

3.2.1.5 Pembuatan Model

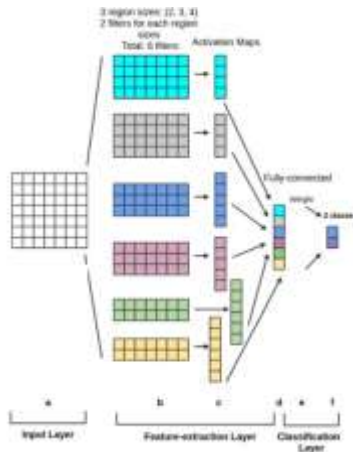
Pembuatan model adalah salah satu proses penting dalam pembelajaran mesin dan kecerdasan buatan yang melibatkan perancangan arsitektur, pelatihan, dan evaluasi model untuk memprediksi atau mengklasifikasikan data. Proses ini krusial untuk penerapan model dalam aplikasi produksi, di mana model dapat digunakan untuk melakukan prediksi secara real-time. Pada penelitian ini, model yang digunakan adalah

MobileNetV2, salah satu model pra-terlatih yang tersedia pada library TensorFlow dan Keras. Untuk memastikan model bekerja secara optimal, dilakukan berbagai langkah penyesuaian dan peningkatan selama proses pelatihan.



Gambar 3.3 Rancangan Model

Gambar di atas mengilustrasikan alur kerja arsitektur *transfer learning* yang diterapkan dalam penelitian ini. Proses diawali dengan gambar makanan (*Food Images*) sebagai masukan. Gambar tersebut kemudian diproses oleh model pra-terlatih MobileNetV2 yang berfungsi sebagai *base model* atau pengekstraksi fitur (tahap *Load Pre-trained Model*). Model ini telah dilatih pada dataset besar sehingga mampu mengenali berbagai pola visual kompleks seperti tekstur, bentuk, dan warna dari sebuah gambar.



Gambar 3.4 Rancangan Model CNN

Untuk memahami cara kerja mengekstraksi fitur, Gambar 3.4 mengilustrasikan konsep dasar pada sebuah *Convolutional Neural Network* (CNN). Sebuah *input layer* (a) akan diproses oleh serangkaian filter pada *feature-extraction layer* (b) untuk menghasilkan *activation maps* (c) yang menyortir fitur-fitur spesifik dari gambar. Fitur-fitur inilah yang kemudian akan digunakan oleh *classification layer* (d, e, f) untuk melakukan prediksi. Dalam penelitian ini, keseluruhan proses ekstraksi fitur yang kompleks tersebut (tahap a, b, c) dijalankan oleh arsitektur MobileNetV2 yang sudah terbukti.

Setelah MobileNetV2 mengekstraksi fitur-fitur penting, keluarannya dilanjutkan ke tahap *Training Phase* (seperti pada Gambar 3.3), di mana beberapa lapisan kustom ditambahkan sebagai *top model* atau *classifier*:

- **Global Average Pooling 2D:** Lapisan ini berfungsi untuk merangkum dan mengurangi dimensi dari *feature maps* yang dihasilkan oleh MobileNetV2, sehingga lebih efisien untuk diproses.

- **Dropout (30%):** Lapisan ini secara acak menonaktifkan 30% neuron selama pelatihan untuk mencegah *overfitting*, yaitu kondisi di mana model terlalu menghafal data latih dan tidak dapat melakukan generalisasi dengan baik pada data baru. Nilai dari 30% ditentukan oleh perancang model sebelum proses pelatihan dimulai. *dropout rate* antara 20% hingga 50% (0.2 - 0.5) adalah rentang yang paling umum dan terbukti efektif untuk lapisan tersembunyi (*hidden layers*). Nilai 30% merupakan pilihan yang sangat wajar dan moderat di dalam rentang ini, berfungsi sebagai titik awal yang baik untuk memberikan regularisasi tanpa terlalu agresif.
- **Dense 16:** Merupakan sebuah *fully connected layer* dengan 16 neuron yang bertugas untuk mempelajari kombinasi pola dari fitur yang telah diekstraksi. Jumlah 16 neuron disesuaikan dengan jumlah kelas makanan yang akan diklasifikasi.
- **FCL with Softmax:** Ini adalah lapisan *output* akhir yang menggunakan fungsi aktivasi Softmax untuk menghitung dan menghasilkan probabilitas untuk setiap dari 16 kelas makanan.

Hasil akhir dari keseluruhan proses ini adalah tahap *Multi Classification*, dimana sistem memberikan label **Klasifikasi Makanan** dengan probabilitas tertinggi sebagai prediksi akhir untuk gambar yang diinputkan.

3.2.1.6 Pelatihan Model

Pelatihan model merupakan tahap krusial di mana model pembelajaran mesin belajar mengenali pola dari data. Dalam penelitian ini, proses pelatihan difokuskan pada *custom classifier* (terdiri dari lapisan *Global Average Pooling 2D*, *Dropout*, *Dense*, dan *Softmax*) yang telah ditambahkan di atas *base model* MobileNetV2 yang fiturnya telah dibekukan. Tujuannya adalah untuk menyesuaikan parameter pada lapisan-lapisan baru ini agar dapat mengklasifikasikan 16 kategori makanan khas Indonesia secara akurat.

Sebelum pelatihan dimulai, model perlu dikonfigurasi (*compile*) terlebih dahulu dengan menentukan tiga parameter kunci:

- **Optimizer:** Dipilih **Adam Optimizer** dengan *learning rate* awal sebesar **0.001**. Adam dipilih karena efisiensinya yang tinggi dan kemampuannya untuk mengadaptasi laju pembelajaran secara otomatis selama proses pelatihan.
- **Fungsi Loss:** Digunakan **sparse_categorical_crossentropy**. Fungsi ini sangat cocok untuk tugas klasifikasi *multi-class* di mana label target berbentuk integer (misalnya, 0 untuk pecel, 1 untuk rawon, dst.). Fungsi ini akan mengukur seberapa besar kesalahan prediksi model dibandingkan dengan label sebenarnya.
- **Metrik:** Metrik **accuracy** digunakan untuk memantau performa model pada setiap *epoch*.

Proses pelatihan dieksekusi menggunakan data latih yang telah disiapkan. Model akan belajar secara iteratif dalam **50 epoch**, di mana satu *epoch* berarti model telah memproses seluruh data latih sebanyak satu kali. Untuk efisiensi komputasi, data diproses dalam kelompok-kelompok kecil (*batches*) berukuran **8 gambar**.

Selama pelatihan, performa model tidak hanya diukur pada data latih, tetapi juga pada **data validasi** yang tidak pernah dilihat model sebelumnya. Pemantauan pada data validasi ini sangat penting untuk mendeteksi adanya *overfitting*. Untuk mengoptimalkan proses pelatihan dan mencegah *overfitting* secara otomatis, dua strategi *callback* diterapkan:

1. **EarlyStopping:** Mekanisme ini akan menghentikan proses pelatihan secara otomatis jika akurasi pada data validasi tidak menunjukkan peningkatan setelah 5 *epoch* berturut-turut (*patience*=5). Hal ini memastikan pelatihan berhenti pada titik performa terbaik dan menghemat waktu komputasi.
2. **ReduceLROnPlateau:** Strategi ini akan mengurangi *learning rate* secara bertahap jika *loss* pada data validasi tidak kunjung menurun

(*patience*=3). Penyesuaian ini membantu model menemukan solusi yang lebih optimal ketika proses pembelajaran mulai melambat.

Setelah melalui seluruh proses ini, model yang dihasilkan memiliki bobot (*weights*) yang telah dioptimalkan untuk tugas klasifikasi makanan dan siap untuk dievaluasi kinerjanya pada tahap pengujian akhir.

3.2.1.7 Pengujian Model

Pengujian model adalah tahap evaluasi final untuk mengukur seberapa baik kinerja model pada data yang sepenuhnya baru, yaitu **data uji** (*test set*). Tahap ini sangat penting untuk menilai kemampuan **generalisasi** model, yaitu kemampuannya untuk membuat prediksi akurat pada data yang tidak pernah terlihat selama proses pelatihan. Evaluasi yang cermat pada tahap ini memastikan model yang dikembangkan dapat diandalkan saat diimplementasikan dalam kondisi nyata.

3.2.1.7.1 Matrik Evaluasi Kinerja

Kinerja model klasifikasi diukur menggunakan beberapa metrik kuantitatif yang berasal dari hasil **Confusion Matrix**. Confusion Matrix membandingkan prediksi model dengan label aktual, yang dikategorikan ke dalam empat istilah dasar:

- **True Positive (TP)**: Data positif yang berhasil diprediksi dengan benar oleh model.
- **True Negative (TN)**: Data negatif yang berhasil diprediksi dengan benar oleh model.
- **False Positive (FP)**: Data negatif yang salah diprediksi sebagai positif (Kesalahan Tipe I).
- **False Negative (FN)**: Data positif yang salah diprediksi sebagai negatif (Kesalahan Tipe II).

Dari keempat komponen tersebut, beberapa metrik utama dihitung sebagai berikut:

1. Akurasi (*Accuracy*)

Akurasi mengukur rasio total prediksi yang benar (baik positif maupun negatif) terhadap keseluruhan data.

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.1)$$

2. Presisi (*Precision*)

Presisi mengukur tingkat ketepatan dari prediksi positif yang dibuat oleh model. Metrik ini menjawab pertanyaan, "Dari semua yang diprediksi sebagai kelas A, berapa persen yang benar-benar kelas A?"

$$Presisi = \frac{TP}{TP + FP} \quad (3.2)$$

3. Recall (*Sensitivitas*)

Recall mengukur kemampuan model untuk menemukan kembali semua sampel positif yang ada pada dataset. Metrik ini menjawab pertanyaan, "Dari seluruh data kelas A yang sebenarnya, berapa persen yang berhasil ditemukan oleh model?"

$$Recall = \frac{TP}{TP + FN} \quad (3.3)$$

4. F1-Score

F1-Score adalah rata-rata harmonik dari Presisi dan Recall. Metrik ini sangat berguna ketika kita membutuhkan keseimbangan antara Presisi dan Recall, terutama jika distribusi kelas tidak seimbang.

$$F1 - Score = 2 \times \frac{Presisi \times Recall}{Presisi + Recall} \quad (3.4)$$

3.2.1.7.2 Mendiagnosis Masalah Model

Hasil dari pengujian ini juga krusial untuk mendiagnosis masalah umum seperti *overfitting* dan *underfitting*.

- **Overfitting:** Terjadi ketika model memiliki performa sangat tinggi pada data latih, namun performanya turun drastis pada data uji. Ini menandakan model terlalu "menghafal" data latih dan gagal melakukan generalisasi.
- **Underfitting:** Terjadi ketika model memiliki performa yang buruk baik pada data latih maupun data uji. Ini menandakan model terlalu sederhana dan gagal menangkap pola yang ada dalam data.

Dengan melakukan pengujian secara menyeluruh, kita dapat memastikan bahwa model tidak hanya berfungsi baik pada data yang sudah dikenal, tetapi juga andal dan siap untuk digunakan pada data baru di dunia nyata.

3.2.1.8 Deployment

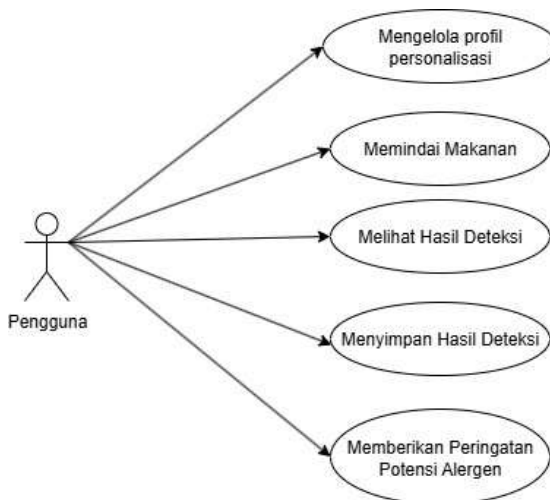
Deployment adalah tahap di mana model yang telah dilatih dan diuji akan diintegrasikan ke dalam sebuah aplikasi *android*. Pada penelitian ini, model yang telah dilatih akan disimpan ke dalam Firebase, sebuah platform cloud yang menawarkan solusi penyimpanan dan hosting yang kuat dan terintegrasi. Firebase memungkinkan penyimpanan model secara aman dan akses cepat oleh aplikasi, yang berarti model dapat diakses dan digunakan untuk prediksi secara real-time.

3.2.2 Perancangan Sistem *Android*

Pada perancangan sistem *android* ini mencakup penggunaan Unified Modeling Language (UML) dengan fokus pada pembuatan Use case

diagram. Usecase diagram ini akan membantu dalam menggambarkan berbagai interaksi antara pengguna dan sistem, mengidentifikasi kebutuhan fungsional, serta memastikan bahwa semua skenario penggunaan utama terdefinisi dengan jelas. Pendekatan ini bertujuan untuk merancang sistem *Android* yang intuitif, efisien, dan mampu memenuhi kebutuhan pengguna secara optimal.

3.2.2.1 Penerapan Use Case



Gambar 3.5 Diagram Use Case Aplikasi

3.2.3 Perancangan User Interface/*Mock-up* aplikasi

Perancangan antarmuka pengguna dalam aplikasi ini bertujuan untuk memberikan kemudahan interaksi bagi pengguna dalam mendeteksi potensi alergi dari makanan yang dikonsumsi. Desain *input/output* disusun berdasarkan alur penggunaan aplikasi yang terdiri dari beberapa komponen utama.

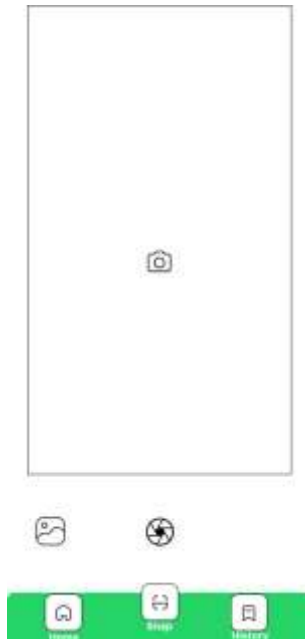
3.2.3.1 Mengelola Profil Personalisasi



Gambar 3.6 Personalisasi Pengguna

Antarmuka ini merupakan implementasi dari use case Mengelola profil personalisasi. Dari ikon profil di pojok kanan atas, pengguna dapat mengakses fitur untuk menambah, melihat, atau menghapus data pribadi, terutama daftar bahan makanan yang memicu alergi. Data yang dimasukkan pada halaman ini akan digunakan oleh sistem untuk menjalankan use case Memberikan Peringatan Potensi Alergen.

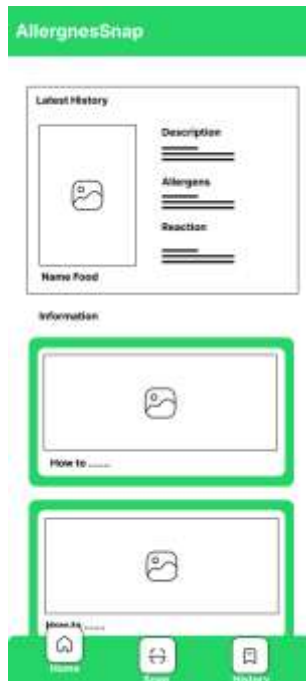
3.2.3.2 Memindai Makanan



Gambar 3.7 Halaman *input* gambar

Halaman ini secara langsung merepresentasikan use case Memindai Makanan. Ketika pengguna menekan tombol "Snap" di navigasi tengah, aplikasi akan menampilkan opsi untuk mengambil gambar melalui kamera atau memilih dari galeri. Setelah gambar dipilih, proses analisis akan dijalankan untuk mengklasifikasi makanan.

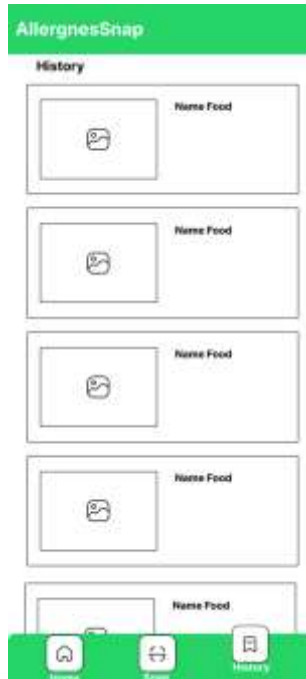
3.2.3.3 Melihat Hasil Deteksi



Gambar 3.8 Halaman Utama

Setelah proses pemindaian selesai, pengguna akan diarahkan ke halaman hasil dengan menampilkan di halaman utama dengan menampilkan informasi lengkap mengenai makanan yang teridentifikasi, seperti nama, deskripsi, bahan-bahan yang berpotensi menjadi alergen, dan kemungkinan reaksi yang ditimbulkan.

3.2.3.4 Menyimpan Hasil Deteksi



Gambar 3.9 Halaman Riwayat Deteksi

Dari Hasil Detek dapat disimpan dihalaman riwayat deteksi. Di Riwayat deteksi ini memunculkan dari gambar yang sudah dianalisa dan nama makanan yang sudah di Analisa dan disimpan.

3.2.3.5 Memberikan Peringatan Potensi Alergen



Gambar 3.10 Nontifikasi Peringatan

Pada fitur ini Jika makanan yang terdeteksi mengandung bahan yang cocok dengan data pada profil pengguna, sistem secara otomatis akan menampilkan notifikasi *pop-up* peringatan, seperti yang ditunjukkan pada Gambar 3.9.

3.3 Rancangan Pengujian

Tujuan pengujian sistem adalah untuk memastikan sistem berfungsi dengan baik dan memenuhi standar yang dipersyaratkan. Dalam studi ini, kinerja model diperiksa menggunakan dua metode: matriks kebingungan dan pengujian kotak hitam. Matriks kebingungan membantu mengevaluasi kinerja model klasifikasi. Matriks ini menunjukkan berapa banyak prediksi yang benar atau salah untuk setiap jenis, yang membantu menemukan dan memahami dimana model mungkin membuat kesalahan. Di sisi lain, pengujian kotak hitam memeriksa sistem dengan melihat apa yang diterima sebagai masukan dan apa yang dihasilkannya sebagai keluaran, tanpa melihat ke dalam sistem. Hal ini membantu memastikan semuanya berfungsi sebagaimana mestinya.

3.3.1 Pengujian *Confusion Matrix*

Pengujian ini bertujuan untuk mengukur dan mengevaluasi performa klasifikasi dari model CNN MobileNetV2 secara menyeluruh. Evaluasi dilakukan dalam dua fase komplementer: pengujian kuantitatif pada dataset statis dan pengujian kualitatif pada aplikasi secara langsung.

1. Pengujian Kuantitatif pada Data Uji

*Pertama, kinerja model diukur secara statistik menggunakan **data uji (test set)** yang belum pernah dilihat oleh model sebelumnya. Hasil prediksi dari model akan dibandingkan dengan label yang sebenarnya dan disajikan dalam bentuk tabel **Confusion Matrix** formal. Dari matriks ini, akan dihitung metrik-metrik evaluasi kuantitatif seperti **Akurasi, Presisi, Recall, dan F1-Score** untuk mendapatkan nilai benchmark objektif atas kemampuan model dalam mengidentifikasi setiap kelas makanan.*

2. Pengujian Kualitatif pada Aplikasi Terintegrasi

Selanjutnya, pengujian kualitatif dilakukan untuk memvalidasi performa model yang telah terintegrasi di dalam aplikasi Android. Berbeda dengan pengujian kuantitatif yang menggunakan dataset statis, pengujian ini mensimulasikan alur penggunaan aplikasi secara end-to-end untuk mengevaluasi kinerjanya dalam skenario yang lebih mendekati kondisi nyata.

Proses pengujian ini melibatkan penggunaan fitur pindai kamera pada aplikasi untuk mendeteksi 10 foto yang berbeda untuk masing-masing dari 16 kelas makanan. Tujuannya adalah untuk memastikan bahwa keseluruhan proses—mulai dari pengambilan gambar melalui antarmuka aplikasi, pra-pemrosesan, hingga inferensi model—dapat berfungsi dengan baik dan memberikan hasil prediksi yang akurat pada serangkaian gambar yang telah disiapkan.

Hasil prediksi dari setiap foto kemudian dicatat untuk dianalisis secara kualitatif. Meskipun tidak menghasilkan metrik statistik formal, hasil ini diinterpretasikan dalam kerangka berpikir Confusion Matrix untuk mengidentifikasi kondisi spesifik yang menyebabkan kesalahan klasifikasi, seperti False Positive (misalnya, salah mengidentifikasi "Rawon" sebagai "Soto") atau False Negative (gagal mengenali "Pecel" dalam kondisi tertentu). Analisis ini memberikan wawasan penting mengenai keandalan model di luar lingkungan pengujian yang terkontrol.

3.3.2 Pengujian *Black Box*

Pengujian *Black Box* adalah metode pengujian perangkat lunak yang berfokus pada fungsionalitas sistem dari sudut pandang pengguna, **tanpa melihat struktur kode internalnya**. Tujuan dari pengujian ini adalah untuk memverifikasi bahwa semua fitur pada **aplikasi Android** berjalan sesuai dengan rancangan dan kebutuhan fungsional.

Pengujian akan dilakukan dengan menjalankan serangkaian **skenario pengujian** yang telah ditentukan. Setiap skenario akan mencakup langkah-langkah aksi, masukan (*input*), dan hasil keluaran (*output*) yang diharapkan. Fitur-fitur utama yang akan diuji meliputi:

- Mengelola profil personalisasi
- Memindai Makanan
- Melihat Hasil Deteksi
- Menyimpan Hasil Deteksi
- Notifikasi Peringatan