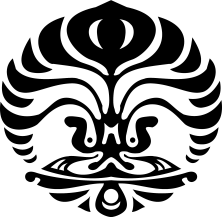
****

**UNIVERSITAS INDONESIA**

**ANALISIS INTEGRASI COLOR CODE DENGAN SISTEM RUMAH PINTAR BERBASIS SISTEM OPERASI ANDROID**

**SKRIPSI**

**JODHI ADHIKAPRANA SARDJONO**

**1106020674**

**FAKULTAS TEKNIK UNIVERSITAS INDONESIA**

**DEPARTEMEN TEKNIK ELETKRO**

**TEKNIK KOMPUTER**

**DEPOK**

**DESEMBER**

**2014**

HALAMAN PERNYATAAN ORISINALITAS

**Skripsi ini adalah hasil karya saya sendiri,**

**dan semua sumber baik yang dikutip maupun dirujuk**

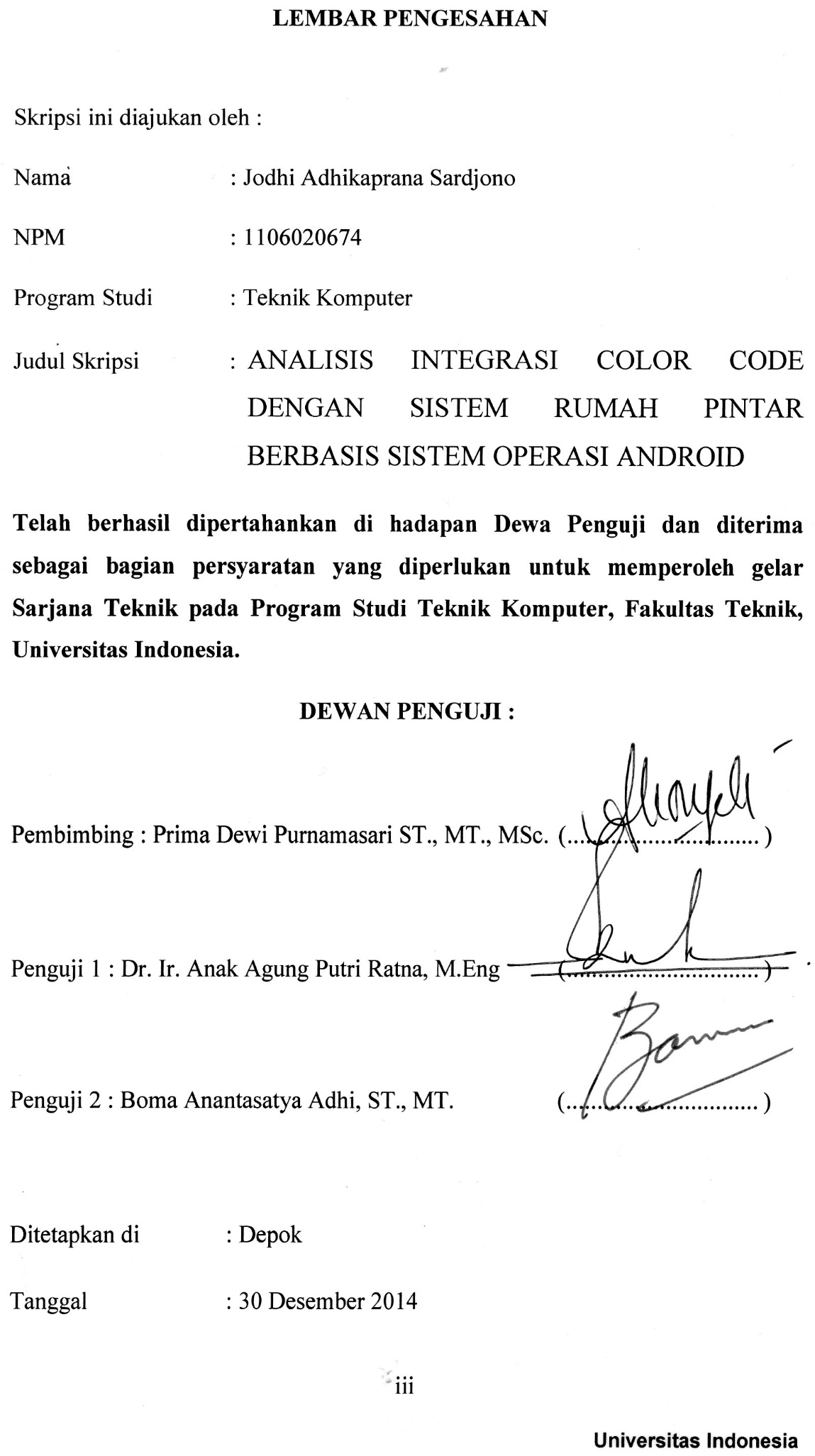
**telah saya nyatakan dengan benar.**

Nama : Jodhi Adhikaprana Sardjono

NPM : 1106020674

Tanda Tangan : 

Tanggal : 30 Desember 2014

****

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah SWT, sehingga dengan rahmat dan karunia-Nya, penulis dapat menyelesaikan karya tulis ini dengan baik. Penulis menyadari bahwa karya tulis ini tidak dapat diselesaikan tanpa dukungan dari berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada pihak-pihak di bawah ini yang telah membantu dalam proses penyelesaian skripsi ini:

1. Ibu Prima Dewi Purnamasari ST., MT., MSc. selaku pembimbing penulisan ini yang telah memberikan pengarahan, koreksi, dukungan, dan waktu selama penulis mengerjakan seminar ini.
2. Ibu, bapak, kakak, dan adik penulis yang telah memberikan dukungan baik moril maupun materil sehingga penulis dapat menyelesaikan seminar ini.
3. Keluarga besar Departemen Teknik Elektro 2011 dan sahabat-sahabat penulis.

Penulis berdoa semoga Allah SWT berkenan membalas kebaikan dari semua pihak yang telah berbaik hati membantu penulis dan semoga seminar ini dapat bermanfaat bagi pengembangan teknologi dan ilmu pengetahuan.

Depok, 30 Desember 2014



Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Indonesia, saya yang bertanda tangan dibawah ini:

Nama : Jodhi Adhikaprana Sardjono

NPM : 1106020674

Program Studi : Teknik Komputer

Departemen : Teknik Elektro

Fakultas : Teknik

Jenis Karya : Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia Hak Bebas Royalti Non Eksklusif (Non-Exclusive Royalty Free Right) atas karya ilmiah saya yang berjudul:

**“ANALISIS INTEGRASI COLOR CODE DENGAN SISTEM RUMAH PINTAR BERBASIS SISTEM OPERASI ANDROID”**

Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/format-kan, mengelola dalam bentuk pangkalan data (database), merawat, dan memublikasikan skripsi saya selama tetap mencatumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di: Depok

Pada tanggal: 30 Desember 2014

Yang menyatakan,



Jodhi Adhikaprana Sardjono

ABSTRAK

|  |  |  |
| --- | --- | --- |
| Nama | : | Jodhi Adhikaprana Sardjono |
| Program Studi | : | Teknik Komputer |
| Judul | : | Analisis Integrasi Color Code dengan Sistem Rumah Pintar Berbasis Sistem Operasi Android |

Sistem rumah pintar dengan sistem operasi Android yang diintegrasikan dengan fitur *Color Code* memiliki kelebihan-kelebihan yang dapat membantu memudahkan kehidupan manusia. Selain itu, dengan penelitian ini, diharapkan mampu pula untuk berkontribusi dalam pelestarian lingkungan. Dengan adanya alat ini, maka pengguna dapat mengaktifkan kamera pada telepon genggam berbasis android, lalu mengarahkan kamera dengan kode yang mengacu pada perangkat elektronik tertentu. Sehingga, perangkat elektronik tersebut dapat menyala atau mati sesuai dengan kondisi sebelumnya. Alat ini bekerja dengan IOIO-OTG, serta menggunakan OpenCV Library, dengan pengembangan algoritma dasar *color blob*. Pada penelitian ini, fokus kepada pengaplikasian kode warna yang diintegrasikan dengan perangkat-perangkat elektronik, yang dibatasi hingga empat jenis warna dan empat perangkat elektronik. Interval waktu antara input gambar dari kamera ponsel dan output berupa pemberian perintah kepada mikroprosesor untuk mengontrol perangkat elektronik yang diimplementasikan pada program ini adalah sebesar 0.892 detik. Sementara pengujian Error Rate pada Color Code bernilai 0%. Pengujian kuisioner berbentuk penilaian terhadap sistem yang terkait dengan efektivitas dari program Color Code untuk integrasi sistem rumah pintar menghasilkan kualitas 100% berhasil pada *System Testing*, 79% kepuasan pada *Consistency Testing*, serta 73% kepuasan pada *User Interface Testing*.

Kata kunci: *Color Code, Smart Home System*, *Android*, *Embedded System*

ABSTRACT

|  |  |  |
| --- | --- | --- |
| Name | : | Jodhi Adhikaprana Sardjono |
| Major | : | Teknik Komputer |
| Title | : | Analysis on Integrating Color Code with Smart Home System Based on Android Operating System |

The purpose of this research is to implement Color Code with Smart Home System with Android Operating System. Color Code, which is a new and innovative way to differentiate things, is visually appealing and is easily understood for the humans’ eyes. Eventually, this research’s expectation is designing a prototype that can help people on their daily tasks and also contributing in Go Green Campaign. The prototype utilizes IOIO On The Go as the processor, color blob algorithm as the basic formula of image processing, and four electrical gadgets with one color code each. Time interval program between image input from the camera and output instruction to control electrical gadget is 0.892 second. Meanwhile, Error Rate testing has a value of 0%. Questionnaire testing of the program according to its effectivity has a result of 100% succeed on System Testing, 79% satisfied on Consistency Testing, and 73% satisfied on User Interface Testing.

Keywords : *Color Code, Smart Home System*, *Android*, *Embedded System*

DAFTAR ISI

[HALAMAN PERNYATAAN ORISINALITAS ii](#_Toc408927579)

[LEMBAR PENGESAHAN iii](#_Toc408927580)

[KATA PENGANTAR iv](#_Toc408927581)

[HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS v](#_Toc408927582)

[ABSTRAK vi](#_Toc408927583)

[ABSTRACT vii](#_Toc408927584)

[DAFTAR ISI viii](#_Toc408927585)

[DAFTAR GAMBAR xi](#_Toc408927586)

[DAFTAR TABEL xiii](#_Toc408927587)

[BAB I](#_Toc408927588) [PENDAHULUAN 1](#_Toc408927589)

[**1.1 Latar Belakang** 1](#_Toc408927590)

[**1.2 Tujuan Penelitian** 3](#_Toc408927591)

[**1.3 Batasan Masalah** 3](#_Toc408927592)

[**1.4 Metode Penelitian** 3](#_Toc408927593)

[**1.5 Sistematika Penulisan** 4](#_Toc408927594)

[BAB II](#_Toc408927595) [PENGOLAHAN CITRA *COLOUR CODE* UNTUK TEKNOLOGI RUMAH PINTAR 5](#_Toc408927596)

[**BAB I** 5](#_Toc408927597)

[**2.1 Android OS** 5](#_Toc408927598)

[2.1.1 Versi Android [3] 6](#_Toc408927599)

[2.1.2 Aplikasi Penulisan Program Android 10](#_Toc408927600)

[2.1.3 Arsitektur Android 11](#_Toc408927601)

[2.1.4 Fundamental Aplikasi 13](#_Toc408927602)

[**2.2 Bahasa Pemrograman** 15](#_Toc408927603)

[2.2.1 Java [4] 15](#_Toc408927604)

[2.2.2 Library OpenCV 16](#_Toc408927605)

[2.2.3 Color Blob [5] 17](#_Toc408927606)

[**2.3 Divais** 20](#_Toc408927607)

[2.3.1 IOIO [6] 21](#_Toc408927608)

[2.3.2 Bluetooth Dongle [7] 21](#_Toc408927609)

[2.2.4 Light Dependent Resistor Light Intensity Sensor [7] 22](#_Toc408927610)

[2.2.5 LM35 Temperature Sensor [7] 23](#_Toc408927611)

[2.2.6 12VDC to 220VAC Relay [7] 24](#_Toc408927612)

[**2.4 Pengolahan Citra** 25](#_Toc408927613)

[BAB III](#_Toc408927614) [PERANCANGAN SISTEM UNTUK COLOUR CODE PADA TEKNOLOGI RUMAH PINTAR 29](#_Toc408927615)

[**3.1 *System Requirements*** 29](#_Toc408927616)

[3.1.1 *Use Case Diagram* 29](#_Toc408927617)

[**3.2 *System and Software Design*** 30](#_Toc408927618)

[3.2.1 *Algoritma Sistem* 30](#_Toc408927619)

[3.2.2 *Sequential Diagram* 33](#_Toc408927620)

[**3.3 Design Perangkat Keras *(Hardware)*** 35](#_Toc408927621)

[3.3.1 *Block Diagram* 35](#_Toc408927622)

[3.4 Skema Color Code 36](#_Toc408927623)

[BAB IV](#_Toc408927624) [ANALISIS IMPLEMENTASI INTEGRASI COLOR CODE DENGAN SISTEM RUMAH PINTAR 43](#_Toc408927625)

[**4.1 Tahap Implementasi Sistem** 43](#_Toc408927626)

[**4.2 Pengujian Interval Waktu Antara Input Gambar dari Kamera dan Output Pemberian Perintah Kepada Mikroprosesor untuk Mengontrol Perangkat Elektronik** 45](#_Toc408927627)

[**4.3 Pengujian Error Rate dalam Software** 48](#_Toc408927628)

[**4.4 Hasil Penilaian Sistem Oleh Pengguna** 51](#_Toc408927629)

[4.4.1 *System Testing* 52](#_Toc408927632)

[4.4.2 Consistency Testing 53](#_Toc408927633)

[4.4.3 *User Interface Testing* 55](#_Toc408927634)

[**4.5 Pengujian Mode Otomatis pada Lampu** 57](#_Toc408927635)

[**4.6 Pengujian Mode Otomatis pada Kipas** 59](#_Toc408927636)

[BAB V](#_Toc408927637) [SIMPULAN 61](#_Toc408927638)

[REFERENSI 62](#_Toc408927639)

[LAMPIRAN 1: Formulir Kuisioner 63](#_Toc408927640)

DAFTAR GAMBAR

Gambar 2.1 Struktur dan Konten OpenCV 17

Gambar 2.2 Color Blob Detection HSV 18

Gambar 2.3 OpenCV Threshold 19

Gambar 2.4 IOIO OTG 21

Gambar 2.5 Bluetooth Dongle 22

Gambar 2.6 LDR Light Intensity Sensor 23

Gambar 2.7 LM 35 Temperature Sensor 23

Gambar 2.8 Relay 24

Gambar 2.9 Matriks dari Citra Digital 26

Gambar 2.10 Citra Biner 27

Gambar 2.11 Citra Grayscale 27

Gambar 2.12 Citra Warna 28

Gambar 3.1 Use Case Diagram 29

Gambar 3.2 Flowchart Diagram 31

Gambar 3.3 Pseudocode Aktivitas Manual dan Otomatis 32

Gambar 3.4 Sequential Diagram 34

Gambar 3.5 Block Diagram 35

Gambar 3.6 Skema Colour Code 37

Gambar 3.7 Rancangan Color Code 38

Gambar 3.8 Warna Pada Colour Code 38

Gambar 3.9 Pseudocode Color Blob Setup dan Pemberi Flag 40

Gambar 3.10 Logo Shroom 41

Gambar 3.11 Aplikasi Mengakses Kamera 42

Gambar 4.1 Mesin Pengolah 43

Gambar 4.2 Rangkaian Sistem (1) 44

Gambar 4.3 Rangkaian Sistem (2) 44

Gambar 4.4 Ponsel Pintar untuk Menjalankan Aplikasi 45

Gambar 4.5 Pengujian Interval Waktu 47

Gambar 4.6 Nomor pengujian 1 Kombinasi Warna Hijau dan Biru 48

Gambar 4.7 Nomor pengujian 2 Kombinasi Warna Biru dan Kuning 49

Gambar 4.8 Nomor pengujian 3 Kombinasi Warna Kuning dan Hitam 49

Gambar 4.9 Nomor pengujian 4 Kombinasi Warna Hijau dan Kuning 49

Gambar 4.10 Pengujian Mode Otomatis pada Lampu 56

Gambar 4.11 Pengujian Mode Otomatis pada Kipas 58

DAFTAR TABEL

Tabel 4.1 Interval Waktu Antara Input dan Output 46

Tabel 4.2 Pengujian Error Rate dalam Sistem 48

Tabel 4.3 Hasil Penilaian System Testing 52

Tabel 4.4 Hasil Penilaian Consistency Testing (Profesional) 53

Tabel 4.5 Hasil Penilaian Consistency Testing (Mahasiswa) 53

Tabel 4.6 Hasil Penilaian UI Testing (Profesional) 54

Tabel 4.7 Hasil Penilaian UI Testing (Mahasiswa) 54

Tabel 4.8 Hasil Pengujian Mode Otomatis pada Lampu 55

Tabel 4.9 Hasil Pengujian Mode Otomatis pada Kipas 58

BAB I

PENDAHULUAN

Pada bab ini akan dijabarkan mengenai latar belakang, tujuan, batasan masalah, metode penulisan, serta sistematika penulisan dari penelitian ini.

## **1.1 Latar Belakang**

Pada zaman modern ini, kemajuan teknologi dapat mendukung pekerjaan sehari-hari masyarakat dunia pada berbagai macam aspek, seperti pada bidang edukasi, kesehatan, pangan, hiburan, energi, maupun pada bidang bisnis. Teknologi dikenal dapat mempermudah pekerjaan sehari-hari manusia. Fungsionalitas teknologi tidak hanya berkontribusi untuk itu, namun, teknologi yang selalu berkembang dan bertambah kualitasnya, dapat juga mengambil peran penting dalam pelestarian lingkungan (*Go Green Campaign*).

Semakin majunya industri properti di dunia, menyebabkan pentingnya pengguna properti untuk beradaptasi dengan perkembangan teknologi yang mendukung properti. Tidak hanya teknologi sebagai kebutuhan, namun juga teknologi sebagai elemen untuk mempermudah kehidupan manusia. Di samping itu, pengembang properti juga perlu mengimplementasikan konsep baru yang bisa meningkatkan daya jual properti mereka. Dengan mengaplikasikan teknologi hijau dan teknologi sebagai alat mempermudah pekerjaan manusia, dapat menarik lebih banyak perhatian dari calon-calon pembeli properti.

Penelitian ini mengangkat konsep teknologi sebagai alat untuk mempermudah pekerjaan manusia pada bidang properti. Diketahui bahwa semakin banyak pekerjaan manusia, akan semakin meningkat pula kebutuhan manusia akan sebuah alat praktis yang dapat menyederhanakan pekerjaan rutinitas manusia. Teknologi atau alat praktis tersebut mampu menghemat waktu manusia yang tidak ternilai. Itulah teknologi yang disebut sebagai teknologi rumah pintar.

Selanjutnya, teknologi pun menjadi kontributor dalam pelestarian lingkungan (*Go Green Campaign*). Sebagai contoh, orang-orang yang pergi jauh dalam jangka waktu yang lama dari rumah mereka, dengan kondisi tidak ada satupun orang di rumah tersebut, memiliki kecederungan untuk menyalakan seluruh lampu depan di rumah mereka, hingga mereka pulang ke kediaman mereka. Akibatnya, akan terjadi penggunaan listrik yang berlebihan dan tidak perlu, karena sesungguhnya lampu tidak perlu dinyalakan pada saat siang hari. Pada kasus tersebut, teknologi rumah pintar dapat membantu menjadi solusi atas sebuah permasalahan itu. Dengan teknologi rumah pintar, orang-orang tidak perlu menyalakan lampu depan rumah mereka sepanjang kepergian mereka. Ketika malam datang, sensor pada teknologi rumah pintar itu membantu mempermudah pekerjaan orang-orang tersebut sehingga pada malam hari (cahaya mencapai ukuran lux tertentu), lampu rumah akan menyala, dan ketika malam berganti siang (cahaya mencapai ukuran lux tertentu), lampu rumah akan mati. Contoh lain adalah dipasangnya sensor pintar untuk memonitor nyala/matinya kipas, sehingga pengatur suhu ruangan akan menyala ketika mencapai suhu tertentu, dan akan mati ketika mencapai suhu tertentu pula. Dengan demikian, terbukti bahwa teknologi ini dapat membantu untuk menghemat energi.

Penggunaan Android berdasarkan penelitian, di dunia sudah mencapai 1 miliar. Dengan semakin banyaknya pengguna Android di dunia, maka peluang untuk mengembangkan teknologi rumah pintar berbasis Android akan semakin terbuka lebar. Oleh karena itu, teknologi rumah pintar ini dikontrol dengan telepon genggam berbasis Android [1].

Maka, dapat disimpulkan bahwa pada era modern ini, masyarakat benar-benar membutuhkan teknologi, khususnya teknologi yang terintegrasi dengan rumah. Konsep teknologi untuk memudahkan kehidupan masyarakat dan aksi penghematan energi (pelestarian lingkungan), dapat diaplikasikan dengan sebuah teknologi rumah pintar.

Teknologi rumah pintar pada penelitian ini, akan ditambahkan kemampuan unik untuk meningkatkan gaya hidup manusia yaitu menggabungkan ilmu pengolahan citra (*image processing)*. Ilmu tersebut mempunyai kemampuan untuk mematikan atau menyalakan perangkat elektronik dengan mengaktifkan kamera mereka, melakukan scanning pada sebuah colour code. Dengan demikian *Color Code* tersebut akan mengacu pada perintah untuk mematikan atau menyalakan perangkat elektronik tertentu.

## **1.2 Tujuan Penelitian**

Tujuan penelitian ini adalah untuk membuat dan menganalisis teknologi rumah pintar yang terintegrasi dengan dua buah sensor yaitu sensor cahaya dan sensor temperatur, dengan kemampuan *Color Code,* yang dikontrol dengan telepon genggam bersistem operasi Android,sehingga dapat mempermudah hidup masyarakat dan berperan dalam pelestarian lingkungan..

## **1.3 Batasan Masalah**

Lingkup penelitian ini adalah menggunakan telepon genggam pintar Android, dengan *processing unit* utama IOIO, juga menggunakan dua buah sensor yaitu sensor temperatur dan sensor cahaya. Sedangkan kemampuan pengolahan citra pada penelitian ini menggunakan *library* Open CV dan terbatas pada empat warna pilihan dan empat perangkat elektronik

## **1.4 Metode Penelitian**

Metodologi penelitian yang digunakan adalah *Software Development Waterfall Model.*

1. *Requirements analysis and definition*: Mengumpulkan kebutuhan perlengkapan secara lengkap kemudian dianalisis dan didefinisikan kebutuhan yang harus dipenuhi oleh program yang akan dibangun. Fase ini dikerjakan secara penuh untuk bisa menghasilkan desain yang lengkap. Pada fase ke-satu ini, disusun sebuah *master plan*, mendefinisikan latar belakang dan tujuan, dan mendaftar alat dan bahan.

2. *System and software design*: Desain dikerjakan setelah kebutuhan selesai dikumpulkan secara lengkap. Desain yang dibutuhkan dalam tahap ini adalah design rangkaian elektronika, desain setiap *activity*, dan desain logika pemrograman.

3. *Implementation and unit testing*: Desain program diterjemahkan ke dalam kode-kode dengan menggunakan bahasa pemrograman yang sudah ditentukan. Program yang dibangun langsung diuji baik secara unit. Pada tahap ini, diterjemahkan kode-kode ke dalam bentuk *xml* (untuk GUI), dan juga menggunakan Java sebagai integrator gadget dengan sensor, serta menggunakan Open CV sebagai *library*. Setelah hal ini dilakukan, prototipe diuji per fungsi.

4. *Integration and system testing*: Penyatuan unit-unit program kemudian diuji secara keseluruhan (*system testing*). Tahap ini merupakan fase untuk menguji prototipe alat secara keseluruhan. Integrasi alat dan sistem seharusnya sudah berjalan dengan sempurna.

5. *System Analysis*: Tahap ini merupakan tahap/fase untuk memonitor, menganalisis, dan meneliti terkait dengan efektifitas dan efisiensi teknologi rumah pintar dengan kemampuan *Colour Code.* Pada fase ini diteliti apakah sistem tersebut mampu dengan jelas memberikan solusi-solusi atas permasalahan yang ada.

## **1.5 Sistematika Penulisan**

Bab 1: Pendahuluan. Bagian ini terdiri dari latar belakang, tujuan penelitian, batasan masalah, metode penelitian, dan sistematika penulisan.

Bab 2: Pengolahan Citra *Color Code* untuk Teknologi Rumah Pintar. Bagian ini akan membahas teori-teori dasar tentang teknologi rumah pintar, gambaran umum sistem rumah pintar dengan penggunaan sensor, dan gambaran umum *Color Code*.

Bab 3: Perancangan Sistem untuk *Color Code* pada Teknologi Rumah Pintar. Bagian ini membahas tentang perancangan *schematic hardware* atau design dari prototipe sistem rumah pintar dengan pengolahan citra.

Bab 4: Analisis Implementasi Integrasi *Color Code* pada Sistem Rumah Pintar. Pada bab ini akan dijelaskan mengenai analisis implementasi integrasi color code dengan sistem rumah pintar yang telah dibuat. Analisis tersebut berupa pengujian terhadap sistem yang telah dibuat dan juga penilaian terhadap keseluruhan fungsi.

Bab 5: Kesimpulan. Bab ini berisi kesimpulan terhadap seluruh penelitian yang dilakukan dan saran-saran untuk pengembangan selanjutnya.

BAB II

PENGOLAHAN CITRA *COLOUR CODE* UNTUK TEKNOLOGI RUMAH PINTAR

Bagian ini akan membahas teori-teori dasar tentang teknologi rumah pintar, yakni pemrograman dengan platform Android, gambaran umum sistem rumah pintar dengan penggunaan sensor, dan gambaran umum *Color Code.*



## **2.1 Android OS**

Android adalah sistem operasi yang berbasis linux atau *open source*. Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, *middleware*, dan aplikasi. Android menyediakan *platform* yang terbuka bagi para pengembang untuk menciptakan aplikasi mereka [2].

Android merupakan platform mobile pertama yang Lengkap, Terbuka, dan Bebas.

• Lengkap (*Complete Platform*): Para desainer dapat melakukan pendekatan yang komprehensif ketika mereka sedang mengembangkan platform Android. Android merupakan sistem operasi yang aman dan banyak menyediakan tools dalam membangun software dan memungkinkan untuk peluang pengembangan aplikasi.

• Terbuka (Open Source Platform): Platform Android disediakan melalui lisensi *open source*. Pengembang dapat dengan bebas untuk mengembangkan aplikasi. Android sendiri menggunakan Linux Kernel 2.6.

• *Free (Free Platform)*: Android adalah platform/aplikasi yang bebas untuk pengembang. Tidak ada lisensi atau biaya royalty untuk dikembangkan pada platform Android. Tidak ada biaya keanggotaan diperlukan. Tidak diperlukan biaya pengujian. Tidak ada kontrak yang diperlukan. Android dapat didistribusikan dan diperdagangkan dalam bentuk apapun.

Aplikasi Android dapat dikembangkan pada sistem operasi berikut:

• Windows

• Mac OS X

• Linux.

### Versi Android [3]

Beberapa versi Android yang pernah dirilis adalah sebagai berikut :

#### Android 3.x Honeycomb

Android 3.0 Honeycomb SDK dirilis pada tanggal 22 Februari 2011. Ini adalah versi tablet Android dan tersedia bersamaan dengan Android Gingerbread untuk ponsel.

Android Honeycomb memperkenalkan interface user dengan semua sentuhan baru, yang tidak bergantung pada setiap tombol navigasi fisik. Sayangnya, itu terganggu oleh masalah performa produknya.

Selain perbedaan visual, Honeycomb untuk pertama kali memperkenalkan akselerasi hardware asli dan dukungan untuk prosesor *multi-core*. Honeycomb menawarkan dukungan untuk panggilan video melalui Google Talk bahkan sebelum smartphone Gingerbread lakukan (fitur ini hanya dirilis di Android 2.3.4).

Kemudian Honeycomb telah diupdate ke Android 3.1 dan 3.2 Gingerbread dan Honeycomb sebelum keduanya diganti dengan Android 4.0 Ice Cream Sandwich.

Android 3.1 menambahkan dukungan untuk aksesori USB seperti keyboard eksternal dan perangkat menunjuk, joystick dan gamepads. Android 3.2 membawa perbaikan dalam kinerja melalui optimasi untuk yang lebih luas dari perangkat tablet.

Rilis Android 3.x Honeycomb tidak disertai dengan perangkat Nexus yang baik, namun Google banyak digunakan Motorola Xoom sebagai duta dan platform demonstrasi utama untuk Honeycomb.

#### Android 4.0.x Ice Cream Sandwich

Android 4.0.x Ice Cream Sandwich SDK dirilis pada tanggal 19 Oktober 2011. Android ICS dirilis bersama dengan *Google Galaxy Nexus* (November 2011) – penerus berikutnya dalam populer Nexus lineup.

Android *Ice Cream Sandwich* atau yang lebih dikenal dengan ICS, terintegrasi cabang telepon dan tablet Android menjadi edisi tunggal dengan antarmuka yang umum, yang lagi-lagi tidak mengandalkan tombol navigasi fisik.

Salah satu perubahan yang paling menonjol di UI adalah bahwa Widget kini pindah ke tab mereka sendiri dalam menu utama. Banyak perubahan lain diperkenalkan juga, seperti jenis huruf baru di seluruh UI – Roboto, kemampuan untuk meluncurkan aplikasi dari lockscreen, dukungan untuk real-time pidato ke teks dikte, Face Unlock, web browser Google Chrome datang preinstlalled dengan tab dan sinkronisasi bookmark, kamera dengan selang waktu dan panorama dan editor foto built-in, analisis data penggunaan, kalender baru dan aplikasi mail. Grafis UI dan animasi mendapat akselerasi hardware dan Wi-Fi Direct fungsi yang sekarang didukung secara native.

Revisi kecil yang kemudian dirilis sebagai update adalah Android 4.0.1 dan 4.0.2.Pada Desember 2011 Android 4.0.3 dirilis untuk memperbaiki bug yang paling terlihat di rilis sebelumnya ditambah dengan menambahkan stabilisasi video yang ditingkatkan.

Android 4.0.4 adalah update minor, yang selanjutnya meningkatkan kinerja OS. Revisi kecil lebih lanjut kemudian dirilis sebagai update Android 4.0.4 mulai dari ke 4.0.5 dan 4.0.6.

#### Android Jelly Bean 4.x

Android 4.1 Jelly Bean SDK dirilis pada tanggal 9 Juli 2012. Sebuah versi baru 4.2 dirilis bersama Jelly Bean moniker.

Android 4.1 Jelly Bean memiliki fokus pada kinerja, yang bertujuan untuk menyingkirkan gagap dengan memperkenalkan *Vsync* (sinkronisasi vertikal) dari waktu *output* *frame* dan *refresh rate* layar itu. Juga, Jelly Bean menambahkan tiga dukungan di grafis.

Alih-alih smartphone, Google memilih untuk menampilkan tablet beserta Android 4.1 Jelly Bean rilis – Google Nexus 7 (Juli 2012) dimaksudkan untuk menunjukkan bahwa Jelly Bean tampak sama baik pada layar tablet.

Selain kinerja, ada fitur menarik lainnya seperti, pengakuan secara *offline* suara dan dikte, pemberitahuan diperluas, output audio USB dan output audio multichannel HDMI, enkripsi dan update Smart App. Font Roboto juga baru, widget sekarang resizable dinamis, dan prediksi kata telah diperbarui juga. Google Now juga merupakan kesepakatan yang cukup besar – ini adalah proses berbagai data tentang lokasi Anda, waktu hari, jadwal Anda, dll – dan itu menunjukkan banyak informasi, yang penting untuk saat ini. Hal ini juga mendukung fungsi suara bahasa alami pencarian lanjutan, yang cukup fleksibel.

Revisi kecil dari 4.1 Jelly bean kemudian dirilis sebagai update seperti Android 4.1.1 dan 4.1.2.

Di atas mereka, Android 4.2 menambahkan beberapa account pengguna, pengaturan *shortcut* asli tepat di panel pemberitahuan, modus kamera Photo Sphere, gerakan mengetik, layar streaming nirkabel (atas protokol Miracast), dinamis “Daydream” *screensaver*, *widget* di *lockscreen* dan selanjutnya peningkatan stabilitas.

Sebuah revisi kecil dari Android 4.2 kemudian dirilis sebagai update – Android 4.2.1. Yang paling penting itu menambahkan dukungan asli untuk Bluetooth gamepads dan joystick.

Pelepasan Android 4.2 didukung dengan pengumuman smartphone Nexus keempat – LG buatan Google Nexus 4 (November 2012). Tidak hanya itu, tapi Google juga meluncurkan tablet 10 inci – Samsung buatan Google Nexus 10.

#### Android Kitkat

Android kitkat dirilis pada tanggal 31 Oktober 2013, dengan berbagai macam fitur-fitur terbarunya. Android kitkat 4.4 menggunakan API 19.

1. Perintah Suara. pada versi terbaru Andorid ini jika pengguna ingin melakukan search, mereka tidak harus menyentuh ponsel, mengirim pesan, atau menerima arahan dari Google Map. Pada menu utama, pengguna dapat mengatakan ' OK Google ' untuk memberikan perintah dengan kerja spesifik yang presisi.

2. *Multi Tasking* Lebih Cepat. Versi Android 4.4 kitkat mengembangkan kemampuan ponselnya agar lebih cepat dalam memberikan respons.

3. Visual yang Lebih Baik. Pada versi 4.4, tampilan yang terkesan menghalangi pada versi android sebelumnya, (sistem operasi Android menampilkan status bar, tombol navigasi, pada saat menonton video atau membaca ebook) bisa dihilangkan.

4. *Photo Editor*. *Photo Editor* pada Android KitKat diperbarui. Terdapat beberapa fitur efek, filter, dan tools untuk memperbarui gambar. Photo Editor juga memiliki kemampuan untuk mengembalikan hasil editan foto ke bentuk semula.

5. *Hangouts*. Aplikasi *Hangouts* juga ditambahkan pada sistem operasi ini, yang mampu mengelompokkan semua SMS, MMS, video call, dan chatting dengan *Google Hangouts* pada satu tempat.

#### Android Lollipop

Android Lollipop 5.0 dirilis pada bulan Oktober 2014. Merupakan versi android terbaru yang dapat berjalan tidak hanya pada tablet dan ponsel, namun dapat juga pada jam tangan, televisi, dan bahkan pada mobil. Seluruh foto-foto, lagu, aplikasi, bahkan pencarian terbaru dapat diakses melalui seluruh perangkat Android. Selain itu, versi Lolliop juga memperbarui dari segi interaksi dengan pengguna.

### Aplikasi Penulisan Program Android

Mengenal SDK, AVD dan ECLIPSE dilakukan sebelum memulai ke dalam penulisan program. Harus diketahui aplikasi-aplikasi apa saja yang dibutuhkan, untuk melakukan penulisan program Android. Perangkat atau aplikasi yang dibutuhkan adalah Android SDK, AVD (*Android Virtual Device*) dan Eclipse. Berikut adalah pembahasan dari masing-masing perangkat-perangkat tersebut.

#### Android SDK (Software Development Kit)

Android SDK adalah tools API (Application Programming Interface) yang diperlukan untuk mulai mengembangkan aplikasi pada platform Android menggunakan Bahasa pemrograman Java. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware*, dan aplikasi kunci yang di release oleh Google. Saat ini disediakan Android SDK (*Software Development Kit*) sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada platform Android menggunakan Bahasa pemrograman Java. Sebagai platform aplikasi-netral, Android memberi Anda kesempatan untuk membuat Aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan Handphone/Smartphone. Beberapa fitur-fitur Android yang paling penting adalah:

• *Framework* Aplikasi yang mendukung penggantian komponen dan reusable.

• Mesin Virtual Dalvik dioptimalkan untuk perangkat mobile.

• Integrated browser berdasarkan engine open source WebKit.

• Grafis yang dioptimalkan dan didukung oleh libraries grafis 2D, grafis 3D berdasarkan spesifikasi openGL ES 1.0 (Opsional akselerasi hardware).

• SQLite untuk penyimpanan data (database).

• Media support yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), GSM Telephony (tergantung hardware).

• Bluetooth, EDGE, 3G, dan WiFi (tergantung hardware).

• Kamera, GPS, kompas, dan accelerometer (tergantung hardware).

Lingkungan Development yang lengkap dan kaya termasuk perangkat emulator, tools untuk debugging, profil dan kinerja memori, dan plugin untuk IDE Eclipse.

#### AVD

AVD kependekan dari *Android Virtual Device* yaitu sebuah emulator untuk menjalankan virtual Android. Jadi tanpa menggunakan atau mempunyai android phone pun, android dapat dirasakan/digunakan, termasuk program android itu sendiri.

#### Eclipse

Eclipse adalah IDE untuk pengembangan java/android yang free dan dapat di download pada situs resminya. Eclipse yang bisa digunakan adalah Eclipse yang mendukung pengembangan pemrograman berbasis Java. Versi yang direkomendasikan adalah Eclipse versi 3.5 Galileo atau versi 3.4 Ganymade. Hal ini karena terdapat sedikit masalah dengan Eclipse 3.6 Helios, walaupun ada beberapa pengembang yang pernah mencoba menggunakan Helios dan dapat berjalan dengan baik untuk Android.

### Arsitektur Android

Secara garis besar Arsitektur Android dapat dijelaskan dan digambarkan sebagai berikut:

#### Applications dan Widgets

*Applications* dan *Widgets* ini adalah layer di mana kita berhubungan dengan aplikasi saja, di mana biasanya kita download aplikasi kemudian kita lakukan instalasi dan jalankan aplikasi tersebut. Di *layer* terdapat aplikasi inti termasuk klien email, program SMS, kalender, peta, browser, kontak, dan lain-lain. Semua aplikasi ditulis menggunakan Bahasa pemrograman Java.

#### Applications Frameworks

Android adalah “*Open Development Platform*” yaitu Android menawarkan kepada pengembang atau memberi kemampuan kepada pengembang untuk membangun aplikasi yang bagus dan inovatif. Pengembang bebas untuk mengakses perangkat keras, akses informasi resources, menjalankan service background, mengatur alarm, dan menambahkan status notifications, dan sebagainya. Pengembang memiliki akses penuh menuju API framework seperti yang dilakukan oleh aplikasi yang kategori inti. Arsitektur aplikasi dirancang supaya kita dengan mudah dapat menggunakan kembali komponen yang sudah digunakan (reuse).

Sehingga bisa disimpulkan Applications Frameworks ini adalah layer dimana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem operasi Android, karena pada layer inilah aplikasi dapat dirancang dan dibuat, seperti content-providers yang berupa sms dan panggilan telepon.

Komponen-komponen yang termasuk di dalam *Applications Frameworks* adalah sebagai berikut:

• *Views*

• *Content Provider*

• *Resource Manager*

• *Notification Manager*

• *Activity Manager*

#### Libraries

Libraries ini adalah layer di mana fitur-fitur Android berada, biasanya para pembuat aplikasi mengakses libraries untuk menjalankan aplikasinya. Berjalan di atas kernel, Layer ini meliputi library C/C++ inti seperti Libc dan SSL, serta :

• Libraries media untuk pemutaran media audio dan video.

• Libraries untuk manajemen tampilan.

• Libraries Graphics mencakup SGL dan OpenGL untuk grafis 2D dan 3D.

• Libraries SQLite untuk dukungan database.

• Libraries SSL dan WebKit terintegrasi dengan web browser dan security.

• Libraries LiveWebcore mencakup modern web browser dengan engine embedded web view.

• Libraries 3D yang mencakup implementasi OpenGL ES 1.0 API’s.

#### Android Run Time

Layer yang membuat aplikasi Android dapat dijalankan di mana dalam prosesnya menggunakan Implementasi Linux. *Dalvik Virtual Machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi Android. Di dalam Android Run Time dibagi menjadi dua bagian yaitu:

• *Core Libraries* : Aplikasi Android dibangun dalam Bahasa java, sementara Dalvik sebagai virtual mesinnya bukan Virtual Machine Java, sehingga diperlukan sebuah libraries yang berfungsi untuk menterjemahkan Bahasa Java/C yang ditangani oleh *Core Libraries*.

• *Dalvik Virtual Machine*: Virtual mesin berbasis register yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien, dimana merupakan pengembangan yang mampu membuat linux kernel untuk melakukan threading dan manajemen tingkat rendah.

#### Linux Kernel

Linux kernel adalah layer dimana inti dari sistem operasi dari Android itu berada. Berisi file-file system yang mengatur *sistem processing*, *memory*, *resource*, *drivers*, dan sistem-sistem operasi Android lainnya. Linux kernel yang digunakan android adlaah linux kernel release 2.6, jika Anda ingin mempelajari linux kernel ini, anda dapat mengunduh linux kernel secara gratis di situs www.kernel.org.

### Fundamental Aplikasi

Aplikasi Android ditulis dalam Bahasa pemrograman java. Kode java dikompilasi bersama dengan data file resource yang dibutuhkan oleh aplikasi, di mana prosesnya di-package oleh tools yang dinamakan “apt tools” ke dalam paket Android sehingga menghasilkan file dengan ekstensi *apk*. File .*apk* itulah yang disebut dengan aplikasi, dan nantinya dapat di install di perangkat mobile.

Terdapat empat jenis komponen pada aplikasi Android, yaitu:

#### Activities

Suatu activity akan menyajikan user interface (UI) kepada pengguna, sehingga pengguna dapat melakukan interaksi. Sebuah aplikasi android bisa jadi hanya memiliki satu activity, tetapi umumnya aplikasi memiliki banyak activity tergantung pada tujuan aplikasi dan desain dari aplikasi tersebut. Satu activity biasanya akan dipakai untuk menampilkan aplikasi atau yang bertindak sebagai user interface (UI) saat aplikasi diperlihatkan kepada user. Untuk pindah dari satu activity ke activity lain kita dapat melakukannya dengan satu *event*, misalnya click tombol, memilih opsi atau menggunakan triggers tertentu. Secara hirarki sebuah *windows activity* dinyatakan dengan *method* Activity.setContentView(). *ContentView* adalah objek yang berada pada root hirarki.

#### Service

*Service* tidak memiliki *Graphic User Interface* (GUI), tetapi *service* berjalan secara background, sebagai contoh dalam memainkan music, *service* mungkin memainkan musik atau mengambil data dari jaringan, tetapi setiap *service* harus berada dalam kelas induknya. Misalnya, media player sedang memutar lagu dari list yang ada, aplikasi ini akan memiliki dua atau lebih activity yang memungkinkan user untuk memilih lagu misalnya, atau menulis sms sambal player sedang jalan. Untuk menjaga music tetap di jalankan, *activity player* dapat menjalankan *service*. *Service* dijalankan pada thread utama dari proses aplikasi.

#### Broadcast Receiver

*Broadcast receiver* berfungsi menerima dan bereaksi untuk menyampaikan notifikasi. Contoh broadcast seperti notifikasi zona waktu berubah, baterai low, gambar telah seslai diambil oleh camera, atau pengubahan referensi Bahasa yang digunakan. Aplikasi juga dapat menginisiasi broadcast misalnya memberikan informasi pada aplikasi lain bahwa ada data yang telah diunduh ke perangkat dan siap untuk digunakan.

*Broadcast receiver* tidak memiliki *user interface* (UI), tetapi memiliki sebuah *activity* untuk merespon informasi yang mereka terima, atau mungkin menggunakan *Notification Manager* untuk memberitahu kepada pengguna, seperti lampu latar atau *vibrating* (getaran) perangkat, dan lain sebagainya.

#### Content Provider

*Content provider* membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. Data disimpan dalam file sistem seperti database SQLite. Content provide menyediakan cara untuk mengakses data yang dibutuhkan oleh suatu activity, misalnya ketika kita menggunakan aplikasi yang membutuhkan peta (*Map*), atau aplikasi yang membutuhkan untuk mengakses data kontak dan navigasi, maka disinilah fungsi *content provider*.

## **2.2 Bahasa Pemrograman**

Digunakan pemrograman Java dan XML untuk membentuk aplikasi Android ini. Java akan mengkonfigurasi fungsi-fungsi aplikasi, dimana XML akan memberikan tampilan pada aplikasinya. Terdapat 3 aktifitas dan 2 tampilan pada aplikasi ini. Aktifitas pertama, adalah IOIOSimpleApp.java dan main.xml yang akan menjadi aplikasi tampilan awal, dan aktifitas berikutnya adalah menu aplikasi yang terdiri dari MenuAct.java dan activity\_menu.xml. Aktifitas-aktifitas tersebut harus dibuat pada manifestasi aplikasi (IOIOSimpleApp manifest). Eclipse digunakan untuk membuat aplikasi dan juga mengkombinasikan kode Android umum dengan kode IOIO sesuai dengan ‘worksheet’.

### Java [4]

Java adalah salah satu bahasa pemrograman komputer yang memungkinkan *programmer* (orang yang membuat program) dapat memberi instruksi pada komputer. Java juga dapat didefinisikan sebagai suatu nama sekumpulan teknologi yang berfungsi untuk membuat dan menjalankan perangkat lunak pada komputer. Kebanyakan dari *programmer* lebih suka menyebut Java sebagai sebuah teknologi dibandingkan dengan hanya sebuah bahasa pemrograman, karena Java itu lebih lengkap dibandingkan dengan semua bahasa pemrograman konvensional.

### Library OpenCV

OpenCV (Open Computer Vision) adalah sebuah API (*Application Programming Interface*) Library yang sudah sangat familiar pada Pengolahan Citra *Computer Vision*. Computer Vision itu sendiri adalah salah satu cabang dari Bidang Ilmu Pengolahan Citra (*Image Processing*) yang memungkinkan komputer dapat melihat seperti manusia. Dengan vision tersebut komputer dapat mengambil keputusan, melakukan aksi, dan mengenali terhadap suatu objek. Beberapa pengimplementasian dari *Computer Vision* adalah *Face Recognition*, *Face Detection, Face/Object Tracking, Road Tracking*, dll. OpenCV adalah library *Open Source* untuk Computer Vision untuk C/C++, OpenCV didesain untuk aplikasi real-time, memiliki fungsi-fungsi akuisisi yang baik untuk image/video.

OpenCV (*Open Source Computer Vision*) adalah library yang dulunya dikembangkan oleh Intel dan sekarang didukung oleh Willow Garage yang bermain pada bidang real-time image processing. OpenCV adalah gratis untuk digunakan dibawah lisensi *Open Source BSD*.

Library OpenCV dapat bekerja secara cross-platform. OpenCV ditulis dalam bahasa C, sehingga mudah dikembangkan ke berbagai platform. Maka dari itu OpenCV banyak dikembangkan supaya mendukung bahasa lain seperti C#, Phyton, Ruby, dan Java (menggunakan JavaCV).

Semenjak hadirnya OpenCV versi 2.0 (sekarang sudah sampai versi 2.1), OpenCV selain menggunakan bahasa C, juga menggunakan bahasa C++. Hal ini bermanfaat dalam mengurangi banyak kesalahan pemrograman ketika menggunakan OpenCV di C. Namun C++ sulit dikembangkan untuk mendukung bahasa lain sehingga banyak bahasa yang tidak terlalu cocok dengan OpenCV 2.0.

OpenCV sendiri terdiri dari 5 library, yaitu :

• *CV* : untuk algoritma Image processing dan Vision.

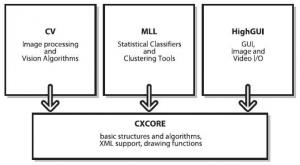
• *ML* : untuk machine learning library

• *Highgui* : untuk GUI, Image dan Video I/O.

• *CXCORE* : untuk struktur data, support XML dan fungsi-fungsi grafis.

• *CvAux*

Struktur dan Konten OpenCV dapat dilihat pada Gambar 2.1:

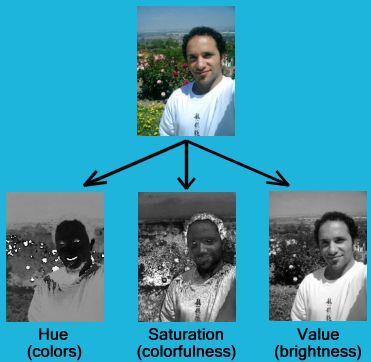


Gambar 2.1 Struktur dan Konten OpenCV

### Color Blob [5]

Blob detection adalah sebuah metode yang simpel dan cepat yang dapat digunakan untuk tugas-tugas pada bidang *machine vision*, seperti melacak bola berwarna merah, mencari warna biru, atau mendeteksi kulit seseorang. Deteksi kulit sangat berguna untuk melakukan deteksi wajah dan pengenalan wajah, juga untuk pengenalan gestur tangan.

Untuk menemukan *colored blobs* (tanda gumpalan warna), hal yang harus dilakukan adalah merubah warna gambar dari format BGR menjadi HSV sehingga warna dapat lebih mudah untuk dibedakan. Dapat juga digunakan program “ColorWheelHSV”, untuk melihat nilai mana (H/S/V) yang ingin di-threshold. Software grafik lain tidak dapat digunakan untuk melihat ambang HSV yang mana yang ingin digunakan karena nilai HSV pada OpenCV berbeda dengan software lainnya. Sebagai contoh, jika mendeteksi sebuah bola biru, dapat dilihat bahwa warna biru memiliki Hues antara 85 hingga 130 pada OpenCV (jika dijalankan program ColorWheelHSV).



Gambar 2.2 Color Blob Detection HSV [5]

Gambar 2.2 diatas merupakan sebuah foto RGB yang dipisahkan menjadi komponen *Hue, Saturation,* dan *Value* (HSV).

Lalu, dapat digunakan “cvThreshold()” pada gambar Hue untuk mendapatkan pixel berwarna biru. Namun karena hitam, putih, dan abu-abu juga memiliki nilai Hue, maka komponen *Saturation* dan *Value* juga perlu dilakukan *threshold* pada gambar HSV agar warna hitam, putih, dan abu-abu diabaikan. Objek dan latar belakang dapat diatur sesuai dengan ekspektasi, sebagai contoh sebuah gambar dengan *Value* (brightness) di bawah 60 dapat membuat gambar menjadi terlalu gelap (hitam), dan gambar dengan *Saturation* di bawah 30 dapat menjadi terlalu kusam. Dan setelah program tersebut berjalan, ambang HSV dapat diatur sesuai dengan keinginan sehingga hasil gambar dapat memenuhi ekspektasi, karena variasi tersebut dapat berjumlah banyak, bahkan dapat digunakan rumus yang lebih rumit seperti mengabaikan sesuatu dengan *Value* yang lebih dari 200, jika memiliki Saturation di bawah 50, dan rumus-rumus lainnya.

Satu trik tersendirinya adalah pada OpenCV, warna merah dapat berada di *Hues* dengan nilai antara 0 hingga 10 atau dapat juga bernilai antara 175 hingga 179, karena komponen Hue pada OpenCV berulang seperti sebuah lingkaran pada nilai 180, sehingga Hue yang bernilai 179 hampir identik/sama dengan Hue bernilai 0.

Setelah mengetahui ambang batas HSV yang digunakan sesuai dengan keinginan, maka program dapat ditulis dengan tahap :

• Mengkonversi sebuah gambar berwarna dengan format BGR menjadi HSV dengan menggunakan “cvConvert()”

• Memisahkan gambar berwarna HSV dengan masing-masing komponen *Hues, Saturation,* dan *Value*.

• Menggunakan “cvThreshold()” untuk melihat pixel-pixel yang berada pada range (jangkauan) HSV yang sesuai.

• Menggunakan salah satu dari berbagai *blob libraries* untuk mendeteksi blob yang ada pada Gambar 2.3 dengan threshold sehingga posisi dan ukuran dapat diketahui, dan blobs dapat dideteksi.



Gambar 2.3 OpenCV Threshold

Teknik image noise filtering juga hampir selalu dibutuhkan untuk mengurangi pixel-pixel yang mengganggu (noisy pixels), seperti dengan kombinasi dari cvSmooth(), cvErode(), cvDilate(), dan sebagainya. Namun, tentu saja masing-masing filter tersebut digunakan sesuai dengan kebutuhan proyek. Buku OpenCV resmi “Learning OpenCV: Computer Vision with the OpenCV Library” menjelaskan beberapa cara untuk melakukan filter gambar (*image filtering*).

## **2.3 Divais**

Berikut ini adalah perangkat hardware yang digunakan dalam penelitian ini:

• Adapter AC to DC 5 Volt

• Motor DC

• Lampu

• Kipas

• Terminal

• Charger laptop

• Resistor 68

• Resistor 470K; 4,7K; 3,3K

• NPN Transistor

• Standard 3-Pin-Header and 4-Pin-Header

• T-Block Header

• LM35 (temperature sensor)

• LM324 (OpAmp)

• Capacitor 100nF

• Diode 1N4002

Devais-devais utama yang digunakan di dalam penelitian ini adalah:

### 2.3.1 IOIO [6]



Gambar 2.4 IOIO OTG

IOIO-OTG yang terdapat pada Gambar 2.4 merupakan board I/O yang dapat bertindak sebagai USB Host ataupun USB Device. Jika IOIO-OTG dihubungkan ke device Android dan aplikasi pendukung dijalankan, maka IOIO-OTG akan bertindak sebagai USB Host. Jika IOIO-OTG dihubungkan ke perangkat komputer (OS Windows, Linux, OSX) maka IOIO-OTG akan bertindak sebagai USB Device. Ketika menjadi IOIO-OTG menjadi USB Device, maka akan terdeteksi sebagai virtual serial port.

Spesifikasi :

• Berbasiskan mikrokontroler PIC 24FJ256

• Catu daya : 5VDC - 15VDC

• Memiliki output ADC, Digital I/O, PWM Output, UART, SPI, I2C, dll

### 2.3.2 Bluetooth Dongle [7]

Pengertian dari *dongle* menunjuk kepada semua perangkat yang ditancapkan ke komputer, terutama perangkat yang menggantung pada komputer laptop. Bluetooth dongle berarti perangkat yang menyediakan koneksi Bluetooth untuk computer atau untuk mikroprosesor. Bluetooth dongle memiliki bentuk yang hampir sama dengan USB Flash Disk.

Sebuah Bluetooth dongle USB memiliki rentang 10 m. Teknologi ini berguna ketika mentransfer informasi antara dua atau lebih perangkat yang berdekatan di situasi rendah bandwidth. Bluetooth biasanya digunakan untuk mentransfer data suara dengan telepon (yaitu, dengan headset Bluetooth) atau data byte dengan komputer genggam (mentransfer file).



Gambar 2.5 Bluetooth Dongle

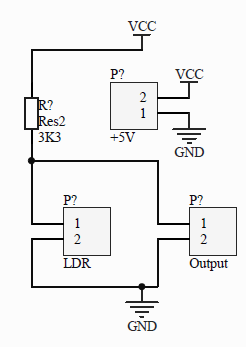
### Light Dependent Resistor Light Intensity Sensor [7]

*Light Intensity Sensor* adalah sebuah modul sensor intensitas cahaya dengan menggunakan LDR (*Light Dependent Resistor*)­. Terkoneksi dengan dua standard 2-pin header masing-masing dengan VCC dan GND dan V output. Rumus yang digunakan untuk rangkaian LDR adalah seperti pada rumus 2.1

Dimana :

* *RL = 500 / Lux K.Ohm*
* *Vo = 5\*RL / (RL+3.3)*
* *Lux = (2500/Vo – 500) / 3.3* (2.1)

Rangkaian Light Dependent Resistor dapat dilihat pada Gambar 2.6.



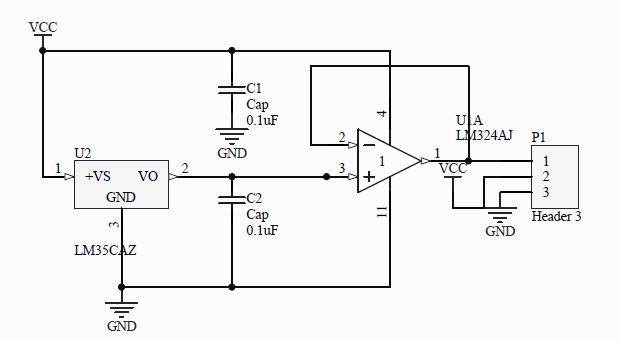
Gambar 2.6 LDR Light Intensity Sensor

### LM35 Temperature Sensor [7]

Adalah sebuah modul sensor untuk mengukur temperatur sebuah ruangan dengan menggunakan IC LM35. Dengan Operation Amplifier untuk menstabilkan sinyal output. Sensor ini terkoneksi dengan standard 3-pin header dengan konfigurasi sebagai berikut:

1. Sinyal Output
2. VCC
3. GND

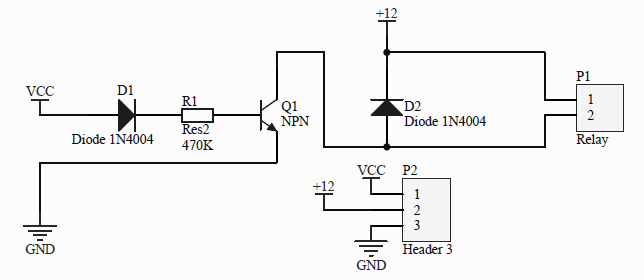
Rangkaian LM35 Temperature Sensor dapat dilihat pada Gambar 2.7.

****

Gambar 2.7 LM 35 Temperatur Sensor

### 12VDC to 220VAC Relay [7]

Modul control relay 220VAC dengan sinyal input 5V yang mengontrol sebuah sinyal 12V yang terkoneksi ke sebuah Relay 220VAC dengan 2-Pin T-Block Header. Input control dengan standard 3-pin signal (VCC 5V, +12V, GND). Rangkaian 12VDC to 220VAC Relay dapat dilihat pada Gambar 2.8.



Gambar 2.8 12VDC to 220VAC Relay

## **2.4 Pengolahan Citra**

Sistem warna pada citra di landasi oleh teori fisika tentang cahaya dan panjang gelombang cahaya serta warna yang di bawa. Dalam ilmu fisika di sampaikan bahwa cahaya adalah gelombang yang memiliki panjang gelombang berbeda beda. Berdasarkan panjang gelombangnya cahaya dibagi menjadi cahaya tampak dan cahaya tidak tampak. cahaya tampak berada pada kisaran panjang gelombang 400 nm sampai dengan 700 nm. di bawah 400 nm merupakan gelombang tidak tampak – ultra violet dan diatas 700 nm merupakan gelombang tidak tampak infra merah.

Pada gelombang tampak, rentang terbesar ada pada gelombang merah, hijau dan biru. Peneliti sepakat bahwa warna warna lain data diperoleh dengan pencampuran ketiga warna tersebut dengan proporsi tertentu. pengkodean warna dengan meotde ini di sebut sistem warna R(red), G(green) dan B(blue).

Selain memisahkan warna menjadi 3 komponen warna pokok (RGB) warna juga dapat di wakili dengan Intensitas (tingkat ke -terangan), Hue (warna itu sendiri) dan saturation (kedalaman).

Iluminasi/intensitas merupakan power/kekuatan cahaya yang diterima dari gelap atau 0 sampai terang, tanpa perduli warna apa yang di pancarkan. Sedangkan hue adalah warna asli dari cahaya, tanpa perduli kekuatan cahayanya. Sedangkan saturation adalah banyaknya warna putih yang dicampurkan dengan hue. Misalkan kita membedakan warna merah tua dengan warna merah muda, pada dasarnya merah, hanya untuk merah tua kandungan warna putihnya = 0 sedangkan pada merah muda kandungan warna putih/saturasinya lebih tinggi.

Pengkodean warna seperti ini disebut HIS (Ilumination, Hue, Saturation) atau HSV (Hue, Satutation, and Value/Intensitas). Model warna HSV ini lebih cocok dengan persepsi warna yang dialami manusia.

Citra digital adalah sebuah fungsi 2D, f(x,y), yang merupakan fungsi intensitas cahaya, dimana nilai x dan y merupakan koordinat spasial dan nilai fungsi di setiap titik (x,y) merupakan tingkat keabuan citra pada titik tersebut. Citra digital dinyatakan dengan sebuah matriks dimana baris dan kolomnya menyatakan suatu titik pada citra tersebut dan elemen matriksnya (yang disebut sebagai elemen gambar atau piksel) menyatakan tingkat keabuan pada titik tersebut. Matriks dari citra digital berukuran NxM (tinggi x lebar), dimana:

N = jumlah baris 0 < y ≤ N – 1

M = jumlah kolom 0 ≤ x ≤ M – 1

L = derajat keabuan 0 ≤ f(x,y) ≤ L – 1

Gambar 2.9 berikut ini adalah gambaran matriks dari citra digital:

|  |
| --- |
|  |

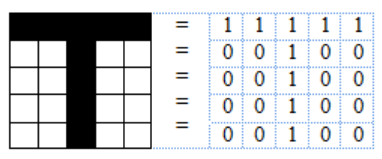
Gambar 2.9 Matriks dari Citra Digital

Dimana indeks baris (x) dan indeks kolom (y) menyatakan suatu koordinat titik pada citra, sedangkan f(x,y) merupakan intensitas (derajat keabuan) pada titik (x,y).

Berdasarkan jenisnya, citra digital dapat dibagi menjadi 3 (Sutoyo, 2009), yaitu:

1. Citra Biner (Monokrom)

Memiliki 2 buah warna, yaitu hitam dan putih. Warna hitam bernilai 1 dan warna putih bernilai 0. Untuk menyimpan kedua warna ini dibutuhkan 1 bit di memori. Contoh dari susunan piksel pada citra monokrom adalah sebagai berikut:



Gambar 2.10 Citra Biner

2. Citra Grayscale (skala keabuan)

Citra grayscale mempunyai kemungkinan warna hitam untuk nilai minimal dan warna putih untuk nilai maksimal. Banyaknya warna tergantung pada jumlah bit yang disediakan di memori untuk menampung kebutuhan warna tersebut. Semakin besar jumlah bit warna yang disediakan di memori, maka semakin halus gradasi warna yang terbentuk. Contoh:

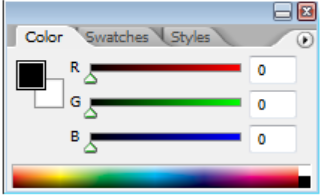


Gambar 2.11 Citra Grayscale

Skala keabuan 2 bit… jumlah kemungkinan 22 = 4 warna. Jadi, kemungkinan warna 0 (minimal) sampai 4 (maksimal).

3. Citra Warna (true color)

Setiap piksel pada citra warna mewakili warna yang merupakan kombinasi tiga warna dasar, yaitu merah, hijau, dan biru (RGB = Red, Green, Blue). Setiap warna dasar menggunakan penyimpanan 8 bit = 1 byte (nilai maksimum 255 warna), jadi satu piksel pada citra warna diwakili oleh 3byte.



Gambar 2.12 Citra Warna

Pengolahan citra digital adalah salah satu bentuk pemrosesan informasi dengan inputan berupa citra (image) dan keluaran yang juga berupa citra atau dapat juga bagian dari citra tersebut. Tujuan dari pemrosesan ini adalah memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin computer. Operasi-operasi pada pengolahan citra digital secara umum dapat diklasifikasikan sebagai berikut:

1. Perbaikan kualitas citra (image enhancement), contohnya perbaikan kontras gelap/terang, penajaman (sharpening), dan perbaikan tepian objek (edge enhancement).

2. Restorasi citra (image restoration), contohnya penghilangan kesamaran (deblurring).

3. Pemampatan citra (image compression).

4. Segmentasi citra (image segmentation).

5. Pengorakan citra (image analysis), contohnya pendeteksian tepi objek (edge enhancement) dan ekstraksi batas (boundary).

6. Rekonstruksi citra (image recronstruction).

BAB III

PERANCANGAN SISTEM UNTUK COLOUR CODE PADA TEKNOLOGI RUMAH PINTAR

Aplikasi ini diberi nama Shroom yang merupakan singkatan dari *Smart Home and Room*.

Pada bab ini dilakukan tahap merancang yaitu proses dimana dibuat sebuah spesifikasi software yang dilakukan untuk mencapai tujuan, menggunakan sebuah set komponen-komponen yang mudah dimengerti dan memiliki suatu batasan. Pada penelitian ini digunakan Diagram UML, yang merupakan elemen sangat penting pada tahap perencanaan. Digunakan 5 jenis (*Unified Modelling Language*) UML.

## **3.1 *System Requirements***

### 3.1.1 *Use Case Diagram*

Diagram ini menjelaskan kepada pengguna fungsi-fungsi apa saja yang diberikan oleh sistem kepada mereka. Pada kasus ini, pengguna dapat memonitor kondisi ruangan, mengontrol perangkat electronik secara manual, dan menset perangkat elektronik untuk berjalan secara otomatis, seperti pada Gambar 3.1.

Pengguna

Mengontrol elektronik dengan *colour code*

Mengontrol perangkat elektronik manual

Set perangkat elektronik bekerja secara otomatis

Gambar 3.1 Use Case Diagram

## **3.2 *System and Software Design***

### 3.2.1 *Algoritma Sistem*

Flowchart pada Gambar 3.2 menjelaskan langkah-langkah aktifitas yang disediakan oleh sistem Shroom. Pertama-tama, pengguna mengaktifkan aplikasi, dan sensor memberikan informasi kepada sistem. Setelah sistem aktif, pengguna dapat memilih kipas atau charger, juga bisa lampu atau terminal. Jika pengguna telah memilih suatu sistem, maka perangkat elektronik yang dipilih akan menyala atau mati tergantung dari kondisi sebelumnya. Tidak hanya itu, pengguna dapat memilih mode otomatis sehingga sistem berjalan sesuai dengan aturan-aturan yang telah ditentukan sebelumnya.

Sementara pseudocode dari aplikasi/*software* Shroom dijelaskan pada Gambar 3.3. Diawali dengan menginisiasi pembacaan sensor dari sistem berupa sensor suhu dan sensor temperatur. Selanjutnya dilakukan pemindaian *(scan)* terhadap warna *base* atau warna *background*, beserta dengan 4 opsi warna lainnya yang disebut sebagai warna *first, second, third,* dan *fourth.* Lalu user dapat memilih menggunakan mode manual atau otomatis. Jika menggunakan mode otomatis, batas yang diset adalah bernilai 9000 lux. Apabila nilai intensitas di bawah itu, maka lampu akan menyala, dan apabila nilai intensitas cahaya di atas itu, maka lampu akan mati. Batas lainnya adalah nilai suhu yaitu 30 derajat celcius. Apabila nilai suhu di atas itu, maka kipas akan menyala, dan apabila nilai suhu di bawah itu, maka kipas akan mati. Jika menggunakan mode manual, maka program mengakses kamera ponsel. Apabila terdeteksi warna *base first second*, maka lampu akan berganti posisi. Apabila terdeteksi warna *base second third*, kipas akan berganti posisi. Apabila terdeteksi warna *base third fourth*, *charger* akan berganti posisi. Dan apabila terdeteksi warna *base first third*, maka terminal akan berganti posisi. Yang dimaksud berganti posisi di sini adalah jika *state* sebelumnya mati, maka akan menyala, dan apabila *state* sebelumnya menyala, maka akan mati.

Y

N

N

N

Y

N

Y

Y

Lampu Mati

Kipas mati

Kemba-li Ke awal?

Kipas menyala

Lampu Menyala

Temperatur di atas 30 derajat C?

Pengguna mematikan aplikasi

Aplikasi dimatikan

Aplikasi aktif

Sensor memberikan informasi ke sistem

Mode Otomatis?

Set Lux dan Temperatur ruangan

Lux 9000?

Memilih lampu, kipas, charger, atau terminal

Mematikan atau Menyalakan perangkat

Gambar 3.2 Flowchart Aplikasi Shroom.

|  |
| --- |
| float luxRead = lux\_.read();  luxRead = ((2500/luxRead) - 500);  float tempRead = temp\_.read();  tempRead = tempRead \* 350;  pickerLux\_.setValue(9000);  pickerTemp\_.setValue(30);  Scan ColorBase = (base)  Scan ColorFirst = (first)  Scan ColorSecond = (second)  Scan ColorThird = (third)  Scan ColorFourth = (fourth)  OnCameraViewStarted {  //automatic mode  If (base) (first) (fourth) {  //Automatic control for Lampu  if(intLuxRead < (userLux\*500)) {  lampu\_.write(true);  lampuStatus\_ = 1;  }  else {  lampu\_.write(false);  lampuStatus\_ = 0;  }    //Automatic control for kipas  if(intTempRead < userTemp) {  if(kipasStatus\_ == 1) {  kipas\_.write(true);  }  }  else {  if(kipasStatus\_ == 0) {  kipas\_.write(false);  }  }  Else {  class Looper extends BaseIOIOLooper {  @Override  {  if (base) {  (first) {  (second) {  lampu\_ = ioio\_.openDigitalOutput(9, change state); }}  (second){  (third) {  kipas\_ = ioio\_.openDigitalOutput(10, change state); }}  (third) {  (fourth) {  charger\_ = ioio\_.openDigitalOutput(11, change state);}}  (first) {  (third) {  terminal\_ = ioio\_.openDigitalOutput(12, change state);}}  lux\_ = ioio\_.openAnalogInput(40);  temp\_ = ioio\_.openAnalogInput(39);  }  } |

Gambar 3.3 Pseudocode Aktivitas Manual dan Otomatis

### 3.2.2 *Sequential Diagram*

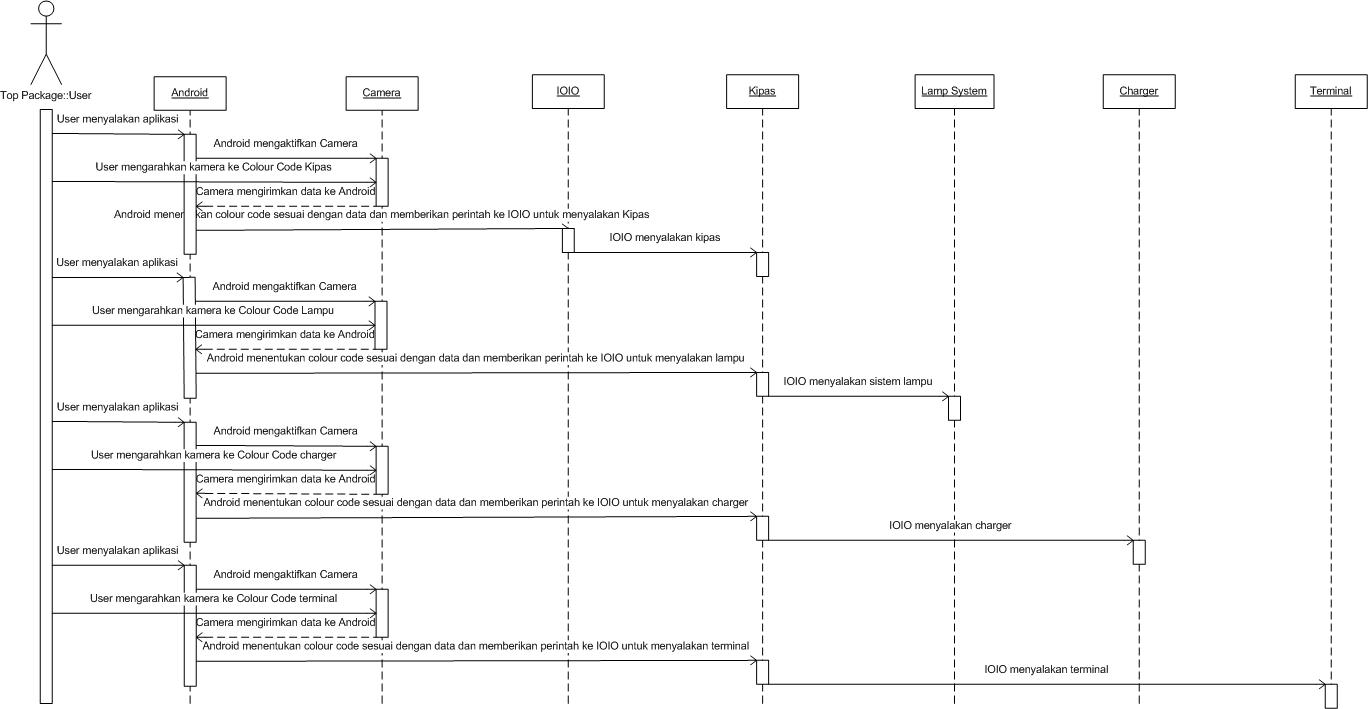
*Sequential Diagram* adalah diagram yang menjelaskan secara terperinci bagaimana sebuah sistem bekerja, khususnya *Embedded System*. Pada Gambar 3.4 dijelaskan proses yang terjadi ketika pengguna menyalakan kipas, lampu, charger, dan terminal.

Ketika pengguna ingin menyalakan kipas, maka pengguna menyalakan aplikasi dan membuka halaman utama aplikasi. Setelah itu, camera pada Android akan diaktifkan. Langkah berikutnya adalah user mengarahkan kamera ke Color Code Aki pas sehingga kamera akan mengirimkan data ke Android mengenai Color Code yang ditangkap kamera. Android, selanjutnya, akan mencocokkan data dengan batas-batas tertentu, dan batas-batas tersebut akan mengirimkan sinyal bahwa Color Code yang ditangkap kamera adalah Color Code kipas. Maka, Android akan mengirimkan perintah kepada IOIO untuk kemudian menyalakan sistem kipas. IOIO pun bekerja dan menyambungkan relay kipas sehingga kipas akan menyala.

Begitu pula ketika pengguna ingin menyalakan lampu, maka pengguna membuka halaman utama aplikasi. Setelah itu, camera pada Android akan diaktifkan. Langkah berikutnya adalah user mengarahkan kamera ke Color Code lampu, sehingga kamera akan mengirimkan data ke Android mengenai Color Code yang ditangkap kamera. Android, selanjutnya, akan mencocokkan data dengan batas-batas yang sudah ditentukan, dan batas-batas itu akan mengirimkan sinyal bahwa Color Code yang ditangkap kamera adalah Color Code lampu. Maka, Android akan mengirimkan perintah kepada IOIO untuk kemudian menyalakan sistem lampu. IOIO pun bekerja dan menyambungkan relay lampu sehingga lampu akan menyala.

Ketika pengguna ingin menyalakan charger, maka pengguna membuka halaman utama aplikasi. Setelah itu, camera pada Android akan diaktifkan. Langkah berikutnya adalah user mengarahkan kamera ke Color Code charger, sehingga kamera akan mengirimkan data ke Android mengenai Color Code yang ditangkap kamera. Android, selanjutnya, akan mencocokkan data dengan batas-batas yang sudah ditentukan, dan batas-batas itu akan mengirimkan sinyal bahwa Color Code yang ditangkap kamera adalah Color Code charger. Maka, Android akan mengirimkan perintah kepada IOIO untuk kemudian menyalakan charger. IOIO pun bekerja dan menyalakan charger sehingga charger akan menyala.

Begitu pula ketika pengguna ingin menyalakan terminal, maka pengguna membuka halaman utama aplikasi. Setelah itu, camera pada Android akan diaktifkan. Langkah berikutnya adalah user mengarahkan kamera ke Color Code terminal, sehingga kamera akan mengirimkan data ke Android mengenai Color Code yang ditangkap kamera. Android, selanjutnya, akan mencocokkan data dengan batas-batas tertentu, dan batas-batas tersebut akan mengirimkan sinyal bahwa Color Code yang ditangkap kamera adalah colour code terminal. Maka, Android akan mengirimkan perintah kepada IOIO untuk kemudian membuka terminal. IOIO pun bekerja dan menyambungkan relay terminal sehingga terminal akan terbuka.



Gambar 3.4 Sequential Diagram

## **3.3 Design Perangkat Keras *(Hardware)***

### 3.3.1 *Block Diagram*

*Block Diagram* mengilustrasikan arsitektur fisik dari sebuah sistem pada hardware. Komunikasi jaringan antara hardware dan node. Dengan menghubungkan Ponsel menggunakan Bluetooth Dongle dengan IOIO, IOIO mengintegrasikan ponsel pintar kepada sistem yang terdiri dari: Sensor suhu, sensor cahaya, relay 1 (lampu), relay 2 (kipas), relay 3 (charger), dan relay 4 (terminal). *Block diagram* dapat dilihat pada Gambar 3.5.



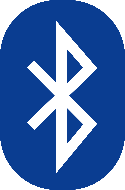
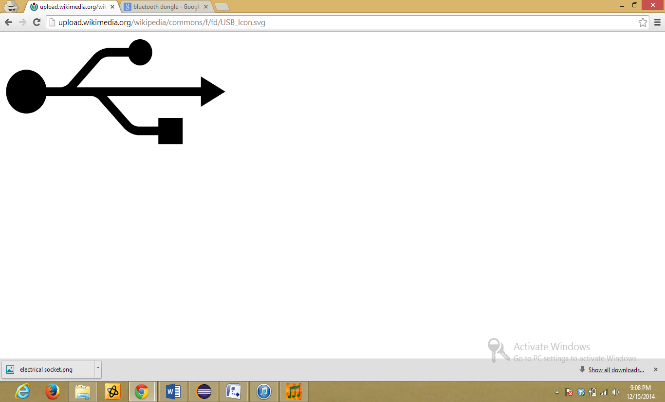
**IOIO**

**RELAY 1**

**RELAY 2**

**RELAY 3**

**RELAY 4**



SENSOR SUHU

SENSOR TEMPERATUR

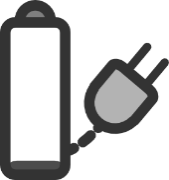
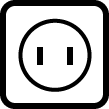
Gambar 3.5 Block Diagram

## 3.4 Skema Color Code

Dewasa ini, terdapat beberapa kode yang digunakan untuk mereferensikan sesuatu. Sebagai contoh ialah Barcode dan QR code. Barcode sering digunakan untuk keperluan mengenali suatu produk, sebagai contoh untuk mengidentifikasikan buku dan produk. Sementara itu, QR code biasanya digunakan untuk mengkodekan URL atau data-data terkait dengan internet. Penulisan ini mengajukan sebuah cara baru yang inovatif untuk membedakan benda-benda dengan menggunakan kode tertentu yang dinamakan Color Code. Color Code secara visual jauh lebih menarik dan mudah untuk dimengerti oleh mata manusia, disamping memiliki kapabilitas yang baik juga untuk dimengerti oleh mesin secara kode. Inilah yang menyebabkan Color Code menang dibandingkan dengan QR code dan Barcode.

Walaupun Color Code memiliki kemampuan untuk mudah diaplikasikan pada banyak tujuan, namun Color Code memiliki tujuan khusus untuk diimplementasikan pada bidang hiburan (entertainment) dan gaya hidup (lifestyle), karena warna itu sendiri memiliki karakteristik lebih mudah dimengerti oleh mata manusia, namun cenderung memiliki kemampuan yang standar terhadap pembacaan dari mesin.

Color Codeadalah inovasi baru/fitur tambahan yang dimasukkan ke dalam Sistem rumah pintar. Ini merupakan cara lain untuk mengontrol (mematikan dan menyalakan) perangkat elektronik secara manual. Hanya dengan melakukan *scanning* ke *color code* tertentu, maka perangkat yang dituju dengan mudah dapat dikontrol. Setelah mengaktifkan aplikasi *Shroom*, pengguna mengaktifkan mode Color Code, setelah itu, pengguna melakukan *scanning* dengan menggunakan kamera yang ada di telepon genggam pintar pengguna. Color Codebekerja dengan menggunakan 1 warna dasar, dalam hal ini, digunakan warna dasar berwarna merah, dan menggunakan 4 warna variasi, yaitu warna kuning, hijau, biru, dan hitam. Aplikasi dengan mode Color Codeakan *scanning* warna merah sebagai background, dan 2 warna lain sesuai dengan kode perangkat elektroniknya. Jika ditemukan suatu Color Codetertentu, sebagai contoh, warna hitam dan hijau, maka Color Codeini akan merujuk kepada pengontrolan terminal. Lalu, dengan telepon genggam yang terhubung ke sistem dengan Bluetooth, data akan dikirimkan kepada sistem untuk membuka atau menyalakan atau mematikan terminal. Scanning yang dilakukan oleh pengguna dapat dilakukan dengan jarak jauh, asalkan kode tersebut dapat ditangkap oleh kamera. Sehingga didapatkan keunggulan lainnya dari Color Code adalah dengan kemampuannya untuk melakukan Scanning kode dengan jarak jauh, sehingga menghemat beberapa detik bagi pengguna untuk mengontrol perangkat elektronik dibandingkan dengan harus berjalan menuju ke saklar perangkat elektronik tersebut.. Skema Color Code dapat dilihat pada Gambar 3.6.



Bluetooth

System

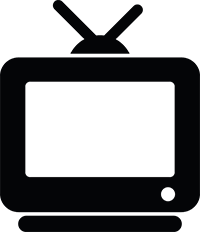


SCAN

(Camera)

Gambar 3.6 Skema Colour Code

Karena Color Code memiliki keunggulan khusus yaitu sifatnya yang menarik pada mata manusia, maka Color Code cenderung digunakan untuk Lifestyle dan Entertainment. Sehingga nilai tambah yang ditawarkan oleh Color Code adalah untuk menambah nilai dari suatu barang. Oleh karena itu, rancangan dari Color Code yang diajukan untuk menambah nilai dari kode yang ada, adalah dengan merancang kode sedemikian rupa sehingga jauh lebih mudah dimengerti oleh mata manusia dibandingkan dengan menggunakan kode lainnya. Pada gambar 3.7 berikut adalah rancangan Color Code sehingga mudah dimengerti oleh mata manusia. Secara mudah terlihat bahwa ini merupakan Color Code untuk mengontrol televisi.



Gambar 3.7 Rancangan Color Code

Metodologi dalam menggunakan color code adalah sebagai berikut :

1. Colors : adalah sebuah kombinasi dari warna yang menarik yang digunakan sebagai kode seperti yang ada pada Gambar 3.8.

Gambar 3.8 Warna Pada Colour Code

2. Melakukan *scanning* pada kombinasi warna yang digunakan adalah tahap kedua untuk mendapatkan data. Langkah ini dilakukan dengan menggunakan kamera, khususnya kamera dari ponsel pintar.

3. Dilakukan proses *install* pada aplikasi *Shroom*, pada ponsel pintar yang terintegrasi dengan kamera. Jika kamera mendapatkan indikasi adanya kesamaan antara warna yang terlihat dan warna yang ada pada batas yang telah diberikan, maka akan diberikan simbol atau yang disebut sebagai *flag*, dengan ketentuan *flag* 1 untuk adanya warna tertentu, dan *flag* 0 untuk ketiadaan warna tertentu.

4. Aksi: merupakan tahap dimana kombinasi data-data berupa *flag* tersebut dibandingkan dengan batas-batas yang telah ditentukan sebelumnya sehingga terjadi beberapa macam kasus yang berbeda sesuai dengan kebutuhan. Setelah didapatkan kasus tertentu, maka setiap kasus tersebut dilakukan pemrosesan untuk *output* tertentu sehingga dalam program *Shroom* ini memberikan arahan kepada mikroprosesor untuk mematikan ataupun menyalakan perangkat elektronik tertentu.

Pada penelitian ini, digunakan pembatasan yaitu 1 warna untuk latar belakang dan 4 warna untuk warna opsi. Tiap 4 warna opsi tersebut dipakai 2 warna sebagai warna kode yang nantinya akan mereferensikan kepada suatu perangkat elektronik tertentu. Dengan pembatasan tersebut, maka didapatkan dengan rumus kombinasi seperti pada rumus 3.1, yaitu:

nCr (3.1)

Atau khususnya, jika kita mengambil kasus untuk warna opsi yang digunakan pada penelitian in, didapatkan perhitungan seperti pada perhitungan 3.2, yaitu:

4C2 (3.2)

Dengan rumus tersebut maka dapat dihitung jumlah kasus yang dapat digunakan adalah seperti pada perhitungan 3.3, yaitu:

(3.3)

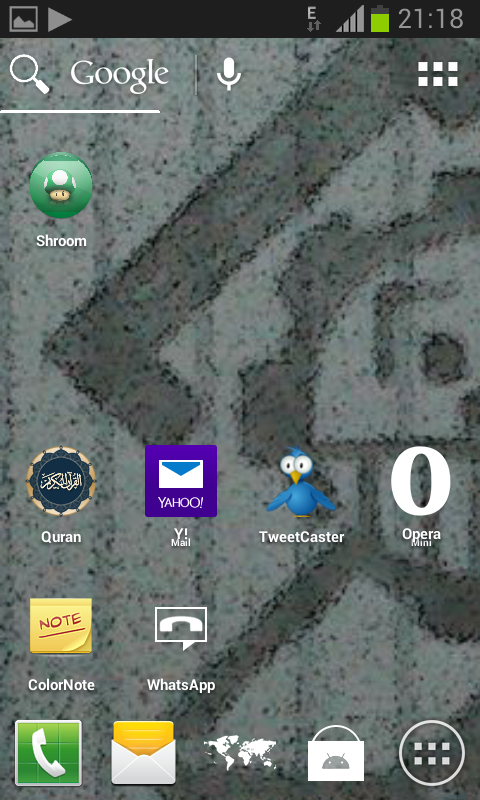
Maka didapatkan 6 kasus atau batasan perangkat elektronik adalah berjumlah 6.

|  |
| --- |
| Inisiasi: spectrum size, contour color : {base, first, second, third, fourth}  Require: average seluruh value {base, first, second, third, fourth}  for (int i = 0; i < mPixelCounter.val.length; i++) {  if(mPixelCounter.val[i] > 500000) mPixelBase += 1; //maximum 4.895.935  }  if(Base == 1) base = true;  else base = false;  }  for (int i = 0; i < mPixelCounter.val.length; i++) {  if(mPixelCounter.val[i] > 100000) //maximum 4.895.935  }  if(First == 1) first = true;  else first = false;  }  for (int i = 0; i < mPixelCounter.val.length; i++) {  if(mPixelCounter.val[i] > 100000) mPixelBase += 1; //maximum 4.895.935  }  if(Second == 1) second = true;  else second = false;  }  for (int i = 0; i < mPixelCounter.val.length; i++) {  if(mPixelCounter.val[i] > 100000) mPixelThird += 1; //maximum 4.895.935  }  if(Third == 1) third = true;  else base = false;  }  for (int i = 0; i < mPixelCounter.val.length; i++) {  if(mPixelCounter.val[i] > 100000) mPixelFourth +=1;//maximum 4.895.935  }  if(Fourth == 1) fourth = true;  else fourth = false;  } |

Gambar 3.9 Pseudocode Color Blob Setup dan Pemberi *Flag*

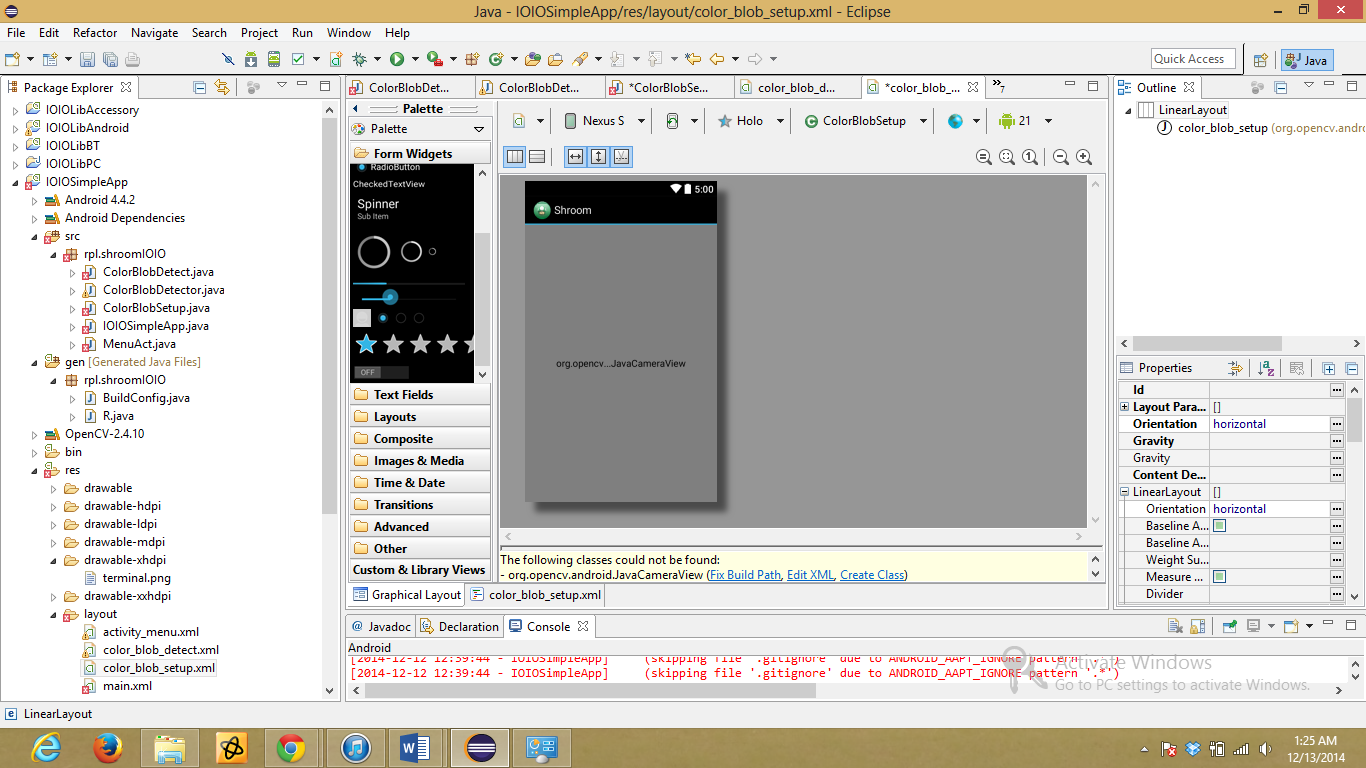
Pseudocode pada Gambar 3.9 menjelaskan bagaimana alur dari setup color blob dan pemberian *flag* pada tiap eksistensinya sebuah warna. Diawali dengan menginisiasi ukuran spektrum dan kontur warna dari warna *base, first, second, third,* dan *fourth.*

Selain itu dibutuhkan juga seluruh value yang terdiri dari *upper bound* dan *lower bound* yang masing-masing memiliki 4 nilai, sehingga total memiliki 8 nilai. 8 nilai tersebut dibutuhkan pada masing-masing warna *base, first, second, third,* dan *fourth.* Ke delapan nilai tersebut dibutuhkan untuk nantinya di ukur rata-rata dari nilai tersebutuntuk dipindai dan dilanjutkan dengan proses berikutnya. Jika pada nilai dari warna *base* lebih besar dari 500.000, maka *flag base* akan bernilai true atau 1, jika warna *base* kurang dari 500.000, maka *flag base* akan bernilai false. Selain itu, jika pada nilai dari warna *first, second, third,* dan *fourth* lebih besar dari 100.000, maka *flag base* akan bernilai true atau 1, jika warna *first, second, third,* dan *fourth* kurang dari 100.000, maka *flag base* akan bernilai false.Seperti yang terlihat di Gambar 3.10, cara memakai aplikasi Shroom adalah melakukan instalasi Shroom.apk terlebih dahulu, lalu klik logo Shroom yang ada di ponsel genggam.



Gambar 3.10 Logo Shroom

Gambar 3.11 menunjukkan aplikasi Shroom yang mengakses kamera ponsel pintarnya, untuk dilakukan *scanning* kombinasi Color Code yang tersedia.



Gambar 3.11 Aplikasi Mengakses Kamera

BAB IV

ANALISIS IMPLEMENTASI INTEGRASI COLOR CODE DENGAN SISTEM RUMAH PINTAR

Pada bab ini akan dijelaskan mengenai analisis implementasi integrasi color code dengan sistem rumah pintar yang telah dibuat. Analisis tersebut berupa pengujian terhadap sistem yang telah dibuat dan juga penilaian terhadap keseluruhan fungsi.

## **4.1 Tahap Implementasi Sistem**

Sistem Color Code menggunakan teknik pengolahan citra. Perangkat keras yang digunakan adalah-IOIO On The Go, 12VDC to 220VAC relay. Perangkat lunak yang digunakan adalah Eclipse untuk Bahasa pemrograman Java dengan *library* Open CV.

Sedangkan mesin pengolah yang digunakan adalah sebuah komputer laptop dengan spesifikasi seperti pada Gambar 4.1:

* Processor Intel® Core™ i5-2410M CPU @ 2.30GHz
* Memori 8.00 GB
* Sistem type: 64-bit Operating System, x64-based processor
* Eclipse Version: Luna Service Release 1 (4.4.1) [8].



Gambar 4.1 Mesin Pengolah

Untuk rangkaian sistem yang dibuat, dapat dilihat seperti pada Gambar 4.2 dan pada Gambar 4.3 dapat dilihat fungsi penuh dari penelitian ini. Hardware sistem terdiri dari:

1. Adaptor

2. IOIO-On The Go

3. USB Cable

4. Bluetooth Module

5. Color Code

6. Relay

7. Rangkaian Sensor Cahaya

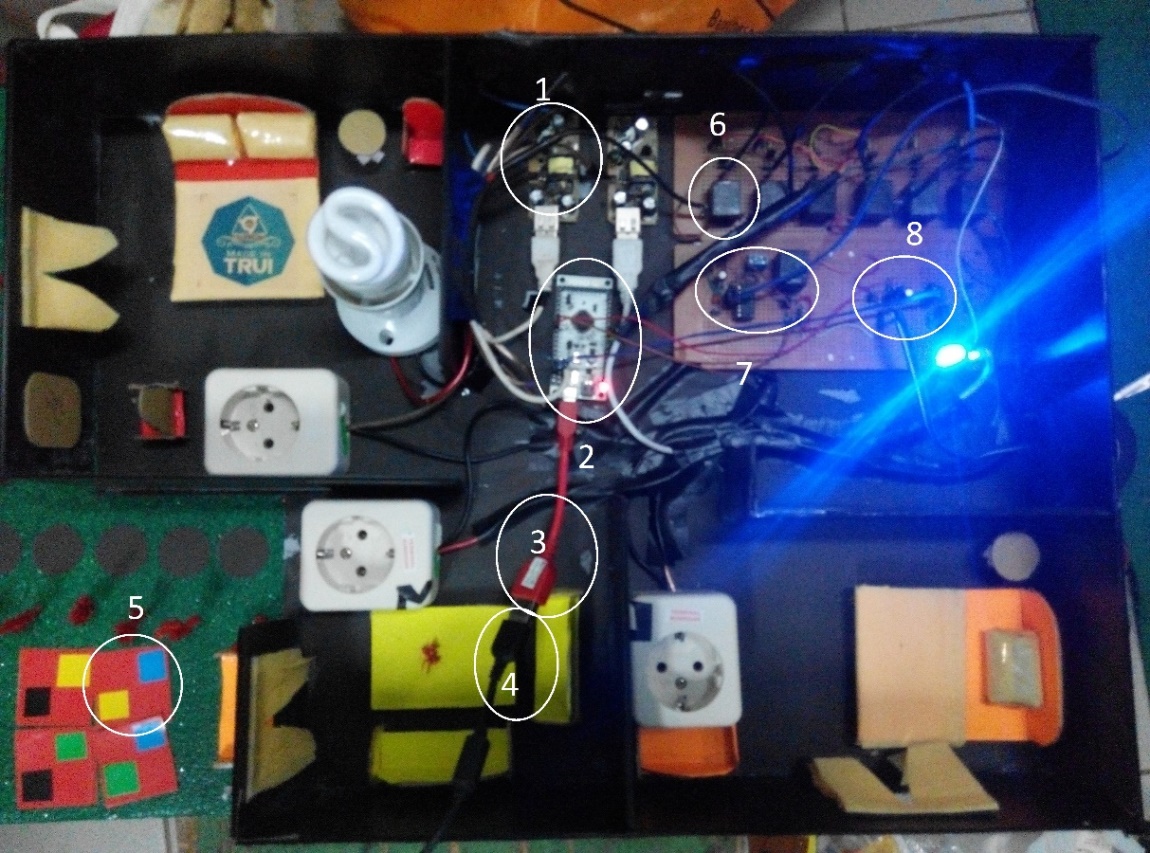
8. Rangkaian Sensor Suhu

9. Kipas

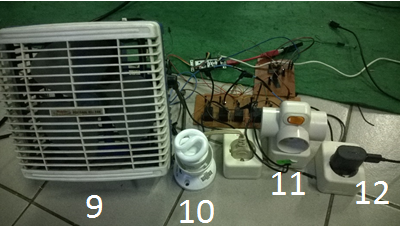
10. Lampu

11. Terminal

12. Charger



Gambar 4.2 Rangkaian Sistem (1)



Gambar 4.3 Rangkaian Sistem (2)

Selain itu untuk menjalankan aplikasi digunakan pula sebuah ponsel pintar seperti pada Gambar 4.4 dengan spesifikasi sebagai berikut:

* Memori 2.00 GB
* Prosesor Octa-core 1.7GHz
* Android version 4.2.2
* Kamera belakang 13 Mega Pixel
* Kamera depan 5 Mega Pixel
* Layar 5.5 inchi
* SD Card Support



Gambar 4.4 Ponsel Pintar untuk Menjalankan Aplikasi

## **4.2 Pengujian Interval Waktu Antara Input Gambar dari Kamera dan Output Pemberian Perintah Kepada Mikroprosesor untuk Mengontrol Perangkat Elektronik**

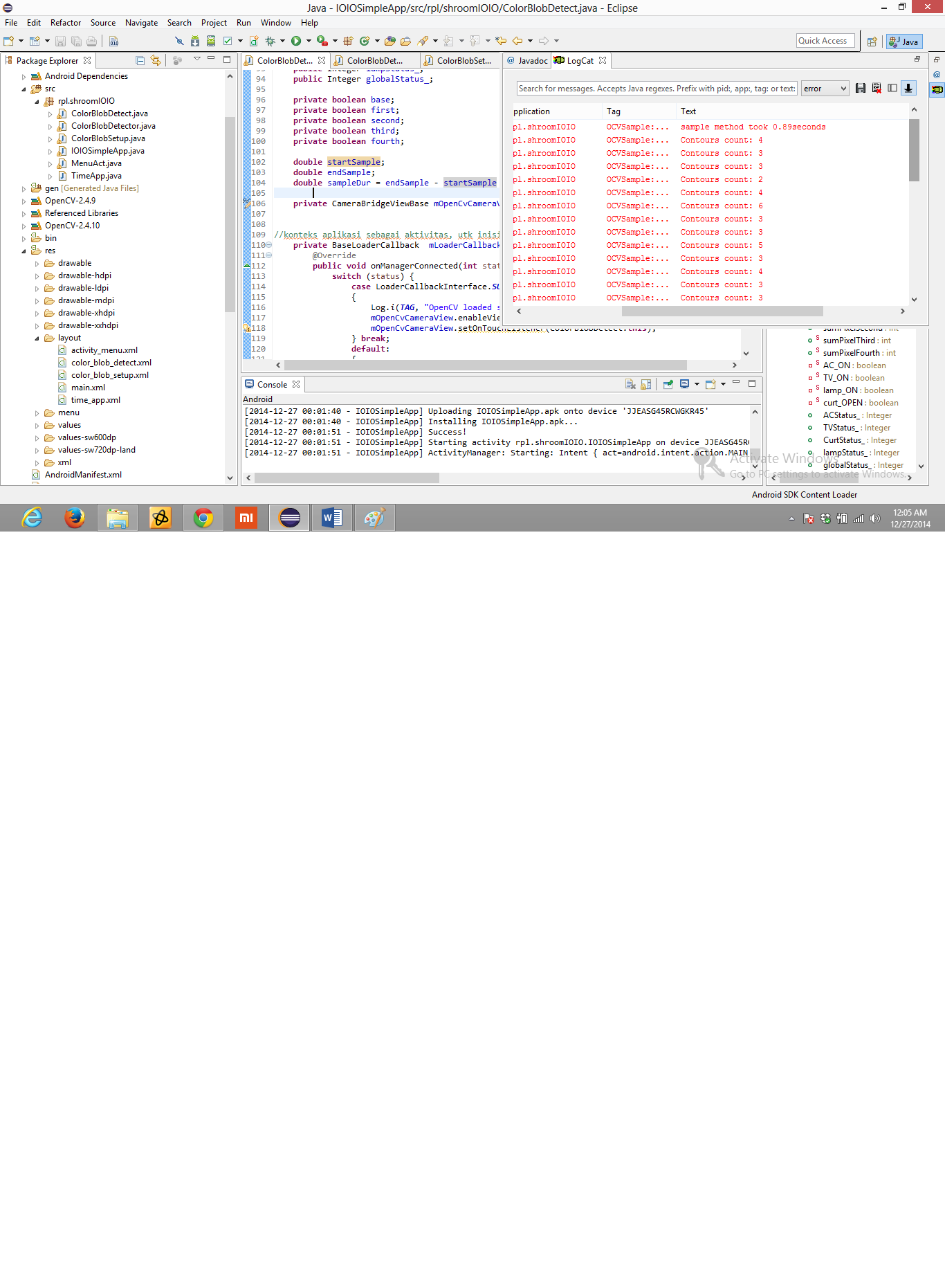
Untuk pengujian interval waktu antara proses melakukan scan hingga pemberian perintah kepada perangkat elektronik dilakukan dengan menggunakan algoritma pada program Java sendiri. Hasil dari pengujian tersebut dapat dilihat pada Tabel 4.1.

Setelah dilakukan percobaan tersebut, didapatkan bahwa interval waktu antara input gambar dari kamera ponsel dan output berupa pemberian perintah kepada mikroprosesor untuk mengontrol perangkat elektronik yang diimplementasikan pada program ini adalah sebesar 0.892 detik.

Tabel 4.1 Interval Waktu Antara Input dan Output

|  |  |
| --- | --- |
| Nomor Percobaan | Hasil (waktu dalam detik) |
| 1 | 0.89 |
| 2 | 0.91 |
| 3 | 0.88 |
| 4 | 0.88 |
| 5 | 0.87 |
| 6 | 0.94 |
| 7 | 0.89 |
| 8 | 0.91 |
| 9 | 0.88 |
| 10 | 0.87 |
| Rata-rata | **0.892** |

Percobaan ini dilakukan dengan mendeteksinya dengan LogCat yang terdapat pada Java Eclipse. Pendeteksian ini dilakukan dengan cara memulai menghitung pada saat program dijalankan, khususnya ketika gambar dari kamera mulai ditangkap, dan mengakhiri menghitung ketika instruksi selesai diberikan kepada mikroprosesor, khususnya IOIO OTG. Keduanya dilakukan dengan metode perhitungan sederhana dengan fungsi ‘StartSample’ dan ‘End Sample’. Setelah itu, dilakukan logging untuk mencetak rentang waktu yang diperlukan untuk melakukan percobaan ini. Rentang waktu tersebut dapat dilihat dengan menggunakan LogCat pada inisial ‘e’, sehingga didapatkan tulisan yaitu ‘sample method took …seconds’ seperti pada gambar 4.5. Percobaan ini dilakukan sebanyak 10 kali sehingga dapat didapatkan rata-rata dari pengujian interval waktu ini adalah sebesar 0.892 detik. Terdapat perbedaan interval waktu dari pengambilan gambar hingga pemberian perintah karena prosesor pada ponsel yang memiliki kondisi yang berbeda-beda, tergantung aplikasi yang sedang berjalan dan pengaruhnya pada prosesor ponsel yang digunakan.



Gambar 4.5 Pengujian Interval Waktu

Dari data tersebut, maka dapat disimpulkan bahwa Shroom merupakan salah satu program atau *software* yang memiliki tingkat efisiensi yang cukup tinggi karena pengguna dapat mengontrol perangkat elektronik dalam waktu hanya kurang dari 1 detik. Bahkan hal ini menjadi suatu keunggulan tersendiri karena pengguna tidak perlu menyalakan ataupun mematikan perangkat elektronik secara manual, sehingga menghemat tenaga dan waktu beberapa detik untuk berjalan mematikan ataupun menyalakan saklar perangkat elektronik dengan tangan pengguna. Kelebihan Color Code yang lain adalah visualnya yang sangat menarik sehingga dapat dibuat kode-kode visual yang mudah dimengerti oleh mata untuk mengontrol perangkat elektronik. Kemudahan Color Code juga tercermin dari pengontrol pusat yang selalu dibawa pengguna selama kesehariannya, yaitu ponsel pintar mereka sendiri. Pengguna tidak perlu menggunakan remote khusus dalam mengontrol aplikasi ataupun sistem ini. Dengan mudah pengguna dapat mengontrol alat elektronik mereka hanya dengan satu sentuhan pada ponsel pintar pengguna. Color Code juga memiliki sifat yang sederhana dan memiliki tingkat keandalan yang tinggi, karena teknologi ini menggunakan Bluetooth untuk koneksi dari ponsel pintar kepada sistem melalui IOIO-OTG. Dengan modul Bluetooth, untuk mengontrol perangkat elektronik tersebut pengguna hanya tinggal menyalakan Bluetooth ponselnya dan menyambungkannya ke Bluetooth sistem. Sehingga dalam jarak kurang lebih 10 meter dengan sistem, pengguna masih tetap dapat melakukan pengontrolan terhadap perangkat elektronik. Sistem Hardware Color Code ini pun tidak memerlukan tempat yang banyak, karena rangkaian-rangkaian elektronik yang digunakan memiliki ukuran yang cenderung kecil.

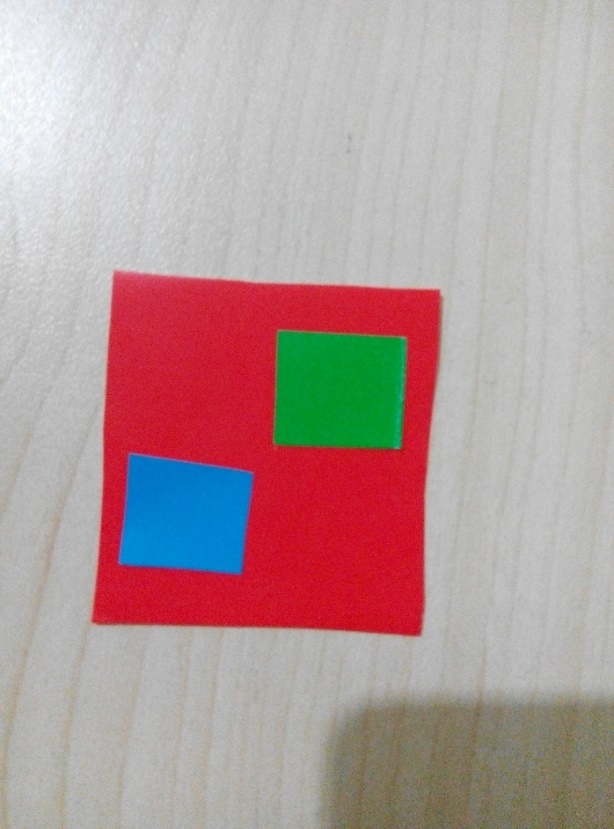
Selain itu, untuk perkembanganya apabila teknologi Color Code ini digabungkan dengan sistem monitor atau pemantauan perangkat elektronik, pengguna dapat mengakses dan memonitor kondisi sensor-sensor pada alat elektronik tertentu sesuai yang diinginkan hanya dengan melakukan *scan* pada Color Code. Akhirnya, Color Code bertujuan untuk memudahkan hidup manusia untuk mengontrol perangkat elektroniknya dengan ponsel pintar mereka sendiri, yang memiliki keunggulan tampilan Color Code-nya yang *user-friendly*.

## **4.3 Pengujian Error Rate dalam Software**

Pengujian Error Rate dilakukan untuk menghitung performa dan tingkat keberhasilan dari masing-masing kombinasi warna yang ada. Dengan 4 opsi warna didapatkan 6 buah kasus, dalam hal ini diambil 4 kasus yang merujuk kepada 4 perangkat elektronik tertentu. 4 kombinasi warna tersebut adalah warna hijau dan biru, warna biru dan kuning, warna hitam dan hijau, dan warna kuning dan hitam. Hasil pengujian tersebut dapat dilihat pada Tabel 4.2

Tabel 4.2 Pengujian Error Rate dalam Sistem

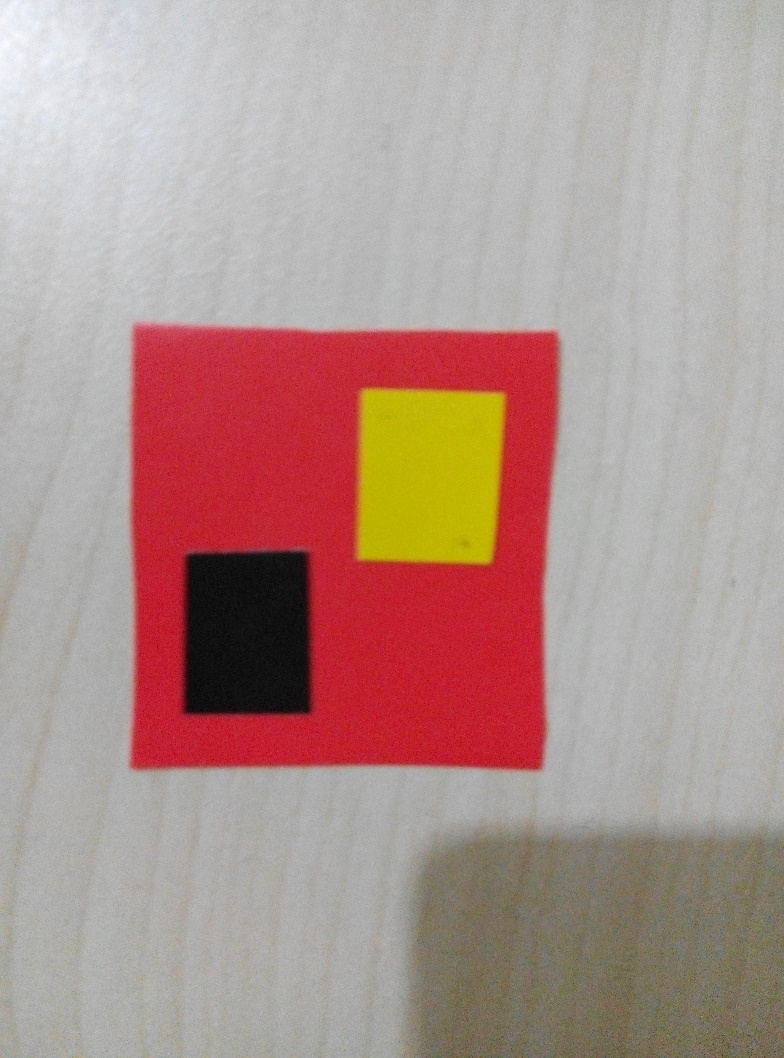
|  |  |  |  |
| --- | --- | --- | --- |
| Nomor Pengujian | Kombinasi Warna | Jumlah Percobaan | Error Rate |
| 1 | Hijau dan Biru | 10 | 0% |
| 2 | Biru Kuning | 10 | 0% |
| 3 | Kuning Hitam | 10 | 0% |
| 4 | Hitam Hijau | 10 | 0% |



Gambar 4.6 Nomor pengujian 1 Kombinasi Warna Hijau dan Biru



Gambar 4.7 Nomor pengujian 2 Kombinasi Warna Biru dan Kuning



Gambar 4.8 Nomor pengujian 3 Kombinasi Warna Kuning dan Hitam



Gambar 4.9 Nomor pengujian 4 Kombinasi Warna Hitam dan Hijau

Dari keempat pengujian kombinasi warna yang dilakukan pada penelitian ini, dilakukan pada saat ruangan dalam keadaan terang, yaitu bernilai sekitar 4000 lux. Pada hasil pengujian ini, didapatkan bahwa seluruh kombinasi Color Code berhasil di-*scan* dan melakukan tugasnya untuk mematikan ataupun menyalakan perangkat elektronik sesuai dengan fungsinya. Berdasarkan analisis, hal ini dapat terjadi karena warna-warna yang digunakan memiliki kontras yang cukup tinggi sehingga kamera pada ponsel pintar dan aplikasi Shroom dapat membedakan dan menerjemahkan warna-warna tersebut menjadi kode dengan lancar.

Pada pengujian ini, dapat dianalisis bahwa warna-warna yang digunakan pada penelitian ini memiliki nilai warna yang cukup jauh secara HSV, sehingga warna pada Color Code yang ditangkap oleh kamera ponsel tidak tertukar. Dengan kata lain, kamera ponsel menangkap seluruh warna dengan tepat, walaupun terdapat toleransi warna secara HSV pada kodingan program. Pada penelitian ini digunakan 1 warna dasar merah dan 4 warna opsional yaitu warna biru, hitam, hijau, dan kuning. Seluruh warna tersebut pada kondisi ruangan dan pencahayaan yang sama memiliki nilai yang cukup stabil sehingga dapat dilakukan pendeteksian warna terhadap seluruh warna tersebut tanpa terjadi kesalahan.

Pengguna dapat melakukan setup Color Code pada menu ColorBlobSetup pada aplikasi Shroom. Setup yang dilakukan adalah dengan melakukan *scanning* kepada lima warna yang menjadi Color Code, yaitu 1 warna *base* dan empat warna opsional. Pada pengujian 4.3 yang dilakukan ini, setup dan deteksi warna untuk pengontrolan perangkat elektronik ini dilakukan pada kondisi ruangan dan pencahayaan yang sama, serta pada waktu yang sama pula (sore hari). Karena kondisi pencahayaan yang mirip itulah dapat ditarik analisisnya, bahwa pengujian dengan tingkat kesuksesan 100% atau error 0% terjadi.

Namun, faktanya, pencahayaan pada ruangan, terlebih lagi pada pencahayaan luar ruangan, memiliki nilai yang sangat volatil atau mudah berubah-ubah. Hal tersebut dapat mengakibatkan kamera untuk mendeteksi nilai warna yang berbeda, walaupun Color Code yang digunakan adalah sama. Karena warna memiliki ketergantungan yang tinggi terhadap pencahayaan. [10] [11].

Itulah sebab besar teknologi pengolahan citra Kartrek, yang digunakan untuk kode kereta pada tahun 1970an, menemui batasnya dimana hingga sekarang teknologi tersebut tidak lagi digunakan. Teknologi tersebut pada kondisi pencahayaan tertentu memang memiliki reliabilitas yang cukup tinggi, serta dapat memudahkan mata manusia untuk membedakan kode tersebut. Namun, apabila pencahayaan yang ada berubah, maka warna yang ditangkap oleh kamera pun berbeda, yang menyebabkan mesin untuk menerjemahkan kode menangkap nilai warna yang lain. Nilai warna yang dimaksud adalah nilai warna HSV, karena pemrosesan gambar tidak bisa diproses secara RGB, melainkan harus dengan nilai warna HSV. Sementara, warna yang ditangkap oleh kamera secara umum adalah warna dengan nilai RGB. Pada benda dengan warna yang sama, dapat memiliki nilai HSV yang berbeda pada pagi hari dan pada sore hari, apalagi ketika malam hari, yang disebabkan karena perbedaan pencahayaan yang diberikan oleh matahari.

Untuk mengatasi hal tersebut, aplikasi Shroom memiliki mode untuk mengatur atau melakukan setup pada warna sehingga deteksi warna bisa dilakukan persis setelah pengguna melakukan setup warna, tidak dengan membatasi warna pada kodingan. Sehingga perbedaan pencahayaan ataupun kesalahan pendeteksian kamera dapat diminimalisir. Secara ideal, jika pengguna ingin menggunakan aplikasi ini dengan optimal, maka setiap waktu ketika terdapat perbedaan pencahayaan, pengguna dapat melakukan setup warna ulang, sehingga mesin dapat mendeteksi warna secara tepat.

## **4.4 Hasil Penilaian Sistem Oleh Pengguna**

Pengujian ini adalah berbentuk penilaian terhadap sistem dalam bentuk kuisioner berupa pertanyaan-pertanyaan yang terkait dengan efisiensi dan efektivitas dari program Color Code untuk integrasi sistem rumah pintar. Responden yang berkontribusi dalam menjawab pertanyaan-pertanyaaan ini terdiri dari 10 orang mahasiswa dan 5 orang yang professional pada bidang teknik komputer. Pertanyaan-pertanyaan yang diberikan kepada pengguna dapat dilihat pada Lampiran 1.

Secara umum, berdasarkan hasil pengujian, yang terdapat pada Lampiran 2, pengujian dengan menggunakan kuisioner mengindikasikan bahwa performa aplikasi Shroom berjalan dengan baik berdasarkan dari nilai rata-rata pada setiap jenis pengujian, yaitu pada *system* *testing*, *consistency* *testing*, serta *user interface testing*. Namun terdapat sedikit kelemahan yaitu pada *consistency testing*, aplikasi Shroom masih memerlukan perkembangan karena masih terdapat beberapa pengguna yang belum dapat dengan mudah mengerti setiap *activity* pada program Shroom.



### 4.4.1 *System Testing*

Sistem harus dapat memberikan informasi kepada user tentang apapun yang sedang terjadi pada sistem melalui respon dan waktu yang tepat. Tabel 4.3 menjelaskan hasil dari formulir kuisioner yang dibagikan kepada 10 pengguna yang merupakan mahasiswa dan 5 pengguna yang merupakan professional di bidang teknik computer, yang mengetes aplikasi Shroom.

Tabel 4.3 Hasil Penilaian System Testing

|  |  |  |
| --- | --- | --- |
| No | System Testing | C |
| 1 | Apakah aplikasi berhasil dijalankan? |  |
| 2 | Apakah mode manual berfungsi? |  |
| 3 | Apakah mode automatis berfungsi? |  |
| 4 | Apakah sensor suhu dapat berfungsi dengan baik? |  |
| 5 | Apakah sensor cahaya dapat berfungsi dengan baik? |  |
| 6 | Apakah Setup Color Code dapat dijalankan? |  |
| 7 | Apakah relay 1 dapat dikontrol secara manual dengan Color Code |  |
| 8 | Apakah relay 2 dapat dikontrol secara manual dengan Color Code |  |
| 9 | Apakah relay 3 dapat dikontrol secara manual dengan Color Code |  |
| 10 | Apakah relay 4 dapat dikontrol secara manual dengan Color Code |  |
| C: Check List (10) | |  |

### 4.4.2 Consistency Testing

Sistem seharusnya menggunakan penamaan yang standard dan konsisten pada keseluruhan sistem, agar tidak menimbulkan ambiguitas ketika digunakan. Hasil consistency testing dapat dilihat pada tabel 4.4 dan 4.5.

Tabel 4.4 Hasil Penilaian Consistency Testing (Profesional)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Consistency Testing | 1 | 2 | 3 | 4 | 5 | AVG |
| 1 | Apakah setiap activity mudah untuk dimengerti? | 3 | 3 | 2 | 4 | 4 | 3.2 |
| 2 | Apakah Anda mengerti dengan hanya membaca judul disetiap halamannya? | 2 | 3 | 3 | 3 | 2 | 2.6 |
| 3 | Apakah setiap tombol memiliki konsistensi dan tidak menimbulkan ambiguitas? | 4 | 3 | 2 | 3 | 3 | 3 |
| RATA-RATA | | | | | | | 2.933333 |

Tabel 4.5 Hasil Penilaian Consistency Testing (Mahasiswa)

|  |
| --- |
| 1, Tidak sesuai. 2, Sedikit tidak sesuai. 3, Sedikit sesuai. 4, Sesuai |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Consistency Testing | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | AVG |
| 1 | Apakah setiap activity mudah untuk dimengerti? | 3 | 4 | 3 | 4 | 4 | 3 | 4 | 4 | 3 | 4 | 3.6 |
| 2 | Apakah Anda mengerti dengan hanya membaca judul disetiap halamannya? | 4 | 2 | 4 | 3 | 3 | 4 | 2 | 3 | 4 | 4 | 3.3 |
| 3 | Apakah setiap tombol memiliki konsistensi dan tidak menimbulkan ambiguitas? | 4 | 3 | 3 | 3 | 4 | 3 | 4 | 3 | 3 | 3 | 3.3 |
| RATA-RATA | | | | | | | | | | | | 3.4 |
| RATA-RATA TOTAL | | | | | | | | | | 3.166666667 | | |

### 4.4.3 *User Interface Testing*

Desain yang dibuat harus mudah dimengerti dan menarik perhatian pengguna. Hasil user interface testing dapat dilihat melalui tabel 4.6 dan 4.7.

Tabel 4.6 Hasil Penilaian UI Testing (Profesional)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | UI Testing | 1 | 2 | 3 | 4 | 5 | AVG |
| 1 | Apakah tampilan aplikasi menarik? | 2 | 3 | 3 | 1 | 3 | 2.4 |
| 2 | Apakah tombol gambar pada menu utama yang digunakan cukup representatif? | 3 | 4 | 3 | 4 | 4 | 3.6 |
| 3 | Apakah warna Color Code pada fitur kontrol alat elektronik mudah dimengerti? | 2 | 3 | 2 | 1 | 1 | 1.8 |
| 4 | Apakah jenis huruf dan ukurannya nyaman untuk dibaca? | 3 | 2 | 2 | 3 | 2 | 2.4 |
| RATA-RATA | | | | | | | 2.55 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | UI Testing | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | AVG |
| 1 | Apakah tampilan aplikasi menarik? | 4 | 4 | 3 | 4 | 3 | 3 | 3 | 3 | 3 | 3 | 3.3 |
| 2 | Apakah tombol gambar pada menu utama yang digunakan cukup representatif? | 3 | 3 | 3 | 2 | 3 | 3 | 4 | 3 | 3 | 4 | 3.1 |
| 3 | Apakah warna Color Code pada fitur kontrol alat elektronik mudah dimengerti? | 2 | 4 | 2 | 3 | 4 | 3 | 3 | 3 | 4 | 4 | 3.2 |
| 4 | Apakah jenis huruf dan ukurannya nyaman untuk dibaca? | 3 | 4 | 3 | 3 | 4 | 4 | 4 | 3 | 4 | 4 | 3.6 |
| RATA-RATA | | | | | | | | | | | | 3.3 |
| RATA-RATA TOTAL | | | | | | | | | | 2.925 | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | | | | | | | | | | | | |
| 1, Tidak sesuai. 2, Sedikit tidak sesuai. 3, Sedikit sesuai. 4, Sesuai | | | | | | | | | | | | |

Tabel 4.7 Hasil Penilaian UI Testing (Mahasiswa)

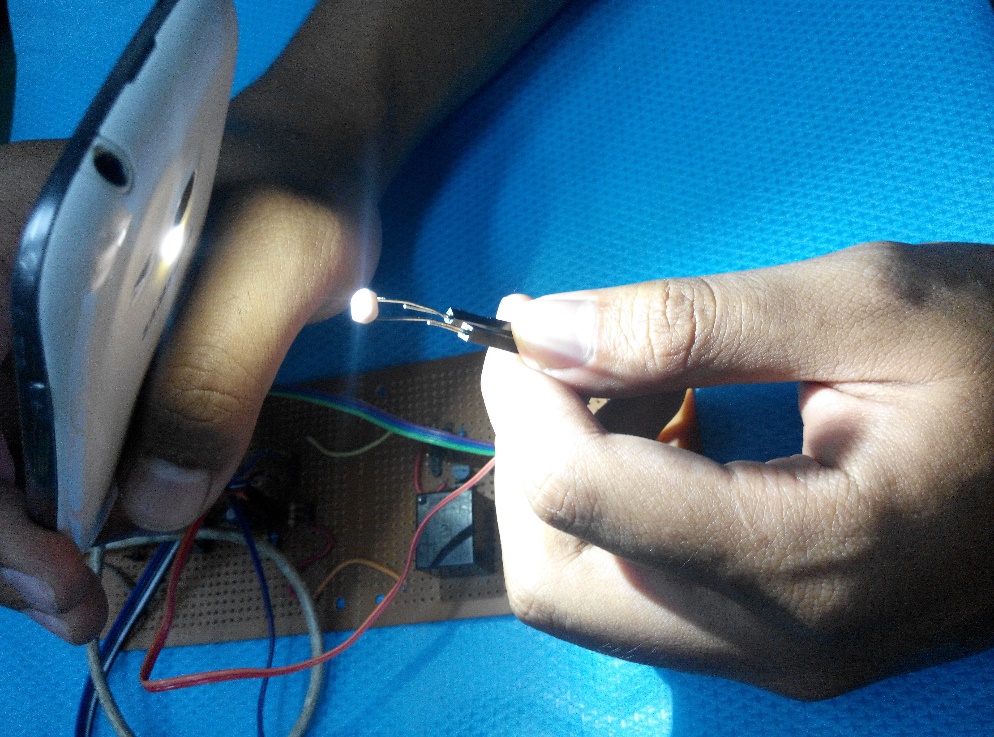
## **4.5 Pengujian Mode Otomatis pada Lampu**

Pada pengujian ini dilakukan pengujian mode otomatis pada lampu dengan memakai sensor cahaya (Light Dependent Resistor) sebagai penentu menyala atau matinya lampu pada sistem. Percobaan ini dilakukan pada ruangan yang memiliki intensitas cahaya sekitar 4000 lux, dengan batas yang ditentukan sebesar 9000 lux. Sistem ini bekerja dengan kondisi apabila intensitas cahaya yang datang dan ditangkap oleh LDR adalah sebesar 9000 lux atau lebih besar, maka lampu pada sistem akan mati, sebaliknya, jika intensitas cahaya yang datang dan ditangkap oleh sensor cahaya adalah kurang dari 9000 lux, maka lampu pada sistem akan menyala.

Dilakukan 10 kali percobaan pada pengujian ini, dimana setiap percobaan dilakukan 2 kali pengetesan terhadap *hardware* sistem. Percobaan tersebut berupa penerangan dengan menggunakan senter ke arah sensor cahaya untuk mendapatkan intensitas cahaya sama atau lebih besar dari 9000 lux, dan mematikan senter sehingga intensitas cahaya lebih kecil dari 9000 lux. Sehingga pengujian akan dikatakan berhasil apabila lampu pada sistem mati ketika senter dinyalakan, dan lampu pada sistem menyala ketika senter dimatikan.

Tabel 4.8 Hasil Pengujian Mode Otomatis pada Lampu

|  |  |  |
| --- | --- | --- |
| Nomor Percobaan | Ketika senter menyala | Ketika senter mati |
| 1 | Berhasil | Berhasil |
| 2 | Berhasil | Berhasil |
| 3 | Berhasil | Berhasil |
| 4 | Berhasil | Berhasil |
| 5 | Berhasil | Berhasil |
| 6 | Berhasil | Berhasil |
| 7 | Berhasil | Berhasil |
| 8 | Berhasil | Berhasil |
| 9 | Berhasil | Berhasil |
| 10 | Berhasil | Berhasil |
| Tingkat Keberhasilan | 100% Berhasil | 100% Berhasil |



Gambar 4.10 Pengujian Mode Otomatis pada Lampu

Analisis yang dapat diberikan terhadap hasil dari pengujian ini adalah LDR, atau sensor cahaya memiliki tingkat reliabilitas yang bagus. Dengan kondisi ruangan yang mendapatkan pencahayaan sore hari, dengan luas ruangan 3x3 meter, dengan kondisi lampu ruangan yang menyala, didapatkan bahwa intensitas cahaya pada ruangan memiliki nilai sekitar 4000 lux. Dengan dinyalakannya senter, yang merupakan flash dari ponsel, lalu didekatkan kepada sensor LDR tersebut, sudah jelas maka sensor akan membaca intensitas cahaya yang ditangkap akan semakin besar. Pada setiap percobaan didapatkan intensitas cahaya bernilai sekitar 10.000 hingga 12.000 lux, dengan kedekatan lampu flash dengan LDR sekitar 4 hingga 10 sentimeter. Hal tersebut yang menyebabkan lampu pada sistem akan mati secara otomatis karena membaca bahwa intensitas cahaya ruangan pada kondisi yang cukup terang (dengan batas intensitas cahaya sebesar 9000 lux). Sementara itu, jika *flash* dimatikan, maka sensor LDR pun akan menangkap intensitas cahaya ruangan pada kondisi awal, yaitu sekitar 4000 lux, yang menyebabkan lampu pada sistem akan menyala secara otomatis karena memiliki nilai kurang dari batas yang ditentukan, yaitu 9000 lux.

**4.6 Pengujian Mode Otomatis pada Kipas**

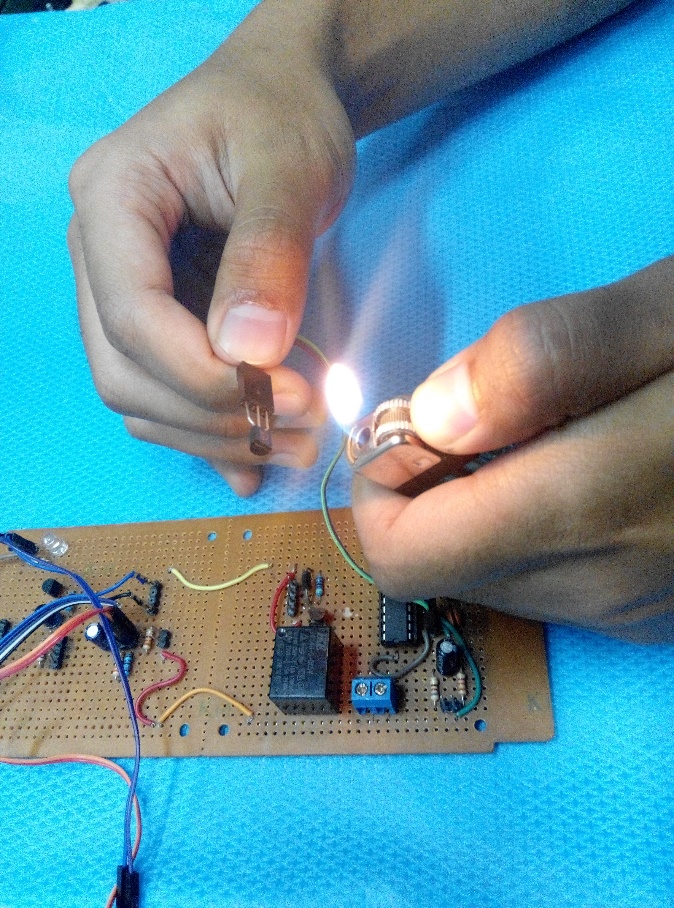
Pada pengujian ini dilakukan pengujian mode otomatis pada kipas yang memakai sensor suhu (LM-35) sebagai penentu menyala atau matinya kipas pada sistem. Percobaan ini dilakukan pada ruangan yang memiliki temperature sekitar 23 derajat celcius, dengan batas yang ditentukan sebesar 30 derajat celcius. Sistem ini bekerja dengan kondisi apabila temperatur yang diukur oleh sensor suhu adalah sebesar 30 derajat atau lebih besar, maka kipas pada sistem akan menyala, sebaliknya, jika temperatur yang diukur oleh sensor suhu adalah kurang dari 30 derajat celcius, maka kipas pada sistem akan mati.

Dilakukan 10 kali percobaan pada pengujian ini, dimana setiap percobaan dilakukan 2 kali pengetesan terhadap *hardware* sistem. Percobaan ini berupa menyalakan korek api ke arah sensor suhu untuk mendapatkan temperatur yang sama atau lebih besar dari 30 derajat celcius, dan mematikan korek api sehingga suhu yang diukur oleh sensor memiliki nilai yang lebih kecil dari 30 derajat celcius. Sehingga pengujian akan dikatakan berhasil apabila kipas pada sistem menyala ketika korek api dinyalakan, dan kipas pada sistem mati ketika korek api dimatikan.

Analisis yang dapat diberikan terhadap hasil dari pengujian ini adalah LM35 yaitu sensor temperatur memiliki tingkat reliabilitas yang cukup bagus. Dengan kondisi ruangan berAC yang di set pada temperature 23 derajat celcius, dan terlihat sensor LM35 pun menunjukkan pada angka sekitar 23 derajat, memiliki arti bahwa sensor tersebut bisa dikatakan cukup akurat. Dengan dinyalakannya korek api, lalu didekatkan dengan sensor tersebut, sudah jelas maka sensor akan membaca suhu yang ditangkap melebihi 30 derajat. Hal tersebut yang menyebabkan kipas pada sistem akan menyala secara otomatis karena membaca bahwa suhu ruangan pada kondisi yang cukup panas. Sementara itu, jika korek api dimatikan, maka sensor LM35 pun akan menangkap suhu ruangan pada kondisi awal, yaitu sekitar 23 derajat celcius, yang menyebabkan kipas pada sistem akan mati secara otomatis.

Tabel 4.9 Hasil Pengujian Mode Otomatis pada Kipas

|  |  |  |
| --- | --- | --- |
| Nomor Percobaan | Ketika korek dinyalakan | Ketika korek dimatikan |
| 1 | Berhasil | Berhasil |
| 2 | Berhasil | Berhasil |
| 3 | Berhasil | Berhasil |
| 4 | Berhasil | Berhasil |
| 5 | Berhasil | Berhasil |
| 6 | Berhasil | Berhasil |
| 7 | Berhasil | Berhasil |
| 8 | Berhasil | Berhasil |
| 9 | Berhasil | Berhasil |
| 10 | Berhasil | Berhasil |
| Tingkat Keberhasilan | 100% Berhasil | 100% Berhasil |



Gambar 4.11 Pengujian mode otomatis pada kipas

BAB V

SIMPULAN

1. Sistem pengolahan citra Color Code berhasil diintegrasikan dengan teknologi rumah pintar. Dengan *color blob* library OpenCV pada Java yang merupakan dasar untuk pengembangan algoritmanya.
2. Teknologi rumah pintar dengan fitur tambahan Color Code mampu untuk mempermudah kehidupan manusia dan dapat membantu dunia dalam pelestarian lingkungan.
3. Interval waktu antara input dan output dari program ini adalah sebesar 0.892 detik. Sementara pengujian Error Rate pada Color Code bernilai 0%.
4. Pengujian kuisioner berbentuk penilaian terhadap sistem yang terkait dengan efisiensi dan efektivitas dari program Color Code untuk integrasi sistem rumah pintar menghasilkan kualitas 100% berhasil pada *System Testing*, 79% kepuasan pada *Consistency Testing*, serta 73% kepuasan pada *User Interface* Testing.
5. Pengujian pada *hardware* berhasil dilakukan dengan tingkat keberhasilan 100% pada mode otomatis untuk kedua *hardware* sistem yaitu untuk lampu dan kipas.

REFERENSI

1. Android users, http://thenextweb.com/google/2014/01/07/android-will-pass-1-billion-users-across-devices-2014-according-gartner/. Diakses 5 Oktober 2014.
2. Nazruddin, Safaat H. Android; Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android. 2011. Bandung : Informatika.
3. Android versions. www.android.com/versions. Diakses 10 Oktober 2014
4. What is java. http://www.java.com. Diakses 2 Oktober 2014.
5. Detecting blobs using cvblobs library. http://8a52labs.com/2011/05/24/detecting-blobs-using-cvblobs-library/. Diakses 25 September 2014.
6. IOIO Documentation. https://github.com/ytai/ioio/wiki. Diakses 14 September 2014.
7. Design Electronics. http://www.designelectronics.net/. Diakses pada 15 Oktober 2014.
8. The Eclipse Foundation open source community website. www.eclipse.org/. Diakses pada 20 Oktober 2014.
9. Android-related code. http://stackoverflow.com/. Diakses pada 20 Oktober 2014.
10. Chong, Gao, dkk. “The Research and Implement of Smart Home System Based on Internet of Things”. 2011.
11. Donoser, Michael, Bischof, Horst, dan Wiltsche, Mario. "Color Blob Segmentation By MSER Analysis," 16/II, pp. 757-760, 2006.

## LAMPIRAN 1: Formulir Kuisioner

**USER TESTING FORM**

**SHROOM**

1. *System Testing*

Tabel 1 System Testing

|  |  |  |  |
| --- | --- | --- | --- |
| No | **System Testing** | **Checklist** | **Komentar** |
| 1 | Apakah aplikasi berhasil dijalankan? |  |  |
| 2 | Apakah mode manual berfungsi? |  |  |
| 3 | Apakah mode automatis berfungsi? |  |  |
| 4 | Apakah sensor suhu dapat berfungsi dengan baik? |  |  |
| 5 | Apakah sensor cahaya dapat berfungsi dengan baik? |  |  |
| 6 | Apakah Setup Color Code dapat dijalankan? |  |  |
| 7 | Apakah relay 1 dapat dikontrol secara manual dengan Color Code? |  |  |
| 8 | Apakah relay 2 dapat dikontrol secara manual dengan Color Code? |  |  |
| 9 | Apakah relay 3 dapat dikontrol secara manual dengan Color Code? |  |  |
| 10 | Apakah relay 4 dapat dikontrol secara manual dengan Color Code? |  |  |

Sistem harus dapat memberikan informasi kepada user tentang apapun yang sedang terjadi pada sistem melalui respon dan waktu yang tepat.

Jawab dengan cara memberikan tanda silang (X) jika tidak berfungsi dengan baik atau tanda centang (✓) jika berfungsi dengan baik.

2. *Consistency Testing*

Sistem seharusnya menggunakan penamaan yang standard dan konsisten pada keseluruhan sistem, agar tidak menimbulkan ambiguitas ketika digunakan.

Tabel 2 *Consistency Testing*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | **Consistency Testing** | **1** | **2** | **3** | **4** |
| 1 | Apakah setiap activity mudah untuk dimengerti? |  |  |  |  |
| 2 | Apakah Anda mengerti dengan hanya membaca judul disetiap halamannya? |  |  |  |  |
| 3 | Apakah setiap tombol memiliki konsistensi dan tidak menimbulkan ambiguitas? |  |  |  |  |

Jawab dengan cara memberikan tanda silang (X) atau centang (✓) pada satu dari empat pilihan, dengan kondisi 1 = tidak sesuai, 2 = sedikit tidak sesuai, 3 = sedikit sesuai, 4 = sesuai.

3. *User Interface Testing*

Desain yang dibuat harus mudah dimengerti dan menarik perhatian pengguna.

Tabel 3 *User Interface Testing*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | **UI Testing** | **1** | **2** | **3** | **4** |
| 1 | Apakah tampilan aplikasi menarik? |  |  |  |  |
| 2 | Apakah tombol gambar pada menu utama yang digunakan cukup representatif? |  |  |  |  |
| 3 | Apakah tiap tombol pada tiap fitur kontrol alat elektronik mudah dimengerti? |  |  |  |  |
| 4 | Apakah jenis huruf dan ukurannya nyaman untuk dibaca? |  |  |  |  |

Jawab dengan cara memberikan tanda silang (X) atau centang (✓) pada satu dari empat pilihan, dengan kondisi 1 = tidak sesuai, 2 = sedikit tidak sesuai, 3 = sedikit sesuai, 4 = sesuai.