

BAB II

TINJAUAN PUSTAKA

A. Kajian Teoritis

1. Klasifikasi Penglihatan Warna

Klasifikasi penglihatan warna pada dasarnya adalah proses menggolongkan hasil pemeriksaan ke dalam kategori *normal*, *protan*, *deutan*, dan *tritan*, serta defisiensi tak tentu (tipe belum dapat dipastikan). Dalam penelitian ini, penggolongan dilakukan secara berbasis aturan (*rule-based*), yakni dari jawaban pengguna pada tes *Ishihara* dan dari pola penyusunan pada tes *Farnsworth D-15*.

Hasil klasifikasi kedua metode lalu dibandingkan untuk melihat seberapa konsisten dan akurat keduanya terhadap hasil pemeriksaan baku. Menurut He (2023) cara menyusun pengukuran sekaligus klasifikasi gangguan warna yang cepat, sedangkan Prajapati & Goyal (2025) menunjukkan pendekatan klasifikasi dapat membuat penilaian penglihatan warna lebih objektif.

Dengan demikian, klasifikasi penglihatan warna, baik berbasis aturan maupun algoritma, dapat dijalankan secara objektif untuk menggolongkan hasil ke kategori *normal*, *protan*, *deutan*, dan *tritan*, sekaligus menjadi dasar untuk membandingkan kinerja metode *Ishihara* dan *Farnsworth D-15* dalam penelitian ini.

2. Penglihatan warna dan Buta warna

Penglihatan warna adalah kemampuan mata menangkap dan membedakan cahaya berdasarkan panjang gelombangnya. Kemampuan ini berasal dari sel kerucut di retina yang terbagi menjadi tiga jenis, masing-masing peka terhadap cahaya gelombang panjang (merah), menengah (hijau), dan pendek (biru). Ketika satu jenis sel kerucut atau lebih tidak bekerja dengan baik, seseorang jadi sulit membedakan warna tertentu. Kondisi inilah yang disebut buta warna atau *color vision deficiency* (CVD).

Buta warna terbagi menjadi dua kelompok besar. Pertama, defisiensi merah-hijau yang mencakup *protan* (*protanomaly* dan *protanopia*) serta *deutan* (*deuteranomaly* dan *deuteranopia*). Kedua, defisiensi biru-kuning atau *tritan* (*tritanomaly* dan *tritanopia*). Dari keduanya, gangguan merah-hijau paling sering dijumpai, sedangkan buta warna total atau *akromatopsia* sangat jarang terjadi.

Sejumlah sumber memperkuat penjelasan ini. Syawal et al. (2018:184) menyebutkan bahwa gangguan penglihatan warna berkaitan dengan kerja sel kerucut di retina dan bisa dikenali lewat pemeriksaan khusus. Gangguan ini umumnya bersifat menurun dan lebih banyak dialami laki-laki (Kitab et al., 2022). Sementara Bano et al. (2024) menjelaskan bahwa aplikasi *mobile* telah dievaluasi ilmiah untuk mengukur penglihatan warna. Dengan demikian, buta warna adalah gangguan persepsi warna akibat sel kerucut yang tidak berfungsi *normal*, terbagi atas *protan*, *deutan*,

dan *tritan*, sehingga perlu dideteksi sedini mungkin dengan metode yang tepat.

3. Metode *Ishihara*

Tes *Ishihara* memakai pelat *pseudoisokromatik*, yaitu gambar yang tersusun dari titik-titik warna dengan ukuran dan kecerahan berbeda-beda hingga membentuk angka atau pola. Orang dengan penglihatan *normal* dapat membaca angka atau pola itu dengan mudah, sedangkan penyandang gangguan merah-hijau akan keliru atau gagal mengenalinya. Karena itu, *Ishihara* cocok dipakai sebagai penyaringan awal untuk mendeteksi gangguan merah-hijau, walaupun belum bisa memastikan seberapa berat tingkat gangguannya.

Penilaian tes *Ishihara* sendiri lebih sederhana, yaitu dengan melihat berapa banyak pelat yang bisa dibaca responden dengan benar. Pada aplikasi ini, tiap pelat dikelompokkan berdasarkan sumbu warnanya, kemudian dihitung seberapa besar fraksi pelat yang salah dibaca (*missFraction*) di tiap sumbu. Bila kesalahan pada sumbu merah-hijau cukup banyak hingga melewati batas tertentu, responden dianggap mengalami gangguan merah-hijau; begitu pula bila kesalahan justru banyak pada sumbu *tritan*, maka yang terdeteksi adalah gangguan *tritan*. Selisih kesalahan antara garis *protan* dan *deutan* lalu dipakai untuk membedakan tipenya. Karena hanya berdasar benar-salahnya pembacaan pelat, tes *Ishihara* lebih cocok dipakai sebagai penyaringan awal daripada untuk mengukur tingkat keparahan gangguan.

Beberapa penelitian menunjukkan versi digital tes *Ishihara* memiliki sensitivitas dan spesifisitas yang sebanding dengan versi cetak, walau hasilnya bisa dipengaruhi karakteristik layar perangkat (Klinke et al., 2024). Menurut (Rashid et al., 2021) tes warna elektronik membuktikan cukup valid dibandingkan pelat *Ishihara* konvensional, dan (Khizer et al., 2022) menilai pemeriksaan lewat *smartphone* berpotensi menjadi alternatif buku *Ishihara* cetak. Artinya, *Ishihara* efektif untuk penyaringan awal gangguan merah-hijau dan tetap valid diterapkan secara digital selama kalibrasi warnanya dijaga. Pada penelitian ini, pelat *Ishihara* dibangkitkan secara digital pada beberapa sumbu konfusi warna (*protan*, *deutan*, dan *tritan*) memanfaatkan ruang warna *CIELAB*, sehingga mampu membedakan ketiga jenis defisiensi warna dan tidak terbatas pada gangguan merah-hijau.

4. Metode *Farnsworth D-15*

Tes *Farnsworth D-15* termasuk metode berbasis penyusunan warna (*color arrangement test*). Tes ini terdiri atas 15 kepingan warna ditambah satu kepingan acuan, dan peserta diminta mengurutkannya sesuai kemiripan warna. Urutan yang keliru kemudian dibaca polanya untuk menentukan jenis gangguan, yaitu *protan*, *deutan*, atau *tritan*, sekaligus tingkat keparahannya. Jadi, jika *Ishihara* berfokus pada gangguan merah-hijau, *Farnsworth D-15* mampu memetakan ketiga jenis gangguan warna sekaligus.

Untuk menilai hasil tes *Farnsworth D-15*, aplikasi ini memakai cara penilaian yang sama seperti tes *Farnsworth D-15* pada umumnya. Urutan kepingan yang disusun responden dihitung nilai kesalahannya menggunakan *Total Error Score* (TES), lalu jumlah *crossing* atau garis yang memotong pusat lingkaran warna dipakai untuk menentukan lulus tidaknya responden. Kalau responden dinyatakan tidak lulus, arah garis kesalahan tadi dibandingkan dengan tiga sumbu warna (*protan*, *deutan*, dan *tritan*) untuk mengetahui tipe gangguannya. Cara penilaian seperti ini mengikuti pendekatan penilaian kuantitatif pada tes susunan warna, seperti perhitungan *Total Error Score* dan sudut kebingungan warna (*confusion angle*), yang masih digunakan dalam penelitian terkini mengenai pemeriksaan *Farnsworth D-15* secara terkomputerisasi (Hsu et al., 2025).

Penelitian terdahulu memperlihatkan tes *Farnsworth D-15* versi aplikasi memiliki tingkat kesesuaian (*agreement*) yang tinggi dengan tes baku, sehingga layak dipakai pada perangkat bergerak (Thammakumpee et al., 2025), dan nilai diagnostiknya pun tetap stabil meski akurasi tampilan warna tiap perangkat berbeda, walaupun faktor sinyal visual tetap perlu diperhitungkan saat klasifikasi (Evans et al., 2021). Menurut Arpadffy-Lovas & Toth-Molnar (2024) juga menemukan nilai diagnostik tes susunan warna *Farnsworth D-15* tetap terjaga meski ada keterbatasan pada akurasi tampilan perangkat. Dengan begitu, *Farnsworth D-15* dapat diandalkan untuk mengklasifikasikan jenis dan derajat gangguan warna, yaitu *protan*, *deutan*, dan *tritan*, serta tetap relevan diterapkan pada perangkat bergerak.

5. Ruang Warna *CIELAB*

Ruang warna *CIELAB* merupakan model warna yang ditetapkan oleh *Commission Internationale de l'Éclairage* (CIE) untuk menyatakan warna dalam tiga nilai, yaitu L^* sebagai tingkat kecerahan, a^* sebagai sumbu merah-hijau, dan b^* sebagai sumbu kuning-biru. Model ini dirancang agar selisih angka antarwarna mendekati selisih yang dirasakan oleh mata manusia, sehingga sesuai untuk membangkitkan dan membandingkan warna secara terukur. Selain itu, *CIELAB* bersifat tidak bergantung pada perangkat (*device-independent*) sehingga nilai warna tidak terikat pada karakteristik satu layar tertentu.

Menurut Marefat et al. (2025) ruang warna *CIELAB* telah diakui luas sebagai acuan pengukuran warna yang bersifat *device-independent* dan *perceptually uniform*, sehingga selisih warna dapat dihitung secara objektif melalui koordinat L , a , dan b . (Suzuki et al., 2024) menunjukkan bahwa perubahan warna dapat direpresentasikan secara konsisten pada ruang *CIELAB* sehingga cocok untuk pelacakan warna. Sementara itu, (Pambudi et al., 2021) memanfaatkan *CIELAB* pada aplikasi *Android* karena ruang warna ini memisahkan komponen kecerahan (L) dari komponen *kromatik* (a dan b), sehingga proses analisis warna lebih stabil terhadap perubahan pencahayaan dibandingkan ruang warna RGB.

Dengan demikian, ruang warna *CIELAB* dipilih dalam penelitian ini sebagai dasar pembangkitan warna pada pelat *Ishihara* dan kepingan *Farnsworth D-15*. Nilai warna dihitung terlebih dahulu pada ruang

CIELAB kemudian dikonversi ke ruang warna layar (sRGB) agar warna yang ditampilkan konsisten dan sesuai dengan garis konfusi tiap jenis buta warna.

6. Analisis Akurasi

Akurasi klasifikasi diukur dengan membandingkan hasil klasifikasi sistem terhadap kelas sebenarnya dari pemeriksaan baku. Nilainya diperoleh dari persentase kesesuaian antara hasil klasifikasi sistem dan hasil pemeriksaan baku, yaitu seberapa banyak data yang diklasifikasikan dengan benar dari keseluruhan data yang diuji. Nilai akurasi inilah yang dipakai untuk membandingkan kinerja metode *Ishihara* dan *Farnsworth D-15* dalam mengklasifikasikan penglihatan warna.

$$\text{Akurasi} = \left(\frac{\text{Jumlah klasifikasi benar}}{\text{Jumlah seluruh data yang diuji}} \right) \times 100\%$$

Parameter akurasi berupa persentase kesesuaian terhadap hasil pemeriksaan baku sudah lazim dipakai untuk menilai kinerja klasifikasi penglihatan warna pada penelitian sebelumnya (He, 2023; Prajapati & Goyal, 2025; Yang, 2024).

Karena mampu menghasilkan angka yang terukur dan objektif, parameter akurasi ini dinilai tepat untuk membandingkan kinerja metode *Ishihara* dan *Farnsworth D-15* terhadap hasil pemeriksaan baku dalam penelitian ini.

7. Analisis Kesepakatan: *Cohen's Kappa*

Cohen's Kappa merupakan ukuran statistik yang digunakan untuk menilai tingkat kesepakatan (*agreement*) antara dua penilai atau dua metode pada data berkategori. Berbeda dari sekadar persentase kecocokan, *Cohen's Kappa* memperhitungkan kemungkinan kecocokan yang terjadi secara kebetulan sehingga hasilnya lebih adil. Nilainya berkisar dari -1 hingga +1, dengan nilai 0 berarti kesepakatan setara peluang acak dan nilai 1 berarti kesepakatan sempurna.

Menurut Rainio et al. (2024) menjelaskan bahwa *Cohen's Kappa* sebaiknya dilaporkan berdampingan dengan metrik akurasi agar gambaran kinerja lebih menyeluruh. Menurut Li et al. (2023) *Cohen's Kappa* tepat digunakan untuk menilai kesepakatan dua penilai pada data kategorik dengan dua kelas maupun kategori tak berjenjang dengan tiga kelas atau lebih. Sejalan dengan itu, Sathyanarayanan & Tantri (2024) menempatkan *Cohen's Kappa* sebagai salah satu metrik penting untuk menilai keandalan hasil klasifikasi.

Dengan demikian, *Cohen's Kappa* digunakan dalam penelitian ini sebagai parameter pelengkap untuk mengukur kekuatan kesepakatan antara hasil klasifikasi metode *Ishihara* dan metode *Farnsworth D-15*. Penggunaan *Cohen's Kappa* membuat perbandingan kedua metode menjadi lebih objektif karena tidak hanya bergantung pada persentase kecocokan semata.

8. Aplikasi *Mobile* Berbasis *Android*

Aplikasi *mobile* berbasis *Android* adalah perangkat lunak yang berjalan di sistem operasi *Android* dan bisa diakses lewat *smartphone* maupun *tablet*. Untuk pemeriksaan penglihatan warna, aplikasi semacam ini menawarkan kemudahan akses, mudah dibawa, serta mampu menghitung hasil secara otomatis dan objektif.

Aplikasi pemeriksaan warna pada perangkat bergerak pun sudah dikembangkan di banyak penelitian sebagai pengganti cara konvensional. Menurut Aklani & Galen (2022), Saini et al. (2022), serta Nauli et al. (2026) menunjukkan bahwa memindahkan tes buta warna ke perangkat *Android* membuat pemeriksaan bisa dilakukan sendiri, praktis, dan mudah dijangkau.

Dengan demikian, aplikasi *Android* memang cocok dijadikan wadah sistem deteksi buta warna karena portabel sekaligus mampu menghitung hasil secara otomatis dan objektif.

a. Bahasa Pemrograman *Kotlin*

Kotlin adalah bahasa pemrograman modern buatan *JetBrains* yang kini resmi didukung *Google* sebagai bahasa utama pengembangan aplikasi *Android*. *Kotlin* dikenal ringkas, aman terhadap nilai kosong (*null safety*), dan bisa dipakai berdampingan (*interoperable*) dengan Java sehingga pustaka lama tetap dapat dimanfaatkan. Karena sifat-sifat itu, *Kotlin* banyak dipilih untuk membangun aplikasi *Android* masa kini. Pada penelitian ini, *Kotlin* dipakai untuk menuliskan logika

tes *Ishihara*, tes *Farnsworth D-15*, klasifikasi penglihatan warna, serta perhitungan akurasi.

Menurut Pratama et al. (2025) menyebut kode *Kotlin* lebih ringkas dan efisien dibandingkan Java pada aplikasi *Android* sehingga kinerjanya lebih baik. Hal serupa ditegaskan Sanjaya & Susilo (2024) yang menemukan *Kotlin* lebih unggul dari sisi kecepatan eksekusi serta lebih hemat memori dan *Central Processing Unit* (CPU) dibanding Java, dan Wicaksono & Leong (2021) yang mencatat pemakaian memori *Kotlin* lebih baik pada aplikasi *Android*. Ketiga temuan ini menguatkan pilihan *Kotlin* untuk membangun aplikasi deteksi buta warna yang ringkas, efisien, dan berkinerja baik.

Dengan kata lain, *Kotlin* merupakan pilihan yang tepat untuk penelitian ini karena mendukung pembuatan aplikasi *Android* yang ringkas, aman, dan efisien.

b. *Android Studio*

Android Studio adalah *Integrated Development Environment* (IDE) resmi dari Google untuk membangun aplikasi *Android*. Dibangun di atas *IntelliJ IDEA*, perangkat ini dilengkapi editor kode cerdas, perancang antarmuka (*layout editor*), *emulator*, dan sistem *build* berbasis *Gradle*.

Menurut Aklani & Galen (2022) memakainya untuk membuat aplikasi tes buta warna berbasis *Android*, sedangkan Kurnia (2025) menggunakannya untuk aplikasi pengelola data ber-fitur CRUD dan

basis data *SQLite*, sedangkan Diantoni et al. (2024) memanfaatkannya untuk aplikasi *Android* berarsitektur modern.

Karena fiturnya lengkap untuk menulis kode, merancang antarmuka, sekaligus menguji aplikasi dalam satu tempat, *Android Studio* dipilih sebagai lingkungan utama pengembangan sistem deteksi buta warna dalam penelitian ini.

c. *Material Design* dan Komponen Antarmuka

Material Design merupakan sistem pedoman desain antarmuka yang dikembangkan oleh *Google* untuk menghasilkan tampilan aplikasi yang konsisten, jelas, dan nyaman digunakan pada berbagai perangkat. Pedoman ini diwujudkan melalui pustaka komponen siap pakai, seperti tombol, kartu (*card*), dan kotak centang (*checkbox*), yang membantu pengembang menyusun antarmuka secara rapi dan seragam. Pada aplikasi ini, komponen *Material Design* dipadukan dengan tata letak fleksibel untuk menata elemen-elemen tes warna.

Menurut Rahmatunnisa (2025) penerapan prinsip desain antarmuka yang baik menjadikan aplikasi *Android* lebih mudah dipahami oleh pengguna. Sanjaya & Susilo (2024) menyatakan bahwa pembangunan antarmuka *Android* menggunakan *Kotlin* memiliki kinerja yang efisien dan ringkas. Selain itu, Nugroho et al. (2024) menunjukkan bahwa penataan struktur kode yang rapi dapat memisahkan tampilan dari logika program sehingga aplikasi lebih mudah dikembangkan dan dipelihara.

Dengan demikian, penggunaan *Material Design* beserta komponen antarmukanya dalam penelitian ini bertujuan menghadirkan tampilan yang konsisten dan mudah digunakan oleh responden saat menjalani tes buta warna, sekaligus memudahkan proses pengembangan sistem pada tahap selanjutnya.

d. Basis Data *SQLite*

SQLite merupakan sistem manajemen basis data relasional yang bersifat ringan dan tertanam langsung di dalam aplikasi tanpa memerlukan server terpisah. Karena tidak membutuhkan koneksi jaringan, *SQLite* sesuai untuk aplikasi *mobile* yang perlu menyimpan data secara lokal dan tetap dapat berjalan dalam kondisi luring (*offline*). Pada *Android*, basis data ini disimpan di area privat aplikasi sehingga relatif aman dari akses aplikasi lain.

Menurut Kurnia (2025) *SQLite* dapat dimanfaatkan sebagai penyimpanan data lokal untuk menjalankan fungsi dasar pengelolaan data, yaitu membuat, membaca, memperbarui, dan menghapus (CRUD). Bartoszek (2024) membandingkan beberapa jenis basis data pada aplikasi *mobile* dan menemukan *SQLite* memberikan hasil terbaik pada kueri kompleks, meskipun performanya lebih rendah pada operasi baca-tulis data dalam jumlah besar. Sejalan dengan itu, Aziz (2022) menunjukkan seluruh fungsi CRUD berbasis *SQLite* dapat berjalan sesuai spesifikasi tanpa memerlukan koneksi internet.

Dengan demikian, *SQLite* dipilih sebagai media penyimpanan pada penelitian ini untuk menyimpan data diri responden serta riwayat hasil tes *Ishihara* dan *Farnsworth D-15*. Data yang tersimpan secara lokal memudahkan proses perbandingan hasil kedua metode pada perangkat pengguna.

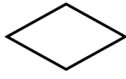
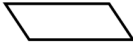
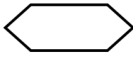
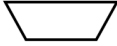
9. *Flowchart*

Menurut Listyoningrum et al. (2023), *Flowchart* adalah gambar yang menyajikan urutan proses atau langkah kerja sebuah program secara sistematis. Lewat simbol-simbol yang dihubungkan panah sesuai urutannya, *Flowchart* menggambarkan algoritma maupun alur kerja sehingga proses lebih mudah dianalisis, dirancang, dan didokumentasikan.

Menurut Tuasamu (2023) *Flowchart* adalah teknik analisis bergambar untuk menjelaskan berbagai bagian sistem informasi secara jelas dan logis, sekaligus menunjukkan bagaimana proses dan aliran dokumen berjalan di sebuah organisasi. Pada penelitian ini, *Flowchart* dipakai sebagai alat bantu untuk menggambarkan alur proses sistem agar lebih mudah dianalisis, dirancang, dan didokumentasikan.

Tabel 2.1 Simbol *Flowchart*

No	Simbol	Nama	Keterangan
1		<i>Terminator</i> (Terminal)	Menyatakan awal (Mulai) dan akhir (Selesai) dari suatu program atau proses.
2		<i>Flow Line</i> (Garis Alir)	Menunjukkan arah aliran proses dari satu langkah ke langkah berikutnya.

No	Simbol	Nama	Keterangan
3		<i>Process</i> (Proses)	Menyatakan proses pengolahan atau perhitungan data.
4		<i>Decision</i> (Keputusan)	Menyatakan percabangan atau pengambilan keputusan berdasarkan kondisi (ya/tidak).
5		<i>Input/Output</i> (Data)	Menyatakan proses masukan (<i>input</i>) atau keluaran (<i>output</i>) data.
6		<i>Preparation</i> (Persiapan)	Menyatakan pemberian nilai awal atau persiapan suatu variabel/penghitung.
7		<i>Predefined Process</i>	Menyatakan proses yang detailnya didefinisikan terpisah (subprogram/prosedur).
8		<i>Connector</i> (Penghubung)	Penghubung bagian-bagian <i>Flowchart</i> pada halaman yang sama.
9		<i>Off-Page Connector</i>	Penghubung <i>Flowchart</i> yang berlanjut ke halaman yang berbeda.
10		<i>Document</i> (Dokumen)	Menyatakan masukan/keluaran berupa dokumen atau laporan tercetak.
11		<i>Manual Input</i>	Menyatakan pemasukan data secara manual melalui <i>keyboard</i> .
12		<i>Manual Operation</i>	Menyatakan proses atau operasi yang dilakukan secara manual (tanpa komputer).
13		<i>Stored Data</i> (Basis Data)	Menyatakan penyimpanan data ke dalam basis data.

10. *Unified Modeling Language (UML)*

Unified Modeling Language (UML) adalah bahasa pemodelan visual untuk menggambarkan, merancang, dan mendokumentasikan sistem perangkat lunak berorientasi objek. UML menyediakan bermacam *diagram* untuk menampilkan struktur maupun perilaku sistem, sehingga komunikasi antara perancang dan pengembang menjadi lebih mudah.

Menurut Ramdany et al. (2021) UML berperan sebagai bahasa pemodelan baku untuk merancang sekaligus mendokumentasikan sistem berorientasi objek, dengan *diagram* yang umum dipakai seperti *Use case diagram*, *Activity diagram*, *Sequence Diagram*, dan *Class diagram*. Sumirat et al. (2023:74) menambahkan bahwa UML dipakai dalam analisis dan perancangan berorientasi objek untuk menggambarkan struktur serta perilaku sistem, sementara Hadi et al. (2025) memadukan UML untuk menggambarkan rancangan sistem berbasis *web* secara sistematis. Mengacu pada ketiga sumber itu, UML dipakai dalam penelitian ini untuk menggambarkan kebutuhan dan rancangan aplikasi deteksi buta warna agar lebih mudah dipahami.

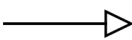
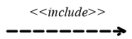
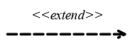
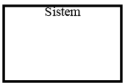
Ringkasnya, UML adalah bahasa pemodelan standar untuk menggambarkan struktur dan perilaku perangkat lunak. Pada penelitian ini digunakan empat jenis *diagram* UML sebagai berikut.

a. *Use case diagram*

Use case diagram adalah *diagram* UML yang menggambarkan interaksi antara aktor dan sistem sekaligus menandai kebutuhan

fungsionalnya. Pada penelitian ini, *Use case diagram* dipakai untuk menggambarkan bagaimana pengguna menjalankan tes *Ishihara* dan *Farnsworth D-15* serta melihat hasil klasifikasinya.

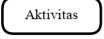
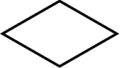
Tabel 2.2 Simbol *Use case diagram*

No	Simbol	Nama	Keterangan
1		<i>Actor</i> (Aktor)	Menggambarkan pengguna atau sistem lain yang berinteraksi dengan sistem.
2		<i>Use case</i>	Menggambarkan fungsionalitas atau layanan yang disediakan sistem kepada aktor.
3		<i>Association</i> (Asosiasi)	Menghubungkan aktor dengan <i>use case</i> yang berinteraksi dengannya.
4		<i>Generalization</i>	Relasi pewarisan, yaitu sebuah elemen merupakan bentuk khusus dari elemen lain.
5		<i>Include</i>	Menunjukkan <i>use case</i> yang selalu memanggil (menyertakan) <i>use case</i> lain.
6		<i>Extend</i>	Menunjukkan <i>use case</i> yang dapat memperluas perilaku <i>use case</i> lain pada kondisi tertentu.
7		<i>System Boundary</i>	Batas ruang lingkup sistem yang memisahkan sistem dari aktor.

b. *Activity diagram*

Activity diagram adalah *diagram* UML yang menggambarkan alur aktivitas atau proses kerja sistem secara berurutan. *Diagram* ini memudahkan pemahaman logika proses, mulai dari pengguna memulai tes sampai sistem menampilkan hasil pemeriksaan.

Tabel 2.3 Simbol *Activity diagram*

No	Simbol	Nama	Keterangan
1	●	<i>Initial Node</i>	Titik awal dimulainya suatu rangkaian aktivitas.
2		<i>Action / Activity</i>	Aktivitas atau aksi yang dilakukan dalam alur proses.
3		<i>Decision / Merge</i>	Percabangan atau penggabungan alur berdasarkan suatu kondisi.
4		<i>Fork / Join</i>	Pemisahan (<i>fork</i>) atau penggabungan (<i>join</i>) alur aktivitas yang berjalan paralel.
5		<i>Control Flow</i>	Menunjukkan arah aliran dari satu aktivitas ke aktivitas berikutnya.
6	⊙	<i>Final Node</i>	Titik akhir berakhirnya rangkaian aktivitas.

c. *Sequence Diagram*

Sequence Diagram adalah *diagram* UML yang menggambarkan urutan interaksi antarobjek berdasarkan waktu kejadiannya. *Diagram* ini memberi gambaran rinci mengenai bagaimana antarkomponen saling berkomunikasi selama proses pemeriksaan.

Tabel 2.4 Simbol *Sequence diagram*

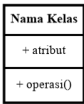
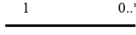
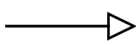
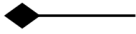
No	Simbol	Nama	Keterangan
1		<i>Object / Lifeline</i>	Objek beserta garis hidup (<i>lifeline</i>) selama interaksi berlangsung.
2		<i>Activation</i>	Periode saat objek aktif menjalankan suatu proses.

No	Simbol	Nama	Keterangan
3		<i>Message</i>	Pesan atau pemanggilan metode dari satu objek ke objek lain.
4		<i>Return Message</i>	Pesan balikan (<i>return</i>) sebagai respon atas pesan yang dikirim.
5		<i>Self Message</i>	Pesan yang dikirim objek kepada dirinya sendiri.

d. *Class diagram*

Class diagram adalah *diagram* UML yang menggambarkan struktur kelas dalam sistem beserta atribut, metode, dan hubungan antarkelasnya. *Diagram* ini memetakan kelas-kelas utama aplikasi, seperti kelas tes, kelas analisis, dan kelas penyimpanan data.

Tabel 2.5 Simbol *Class diagram*

No	Simbol	Nama	Keterangan
1		<i>Class (Kelas)</i>	Menggambarkan kelas yang terdiri atas nama kelas, atribut, dan operasi/metode.
2		<i>Association</i>	Relasi struktural yang menghubungkan dua kelas.
3		<i>Multiplicity</i>	Menyatakan jumlah keterhubungan antar objek (misalnya 1, 0..*, 1..*).
4		<i>Generalization / Inheritance</i>	Relasi pewarisan; subkelas mewarisi atribut dan operasi dari superkelas.
5		<i>Aggregation</i>	Relasi keseluruhan-bagian yang bersifat lemah (bagian dapat berdiri sendiri).
6		<i>Composition</i>	Relasi keseluruhan-bagian yang bersifat kuat (bagian

No	Simbol	Nama	Keterangan
			bergantung pada keseluruhan).
7	----->	<i>Dependency</i>	Ketergantungan; perubahan pada satu kelas dapat memengaruhi kelas lain.
8	-----▷	<i>Realization</i>	Relasi antara antarmuka (<i>interface</i>) dengan kelas yang mengimplementasikannya.
9	+ - #	Hak Akses	Visibilitas anggota kelas: + <i>public</i> , - <i>private</i> , # <i>protected</i> .

11. Metode *Prototype*

Metode *Prototype* adalah metode pengembangan perangkat lunak yang menekankan pembuatan purwarupa (*Prototype*) sistem secara cepat agar diperoleh gambaran awal dan umpan balik dari calon pengguna. Sumirat et al. (2023:33) menjelaskan bahwa *Prototype* membantu memvalidasi kebutuhan dan memahami persyaratan dengan lebih baik sebelum pembangunan sistem yang sesungguhnya dimulai. Lewat *Prototype*, pengembang dan pengguna dapat berinteraksi langsung sehingga kebutuhan sistem makin jelas dan bisa disempurnakan sedikit demi sedikit. Metode ini pas untuk penelitian ini karena aplikasi deteksi buta warna sangat bergantung pada tampilan antarmuka yang perlu diuji dan diperbaiki mengikuti tanggapan pengguna.

Menurut Albab & Suhendra (2024) menerapkan metode *Prototype* pada pengembangan aplikasi *Android* melalui lima tahap, yaitu analisis kebutuhan, pembentukan *Prototype*, evaluasi *Prototype*, pengodean, dan pengujian. Aprilisa & Aulia (2024) juga memakai metode *Prototype* untuk

sistem *informasi* berbasis *web* dan menemukan bahwa keterlibatan pengguna saat mengevaluasi *Prototype* membantu menyesuaikan sistem dengan kebutuhan. Kedua kajian ini memperlihatkan metode *Prototype* memang cocok untuk aplikasi yang mengandalkan interaksi dan umpan balik pengguna.

Dengan demikian, metode *Prototype* tepat dipakai dalam penelitian ini karena kebutuhan sistem bisa diperjelas secara bertahap lewat pembuatan dan evaluasi *Prototype* bersama pengguna.

12. Pengujian *Black box*

Pengujian *black box* adalah teknik menguji perangkat lunak yang berfokus pada fungsi sistem tanpa melihat isi kodenya. Caranya, sistem diberi masukan tertentu lalu diperiksa apakah keluarannya sudah sesuai harapan. Teknik ini dipakai untuk memastikan setiap fitur utama aplikasi bekerja sebagaimana mestinya.

Menurut Zen et al. (2024), pengujian *black box* menitikberatkan pada fungsi perangkat lunak dengan mencocokkan masukan dan keluaran tanpa meninjau struktur kodenya. Handayanto & Nuryasin (2024) menerapkan *Black box Testing* lewat teknik *decision table* untuk memastikan tiap fungsi aplikasi *mobile* berjalan sesuai tujuannya. Merujuk kedua sumber itu, pengujian *black box* dipakai dalam penelitian ini untuk memastikan seluruh fitur aplikasi deteksi buta warna berfungsi dengan benar.

Dengan kata lain, *black box* merupakan teknik pengujian yang praktis dan berfokus pada fungsi sistem, sehingga cocok untuk memastikan setiap fitur aplikasi berjalan sesuai kebutuhan.



Gambar 2.1 *Black box Testing*

Sumber: Fedianto et al (2024)

B. Kajian Empiris

Kajian empiris dalam penelitian ini diambil dari sejumlah penelitian terdahulu yang relevan pada rentang 2021-2026. (Thammakumpee et al., 2025) menunjukkan adanya kesesuaian (*agreement*) yang baik antara tes *Farnsworth D-15* versi aplikasi *smartphone* dan versi bakunya, sehingga memperkuat penerapan tes *Farnsworth D-15* pada perangkat bergerak. (Klinke et al., 2024) membandingkan sensitivitas dan spesifisitas tes *Ishihara* pada berbagai jenis tampilan dan mendapati jenis layar ikut memengaruhi hasil, sehingga *kalibrasi* warna menjadi sangat penting.

Menurut Khizer et al. (2022) pemeriksaan warna berbasis *smartphone* berpeluang menjadi alternatif buku *Ishihara* konvensional, sedangkan (Rashid et al., 2021) membuktikan validitas tes warna elektronik terhadap pelat *Ishihara* konvensional. Arpadffy-Lovas & Toth-Molnar (2024) juga menemukan nilai diagnostik tes *Farnsworth D-15* tetap terjaga meski tampilan perangkat memiliki keterbatasan akurasi.

Dari sisi metode klasifikasi, penelitian tentang pengukuran cepat dan klasifikasi gangguan warna berbasis *machine learning* serta beberapa penelitian lain menunjukkan pendekatan klasifikasi mampu meningkatkan objektivitas penilaian penglihatan warna (He, 2023; Henshaw et al., 2025; Prajapati & Goyal, 2025). Di tingkat nasional, penelitian seperti perancangan aplikasi tes buta warna metode *Ishihara* berbasis *Android* memperlihatkan digitalisasi tes *Ishihara* pada perangkat *Android* dapat menunjang pemeriksaan mandiri (Aklani & Galen, 2022; Nauli et al., 2026; Saini et al., 2022; Widyadnyana et al., 2024). Sementara itu, Fareza & Toyib (2024) memakai pengolahan citra digital pada buku *Ishihara* dengan metode *K-Means* untuk menentukan tingkat buta warna, dengan akurasi yang berbeda-beda antar citra.

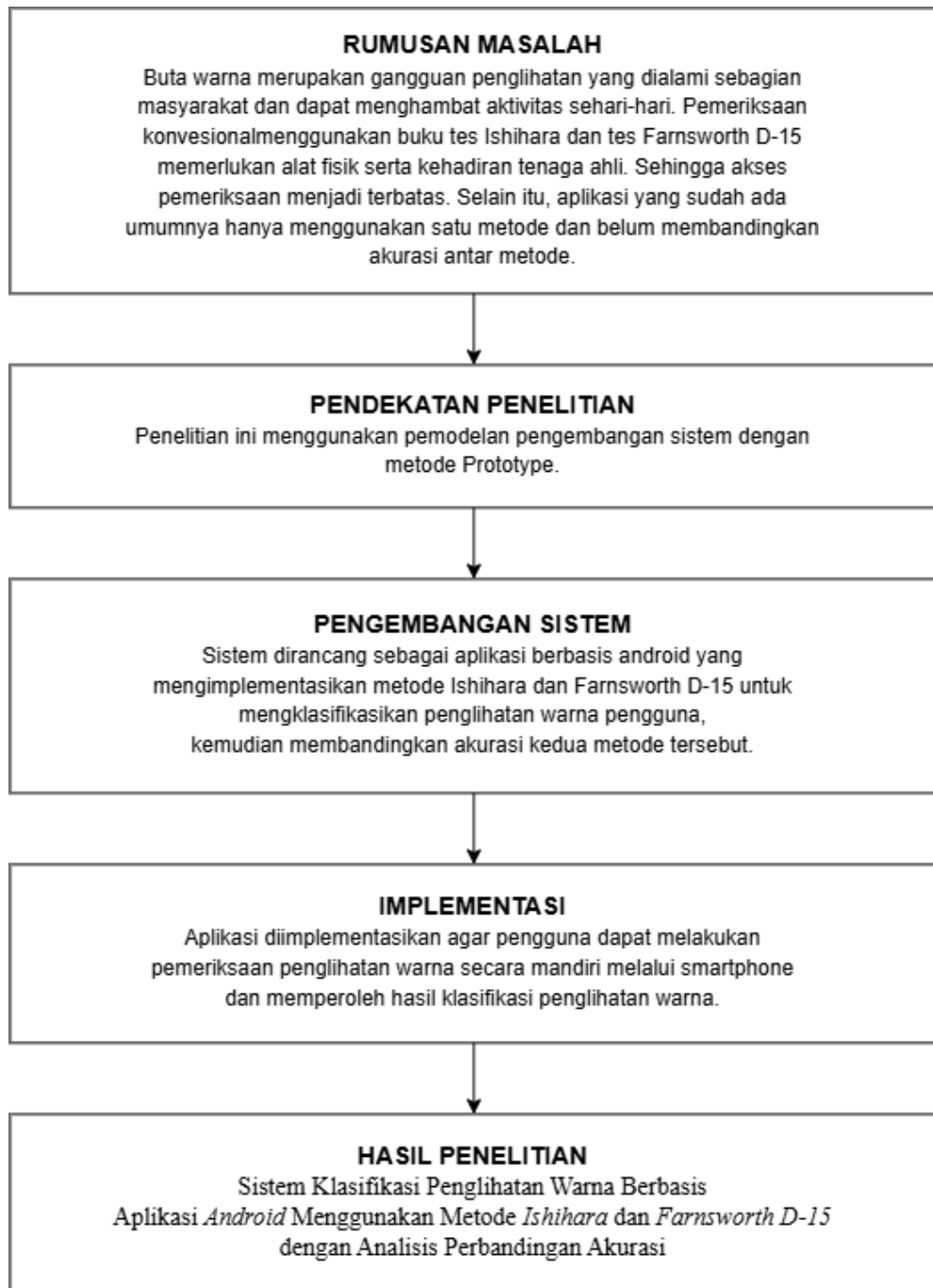
Dari berbagai penelitian terdahulu itu terlihat bahwa kajian sebelumnya sudah berkontribusi pada digitalisasi tes *Ishihara* dan *Farnsworth D-15*, tetapi umumnya masih dilakukan terpisah. Di sinilah muncul celah penelitian (*research gap*): belum banyak sistem yang menyatukan kedua metode dalam satu aplikasi sekaligus membandingkan akurasinya dalam mengklasifikasikan penglihatan warna. Celah itulah yang menjadi fokus penelitian ini.

C. Kerangka Berpikir

Penelitian ini berawal dari masalah bahwa pemeriksaan buta warna konvensional dengan buku *Ishihara* dan tes *Farnsworth D-15* memiliki sejumlah keterbatasan, mulai dari ketersediaan alat, ketergantungan pada pencahayaan, hingga kemungkinan penilaian yang subjektif. Selain itu,

penelitian terdahulu kebanyakan hanya memakai satu metode dan belum membandingkan akurasi keduanya dalam satu sistem.

Mengacu pada kajian empiris, langkah yang relevan adalah membangun aplikasi *Android* yang menjalankan metode *Ishihara* dan *Farnsworth D-15*. Sistem dirancang agar mampu menjalankan kedua tes, menggolongkan hasilnya ke kategori *normal*, *protan*, *deutan*, dan *tritan*, lalu membandingkan akurasi kedua metode terhadap hasil pemeriksaan baku menggunakan analisis akurasi dan *Cohen's Kappa*. Subjek uji cobanya adalah masyarakat umum sebagai responden. Luaran yang diharapkan berupa aplikasi deteksi buta warna sekaligus hasil analisis perbandingan akurasi metode *Ishihara* dan *Farnsworth D-15*.



Gambar 2.2 Kerangka Berpikir