LEMBAR PENGESAHAN

AUTOMATIC BACKGROUND UNTUK STUDIO FOTO DENGAN PENGENDALI NIRKABEL BERBASIS BLUETOOTH

Oleh :

Dwi Arta Putra	NIRM	0031737
Ervina Sugianti	NIRM	0031738

Laporan ini terlah disetujui dan disahkan sebagai salah satu syarat kelulusan Program Diploma III Politeknik Manufaktur Negeri Bangka Belitung

Menyetujui,

Pembimbing 1	Pembimbing 2
Surojo, M. T	Yudhi, M.T

Penguji 1	Penguji 2	Penguji 3
Aan Febrianyah, M.T	Ocsirendi, M.T	Charlotha, M.Tr.T

PERNYATAAN BUKAN PLAGIAT

Yang bertanda tangan dibawah ini :

Nama Mahasiswa 1 : Dwi Arta Putra NIRM : 0031737
Nama Mahasiswa 2 : Ervina Sugianti NIRM : 0031738

Dengan Judul : Automatic Background Untuk Studio Foto Dengan Pengendali Nirkabel Berbasis Bluetooth

Menyatakan bahwa laporan akhir ini adalah hasil kerja kami sendiri dan bukan merupakan plagiat. Pernyataan ini kami buat dengan sebenarnya dan bila ternyata dikemudian hari ternyata melanggar pernyataan ini, kami bersedia menerima sanksi yang berlaku.

Sungailiat, 19 Agustus 2020

	Nama Mahasiswa	Tanda Tangan
1.	Dwi Arta Putra	.....
2.	Ervina Sugianti	.....

ABSTRAK

Pada umumnya di studio fotografi menggunakan latar belakang dengan gambar pemandangan atau warna tertentu yang dapat diubah-ubah sesuai dengan keinginan. Untuk proses pergantian latar biasanya memakan banyak tenaga dan waktu, terutama di studio yang masih menggunakan cara konvensional untuk mengganti latar belakang foto sehingga dianggap kurang praktis. Maka dari itu, untuk mempermudah proses pergantian warna latar dibuatlah Automatic Background yang dapat membantu proses pergantian warna latar menjadi lebih cepat dan praktis. Alat ini dilengkapi dengan konstruksi mekanis berukuran 230 x 130 mm dengan pengontrol aplikasi android jarak jauh yang didukung oleh komponen-komponen elektronika seperti sensor TCS230, modul Bluetooth HC-05 sebagai komunikasi data, motor wiper sebagai penggeraknya dan modul relay 4 channel untuk mengatur kecepatan motor. Komponen-komponen tersebut diintegrasikan oleh mikrokontroler Arduino UNO, menggunakan software Arduino IDE sebagai pemrogramannya dan software Android Studio IDE untuk membuat aplikasi android sebagai pengontrol jarak jauh. Alat ini dapat dioperasikan secara otomatis dengan menampilkan warna latar yang diinginkan setelah tombol ditekan dengan kecepatan pergerakan latar yang dapat diubah-ubah.

Kata Kunci : Background Studio Foto, Pengendali Jarak Jauh, Modul Bluetooth HC-05, Sensor TCS-230, Android Studio IDE

ABSTRACT

In general, a photography studio uses a background with a certain landscape image or color that can be changed as desired. The process of changing the background usually takes a lot of effort and time, especially in studios that still use conventional methods to change the background of the photo so it is considered impractical. Therefore, to simplify the process of changing the background color, an Automatic Background is created which can help the process of changing the background color faster and more practical. This tool is equipped with a mechanical construction measuring 230 x 130 mm with an android remote application controller supported by electronic components such as TCS230 sensor, HC-05 Bluetooth module as data communication, a wiper motor as the driving force and a 4 channel relay module to regulate motor speed . These components are integrated by the Arduino UNO microcontroller, using Arduino IDE software as programming and Android Studio IDE software to create the android application as a remote controller. This tool can be operated automatically by displaying the desired background color after the button is pressed with the background speed that can be changed.

Keywords: Photo Studio Background, Remote Control, Bluetooth Module HC-05, TCS-230 Sensor, Android Studio IDE

KATA PENGANTAR

Puji dan syukur penulis ucapkan kepada Allah SWT, karena berkat dan rahmat-Nyalah penulis dapat menyelesaikan laporan akhir ini. Serta Shalawat beserta salam penulis ucapkan kepada Rasulullah Muhammad SAW, yang telah membawa umat manusia ke zaman yang terang dan penuh ilmu pengetahuan.

Proyek akhir "*Automatic Background* untuk studio foto dengan pengendali nirkabel berbasis bluetooth" merupakan salah satu syarat wajib setiap kelompok kerja proyek akhir untuk memenuhi persyaratan pendidikan Diploma III Politeknik Manufaktur Negeri Bangka Belitung.

Karya tulis ini berisikan hasil penelitian yang penulis lakukan selama program Proyek Akhir berlangsung. Adanya media pembelajaran ini diharapkan dapat membantu mahasiswa.

Pada kesempatan ini penulis ingin menyampaikan rasa terima kasih kepada semua pihak yang telah banyak membantu dalam penyelesaian proyek akhir ini :

1. Orang tua dan keluarga penulis yang telah banyak memberikan dukungan baik materil maupun moril serta diiringi doa.
2. Bapak I Made Andik Setiawan, M.Eng, Ph.D selaku Direktur Politeknik Manufaktur Negeri Bangka Belitung.
3. Bapak Eko Sulistyo, M.T. selaku Ka. Prodi D3 Teknik Elektronika
4. Bapak Surojo, M T. selaku pembimbing I.
5. Bapak Yudhi, M.T. selaku pembimbing II.
6. Seluruh dosen dan instruktur yang telah banyak membantu dalam penyelesaian proyek akhir ini.
7. Rekan seangkatan dari semua prodi yang turut membantu dalam penyelesaian proyek akhir ini.
8. Orang-orang terdekat yang telah memberikan semangat dan dukungan serta doa bagi penulis.

Penulis menyadari bahwa laporan proyek akhir ini jauh dari kata sempurna, terutama dalam segi isi maupun rancangan karena keterbatasan waktu dan hambatan yang penulis hadapi. Oleh karena itu penulis sangat mengharapkan saran dan kritik dari pembaca agar dapat lebih baik kedepannya.

Besar harapan penulis semoga karya tulis ini dapat memberikan manfaat dan motivasi bagi pembaca khususnya dan baik bagi perkembangan ilmu pengetahuan.

Sungailiat, 19 Agustus 2020

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Otomatisasi pada pekerjaan telah merambah ke hampir segala aspek kehidupan manusia. Dengan adanya otomatisasi pekerjaan, maka pekerjaan manusia dapat menjadi lebih ringkas baik dari segi pengendalian dan penggunaanya.

Pada umumnya di studio fotografi menggunakan latar belakang dengan pemandangan atau warna tertentu yang dapat diubah-ubah sesuai dengan selera dan keinginan pelanggan.

Oleh karena itu terdapat proses dimana sang juru kamera harus mengganti latar belakang sesuai dengan permintaan pelanggan, rata-rata ukuran latar belakang foto di studio ialah $4x3m^2$ yang mana proses tersebut biasanya memakan banyak tenaga dan waktu terutama di studio yang masih menggunakan cara konvensional untuk mengganti latar belakang foto, yang mana pada proses penggantian warnanya dilakukan dengan cara melepas atau memasangkan kain latar berwarna secara manual, sedangkan di Lea *Photo Studio* yang beralamat di Jl. Jendral Sudirman No. 150 Ruko Pondok Mulya, Sungailiat telah menggunakan sistem perangkat teknologi semi otomatis yang mana perangkat tersebut telah dipatenkan oleh Gary D. Clegg pada tanggal 20 Januari 2009 dengan judul *Rapidly Changeable Photographic and Stage Backdrop Lifting Device* namun perangkat tersebut memiliki beberapa kekurangan yakni,

1. Sistem pengendali, teknologi tersebut hanya memiliki satu buah pengontrol berupa pengendali yang terhubung dengan perangkat melalui kabel, apabila pengendali tersebut rusak maka perangkat tidak dapat digunakan.
2. Kurang Praktis, untuk mengganti latar belakang foto, maka juru fotografi harus menekan tombol pada pengendali secara terus menerus sampai

gambar latar belakang tampil dengan sempurna dan penggerek akan berhenti bergerak apabila tombol pengendali dilepas.

3. Proses awal pemasangan perangkat yang rumit, untuk memasang perangkat ini, diperlukan proses pengeboran pada dinding ruangan yang mana akan mengakibatkan permukaan dinding jadi cacat berlubang bahkan dapat mengakibatkan dinding menjadi retak agar perangkat dapat menempel pada dinding.

Dikarenakan perihal yang telah dijelaskan diatas maka akan lebih baik bila dirancang bangun sebuah perangkat yang dapat membantu menyelesaikan masalah tersebut, oleh karena itu pada proyek akhir ini akan merancang penggerak latar belakang foto otomatis dengan kendali tanpa kabel yang akan membantu dan mempermudah sang juru foto dalam proses penggantian latar belakang foto.

1.2 Rumusan Masalah

Masalah yang timbul adalah sebagai berikut :

1. Bagaimana agar kerja *automatic background* menjadi lebih efisien dari segi kecepatan waktu pada pergantian tiap warna dibandingkan dengan *background semi* otomatis dan *background* konvensional?
2. Bagaimana agar *automatic background* dapat dikendalikan melalui perangkat android *?*

1.3 Tujuan

Tujuan dari pembuatan alat ini adalah

1. Membantu proses penggantian latar belakang foto menjadi lebih cepat dan praktis.
2. Membuat alat berupa konstruksi mekanis yang mampu mendeteksi pergantian warna menggunakan sensor TCS-230.
3. Membuat *automatic background* dapat dikendalikan secara nirkabel melalui perangkat *android.*

BAB II
DASAR TEORI

2.1 Background

Background atau *backdrop* adalah latar belakang yang digunakan untuk membantu pada proses pemotretan foto di studio fotografi dengan berbagai macam gambar, pola dan warna. Salah satu contoh *background* yang sering digunakan ialah jenis *background* menggunakan *stand* dan *background semi*-otomatis yang menempel di dinding.

Untuk gambar contoh jenis-jenis *background* dapat dilihat pada gambar 2.1 dan 2.2 dibawah:

Gambar 2. 1 *Background* konvensional dengan *stand.*

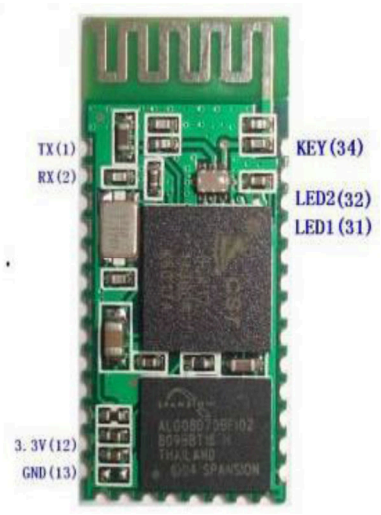
Gambar 2. 2 *Background* semi otomatis

2.2 Bluetooth

Bluetooth adalah protokol komunikasi *wireless* yang bekerja pada frekuensi radio 2.4 GHz untuk pertukaran data pada perangkat bergerak seperti PDA, laptop, HP, dan lain-lain.

Modul *Bluetooth* HC-05 merupakan salah satu modul *Bluetooth* yang dapat ditemukan dipasaran dengan harga yang realtif murah. Modul *Bluetooth* HC-05 terdiri dari 6 pin konektor, yang setiap *input* konektor memiliki fungsi yang berbeda-beda. (1)

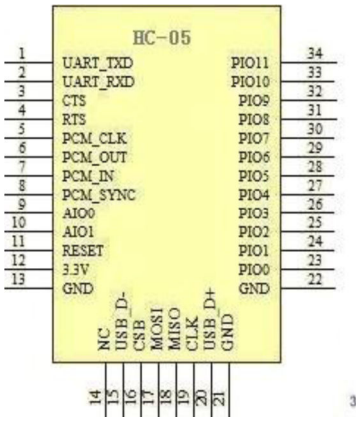
Untuk gambar *module bluetooth* dapat dilihat pada gambar 2.3 dibawah ini:



Gambar 2. 3 Modul *Bluetooth HC-05*

Modul *bluetooth* HC-05 dengan *supply* tegangan sebesar 3,3 V ke pin 12 modul *bluetooth* sebagai VCC. Pin 1 pada modul *Bluetooth* sebagai *transmitter,* kemudian pin 2 pada *Bluetooth* sebagai *Receiver*. (1)

Berikut merupakan konfigurasi pin *Bluetooth HC-05* ditunjukkan pada gambar 2.4 dibawah ini :



Gambar 2. 4 Konfigurasi Pin *Bluetooth HC-05.*

Konfigurasi pin modul *Bluetooth HC-05* dapat dilihat pada tabel 2.1 berikut ini :

Tabel 2. 1 *Bluetooth-to-Serial-Module HC-05*

No	Nomor Pin	Nama	Fungsi
1	Pin 1	Key	-
2	Pin 2	VCC	Sumber tegangan 5V
3	Pin 3	GND	Ground tegangan
4	Pin 4	TXD	Mengirim data
5	Pin 5	RXD	Menerima data
6	Pin 6	STATE	-

Module Bluetooth HC-05 merupakan *Module Bluetooth* yang bisa menjadi *slave* ataupun *master* hal ini dibuktikan dengan bisa memberikan notifikasi untuk melakukan *pairing* ke perangkat lain, maupun perangkat lain tersebut yang melakukan *pairing* ke *module Bluetooth HC-05*. Untuk mengeset perangkat *Bluetooth* dibutuhkan perintah-perintah AT *Command* yang mana perintah AT *Command* tersebut akan di respon oleh perangkat *Bluetooth* jika modul *Bluetooth*

tidak dalam keadaan terkoneksi dengan perangkat lain. Tabel 2.2 dibawah adalah table AT *Command Module Bluetooth H*C-05. (1)

Keterangan AT *Command Module Bluetooth* HC-05 dapat dilihat pada table 2.2 berikut:

Tabel 2. 2 *AT Command Module Bluetooth* HC-05

No	Perintah	Kirim	Terima	Keterangan
1	Test Komunikasi	AT	ON	-
2	Ganti Nama Bluetooth	AT+NAMEnamaBT	OKnamaBT	-
3	Ubah Pin Code	AT+PINxxxx	OKsetpin	Xxxx digit key
4	Ubah Baudrate	AT+BAUD1	OK1200	1 ------- 1200
		AT+BAUD2	OK2400	2 ------- 2400
		AT+BAUD3	OK4800	3 ------- 4800
		AT+BAUD4	OK9600	4 ------- 9600
		AT+BAUD5	OK19200	5 -------19200
		AT+BAUD6	OK384000	6 ------- 38400
				7 -------57600
				8 ------- 115200

2.3 Smartphone

Smartphone adalah telepon genggam yang mempunyai kemampuan dengan penggunaan dan fungsi yang menyerupai komputer. Belum ada standar pabrik yang menentukan arti *Smartphone*. Bagi beberapa orang, *smartphone* merupakan telepon yang bekerja menggunakan seluruh perangkat lunak sistem operasi yang menyediakan hubungan standar dan mendasar bagi pengembang aplikasi. Dengan kata lain, *smartphone* merupakan komputer kecil yang mempunyai kemampuan sebuah telepon. Pertumbuhan permintaan akan alat canggih yang mudah dibawa ke mana-mana membuat kemajuan besar dalam

pemroses, peringatan, layar dan sistem operasi yang di luar dari jalur telepon genggam sejak beberapa tahun ini. (2)

Belum ada kesepakatan dalam industri ini mengenai apa yang membuat telepon menjadi "pintar", dan pengertian dari *smartphone* itu pun berubah mengikuti waktu. Menurut David Wood, Wakil Presiden Eksekutif PT *Symbian OS*,"*Smartphone* dapat dibedakan dengan telepon genggam biasa dengan dua cara fundamental, yakni bagaimana mereka dibuat dan apa yang mereka bisa lakukan." Pengertian lainnya memberikan penekanan perbedaan dari dua faktor ini. (2)

Kebanyakan alat yang dikategorikan sebagai *Smartphone* menggunakan sistem operasi yang berbeda. Dalam hal fitur, kebanyakan *smartphone* mendukung sepenuhnya fasilitas surel dengan fungsi pengatur personal yang lengkap. Fungsi lainnya dapat menyertakan miniatur papan ketik QWERTY, layar sentuh atau *D-Pad*, kamera, pengaturan daftar nama, penghitung kecepatan, navigasi piranti lunak dan keras, kemampuan membaca dokumen bisnis, pemutar musik, penjelajah foto dan melihat klip *video*, penjelajah *internet* atau hanya sekedar akses aman untuk membuka surel perusahaan, seperti yang ditawarkan oleh *BlackBerry*. (2)

Sistem operasi yang dapat ditemukan di *smartphone* adalah *Symbian OS*, *iOS*, *RIM Blackberry*, *Windows mobile*, *Linux, Palm*, *WebOS* dan *Android*. (2)

2.4 Android Studio

Android Studio adalah lingkungan pengembangan terpadu *Integrated Development Environment* (IDE) untuk pengembangan aplikasi android, berdasarkan *IntelliJ IDEA*. Selain merupakan editor kode *IntelliJ* dan alat pengembang yang berdaya guna, Android Studio menawarkan fitur lebih banyak untuk meningkatkan produktivitas anda saat membuat aplikasi android, (3) misalnya

1. Sistem versi berbasis *Gradle* yang fleksibel.
2. Emulator yang cepat dan kaya fitur.
3. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat android.

4. *Instant Run* untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru.
5. *Template* kode dan *Integrasi* GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh.
6. Alat penguji dan kerangka kerja yang ekstensif
7. Alat Lint untuk meningkatkan kinerja, kegunaan, kompatibilitas versi, dan masalah-masalah lain.
8. Dukungan C++ dan NDK
9. Dukungan bawaan untuk *Google Cloud Platform*, Mempermudah pengintegrasian *Google Cloud Messaging* dan *App Engine.*
10. Laman ini berisi pengantar dasar fitur-fitur *Android Studio*. Untuk memperoleh rangkuman perubahan terbaru, lihat catatan rilis *Android Studio*. (3)

2.5 Arduino Uno

Uno Arduino adalah *board* berbasis mikrokontroler pada *ATmega328. Board* ini memiliki 14 digital input / output pin (dimana 6 pin dapat digunakan sebagai *output PWM*), 6 input analog, 16 MHz osilator kristal, koneksi *USB, jack* listrik tombol *reset*. Pin-pin ini berisi semua yang diperlukan untuk mendukung mikrokontroler, hanya terhubung ke komputer dengan kabel *USB* atau sumber tegangan bisa didapat dari adaptor AC-DC atau baterai untuk menggunakannya (4).

Berikut merupakan gambar *board Arduino Uno* yang dapat dilihat pada gambar 2.5 dibawah ini:

Gambar 2. 5 *Arduino Uno*

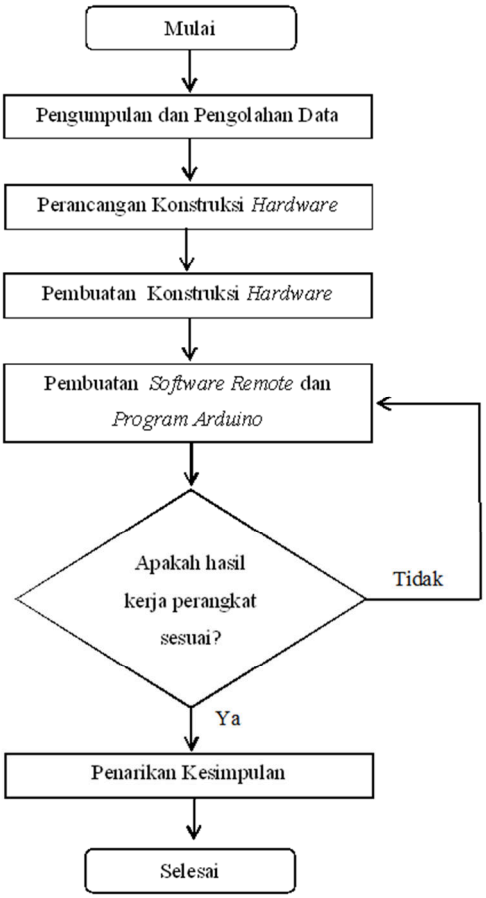
Sedeangkan untuk melihat deskripsi atau spesifikasi dari *Arduino uno* dapat dilihat pada tabel 2.3 dibawah ini:

Tabel 2. 3 Deskripsi *Arduino Uno*

Mikrokontroller	Atnega 328
Operasi *Voltage*	5V
Input Voltage	7-12V (Rekomendasi)
Input Voltage	6-20V(Limits)
I/O	14 pin (6 pin untuk PWM)
Arus	50mA
Flash Memory	32KB
Bootloader	SRAM 2KB

BAB III
METODE PELAKSANAAN

Metode pelaksanaan yang digunakan dalam proyek akhir ini adalah dengan merancang kegiatan-kegiatan dalam bentuk diagram alir menurut VDI 2222, dengan tujuan agar tindakan yang dilakukan lebih terarah dan terkontrol sehingga target-target yang diharapkan dapat tercapai. Diagram alir ini dapat dilihat pada Gambar 3.1 dibawah ini:



Gambar 3. 1 Diagram Alir Metode Pelaksanaan

3.1 Tahapan Persiapan

Pada tahapan persiapan, dilakukan proses pengamatan dan penelitian mengenai alat yang akan dirancang dan dibuat guna untuk menyelesaikan program tugas akhir. Selanjutnya dari hasil pengamatan dan penelitian nantinya

dilakukan perbandingan untuk menentukan perencanaan mengenai produk yang akan diproses.

3.2 Metode Pengumpulan Data

Pengumpulan data ini berfungsi untuk mengetahui kebutuhan-kebutuhan pasar tentang suatu jenis mesin atau mencari kemungkinan-kemungkinan untuk melakukan modifikasi terhadap alat yang sudah ada. Data yang sudah terkumpul nantinya akan dimasukan dalam penulisan makalah sebagai landasan dalam pembuatan alat tersebut. Metode pengumpulan data yang diterapkan antara lain metode *interview*, studi pustakaan, dan *survey* lapangan.

1) Metode Studi Pustaka

Untuk menunjang pembuatan alat ini dikumpulkan data dari berbagai sumber yang terkait dengan masalah-masalah yang akan dibahas. Sumber yang diambil adalah referensi buku.

2) *Survey* Lapangan

Yaitu melakukan pengamatan langsung terhadap proses pergantian dari setiap latar dan proses pemasangan atau perakitan pada *background manual* di *studio* fotografi. Selain itu dilakukan pemahaman tentang masalah yang timbul dan kendala yang terjadi.

3.3 Perencanaan

Perencanaan yang dilakukan dengan menganalisa proyek akhir yang dibuat sehingga diperoleh pokok-pokok bagian yang akan diperhitungkan berdasarkan target yang dicapai sesuai dengan data-data yang diperoleh dari hasil pengumpulan data dan berdasarkan alternatif pilihan.

Dalam melakukan perancangan proyek akhir ini harus mengetahui proses yang akan dilakukan sehingga hasil yang didapatkan lebih maksimal dan sebaiknya menggunakan metode perancangan agar dapat mengurangi kesalahan pada proses pembuatannya kelak dan agar proses pembuatan proyek akhir tersebut dapat lebih terorganisir.

3.4 Pengadaan Bahan dan Komponen

Setelah dilakukan perencanaan, tahap berikutnya adalah melakukan pengadaan bahan dan komponen. Dalam hal pengadaan ini, harus disesuaikan dengan tahap perencanaan untuk mendapatkan hasil yang diinginkan.

3.5 Pembuatan *Hardware*

Pembuatan *hardware* dilakukan berdasarkan rancangan yang telah di analisis dan diperhitungkan sehingga mempunyai arah yang jelas dalam proses pembuatannya. Setelah perencanaan, pengadaan bahan dan komponen selesai maka tahap awal pembuatan *hardware* yaitu:

1. Pembuatan konstruksi *Background*
2. Pembuatan *box* komponen
3. Pemasangan komponen-komponen yang digunakan

3.6 Pembuatan *Software dan Program*

Pembuatan *software* menggunakan *software Android Studio* dan diunduh ke *Smartphone Android* untuk pembuatan program menggunakan *software Arduino* lalu diunduh ke *Arduino*. Pada pembuatan *software dan program* harus terlebih dahulu memahami proses kerja *hardware*, sehingga dapat mengurangi kesalahan-kesalahan yang akan terjadi. Tahap akhir dari pembuatan alat ini ialah penggabungan antara *software* dan *hardware* menjadi suatu sistem yang utuh dan siap dioperasikan.

3.7 Uji Coba (*Trial*)

Dalam suatu percobaan alat biasanya mengalami *trial and error* sehingga sebelum dilakukan proses percobaan alat sebaiknya dipersiapkan semaksimal mungkin baik *hardware* dan *software* yang akan dicoba sehingga pada saat uji coba alat dapat bekerja sesuai dengan yang diinginkan. Apabila dalam uji coba mengalami gangguan (*error*) sehingga alat tidak bekerja sesuai yang diinginkan maka proses berikutnya adalah perbaikan pada sistem yang mengalami gangguan

tersebut. Setelah itu dilakukan uji coba kembali, jika berhasil bekerja sesuai dengan yang diinginkan maka pembuatan selesai.

Uji coba alat (*Trial*) dilakukan sebagai tolak ukur berhasil atau tidaknya mesin yang dibuat. Dengan begitu, dapat dievaluasi kualitas dari alat yang dibuat.

3.8 Perbaikan

Tahap ini dilakukan apabila hasil yang ada tidak sesuai dengan yang diharapkan, maka langkah yang harus dilakukan ialah menganalisa kerusakan lalu diteruskan dengan perbaikan pada sistem yang mengalami gangguan.

3.9 Kesimpulan

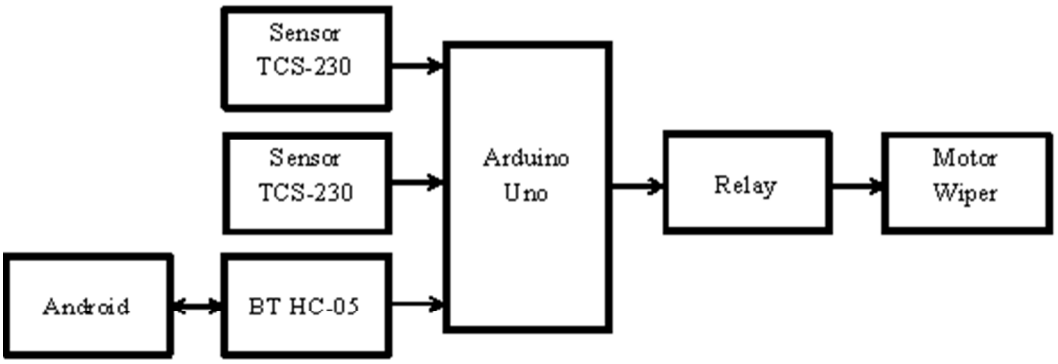
Setelah melewati tahap uji coba dan dinyatakan sesuai dengan perencanaan, maka peralatan yang penulis buat akan dianalisa dengan melakukan pengamatan terhadap kelayakan sistem kerja dari alat tersebut. Hasil pengamatan ini akan menentukan kelebihan dan kekurangan alat yang telah dibuat, serta mencari solusi untuk menutupi kekurangan-kekurangan tersebut sehingga menjadi suatu kelebihan. Dari hasil inilah yang kemudian akan dibuat laporan beserta kesimpulannya.

BAB IV
PEMBAHASAN

Pada bab ini akan dibahas mengenai proses serta metode yang digunakan dalam pembuatan alat proyek akhir dengan judul "*Automatic Background Roller* dengan *Bluetooth Control System*" yaitu sebagai berikut.

4.1 Blok *Diagram Hardware*

Berikut merupakan blok *diagram hardware* dari *Automatic Background* Untuk Studio Foto Dengan Pengendali Nirkabel Berbasis *Bluetooth* pada gambar 4.1 dibawah:

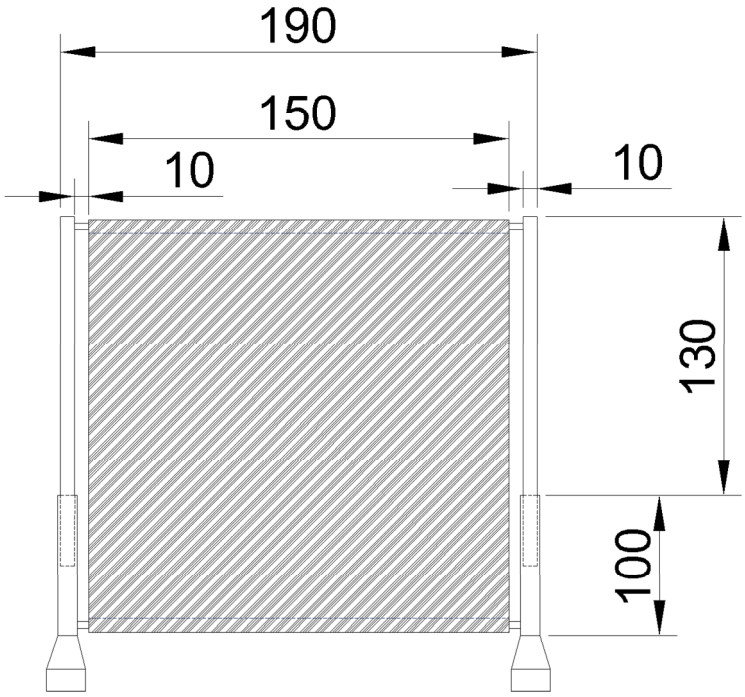


Gambar 4. 1 Blok *Diagram Hardware*

Pada blok diagram *hardware* diatas dapat dilihat bahwa terdapat *input*, proses dan o*utput*. Untuk *input* nya menggunakan *android* yang kemudian datanya akan melalui modul *Bluetooth HC 05* dan dua buah sensor warna TCS230. Lalu untuk mengolah datanya digunakan *Arduin*o uno dan relay sebagai *processing* nya. Maka, setelah data selesai diproses, *output* berupa *motor wiper* dapat dijalankan.

4.2 Konstruksi *Automatic Background*

Perancangan konstruksi adalah proses pembuatan desain *Automatic Background* . Proses ini dibuat secara bertahap dimulai dari perancangan gambar 4.2 dibawah:



Gambar 4. 2 Konstruksi *hardware*

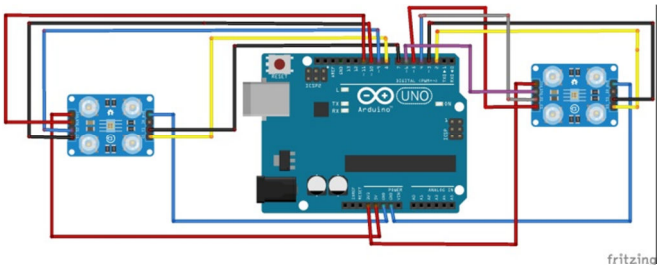
4.3 *Hardware* Elektrik *Automatic Background*

4.3.1 Sensor TCS 230

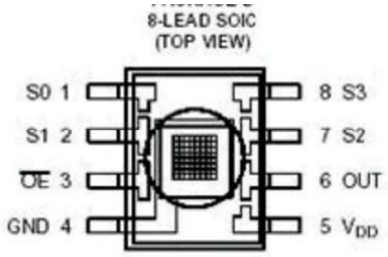
Sensor warna TCS230 adalah sensor warna yang sering digunakan pada aplikasi mikrokontroler untuk pendeteksian suatu objek benda atau warna sari objek yang di *monitor*. Sensor warna TCS230 juga dapat digunakan sebagai sensor gerak, dimana sensor mendeteksi gerakan suatu objek berdasarkan perubahan warna yang diterima oleh sensor. Pada dasarnya sensor warna TCS230 adalah rangkaian photo dioda yang disusun secara matrik *array* 8x8 dengan 16 buah konfigurasi photodioda yang berfungsi sebagai *filter* warna merah, 16 photodioda sebagai *filter* warna biru dan 16 photodioda lagi tanpa *filter* warna. Sensor warna

TCS230 merupakan sensor yang dikemas dalam *chip* DIP 8 pin dengan bagian muka transparan sebagai tempat menerima intensitas cahaya yang berwarna (5).

Berikut merupakan gambar 4.3 yang menunjukkan skema pemasangan pin sensor warna TCS230 dengan *Arduino,* dan gambar 4.4 untuk skema pin sensor tersebut.



Gambar 4. 3 Skema pemasangan pin sensor TCS230 ke *Arduino*



Gambar 4. 4 Skema pin sensor TCS230

Untuk mengetahui hasil pengujian dari sensor TCS 230, maka dapat dilihat dari tabel 4.1 dan tabel 4.2 dibawah:

Tabel 4. 1 Hasil Uji Sensor TCS230 tanpa Program.

Warna	Jarak(cm)								
	1			2			3		
	R	G	B	R	G	B	R	G	B
Merah	2	22	16	9	28	20	22	48	34
Hijau	18	10	12	23	14	16	33	22	22
Biru	19	11	5	15	22	8	40	29	15

Warna	Jarak(cm)								
	4			7			9		
	R	G	B	R	G	B	R	G	B
Merah	29	56	39	37	61	40	-	-	-
Hijau	-	-	-	-	-	-	-	-	-
Biru	35	27	15	52	48	29	54	55	35

Tabel 4. 2 Hasil Uji Sensor TCS230 dengan Program.

Warna	Jarak(cm)								
	1			2			3		
	R	G	B	R	G	B	R	G	B
Merah	0	20	16	7	26	20	22	48	34
Hijau	19	14	12	24	17	16	32	18	20
Biru	21	12	7	16	22	8	40	29	14

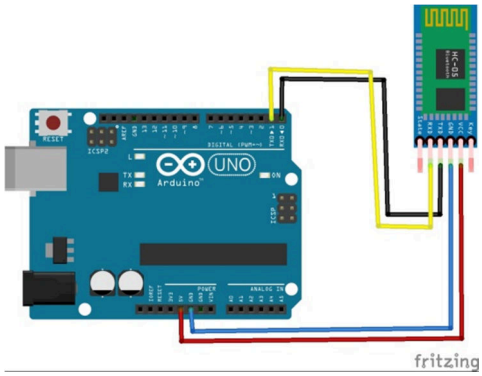
Warna	Jarak(cm)								
	4			7			9		
	R	G	B	R	G	B	R	G	B
Merah	26	58	37	34	58	44	-	-	-
Hijau	-	-	-	-	-	-	-	-	-
Biru	33	28	12	50	46	31	56	55	33

Berdasarkan hasil tabel 4.1 dan tabel 4.2 diatas maka dapat disimpulkan bahwa warna biru merupakan warna dengan jarak paling jauh yang dapat dideteksi oleh sensor tcs-230 ini. Sedangkan warna hijau sudah tidak dapat dideteksi nilai warnamya dimulai dari jarak 3cm dan untuk warna merah tidak dapat dideteksi nilai warnanya dimulai dari jarak 8cm. Semakin jauh jarak suatu warna maka nilai yang akan dihasilkan dari sensor

akan semakin besar juga, begitu pun sebaliknya jika jarak antar warna dan sensor tersebut semakin dekat maka nilai yang diperoleh dari warna yang terbaca akan semakin kecil.

4.3.2 *Module Bluetooth HC-05*

Pengujian *Module Bluetooth HC-05* bertujuan untuk mengetahui apakah modul tersebut berfungsi dan dapat dikoneksikan dengan *smartphone android* sebagai penghubung jalur komunikasi antara *smartphone android* dan *arduino.* Berikut adalah skematik pemasangan pin *module bluetooth HC-05* ke *arduino,* dapat dilihat pada gambar 4.5:



Gambar 4. 5 Skematik pemasangan pin *Module Bluetooth* ke Arduino

Kemudian pada tabel 4.3 dibawah, dapat dilihat bahwa *module bluetooth* dihubungkan ke *power supply* pada pin 3.3V dan pin GND, kemudian untuk proses penerimaan dan pengiriman data dihubungkan di pin 1 dan pin 2 pada *arduino.* Ketika rangkaian sudah terhubung maka akan ditandai dengan lampu *LED* berwarna merah pada modul *Bluetooth* yang berkedip dan menyala.

Tabel 4. 3 Module *Bluetooth* pada Pin Kontrol *Arduino*

Module *Bluetooth*	Arduino *Pin Control*
VCC	3.3 V
GND	GND
TX	Pin 1

RX	Pin 2

Kemudian, untuk mengetahui apakah modul tersebut dapat bekerja dengan baik maka dilakukan pengujian koneksi dari *smartphone android* ke *module bluetooth* dengan jarak tertentu yang mana hasilnya dapat dilihat pada tabel 4.4

Tabel 4. 4 Hasil Pengujian Koneksi *Bluetooth*

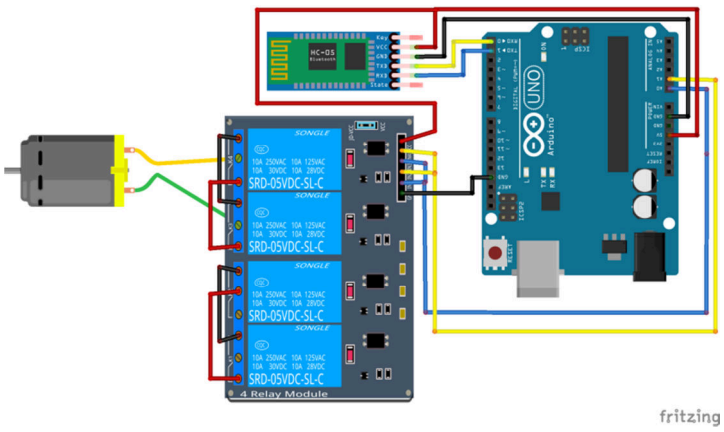
Jarak (m)	**Hasil**
1	Terdeteksi
2	Terdeteksi
3	Terdeteksi
4	Terdeteksi
5	Terdeteksi
6	Terdeteksi
7	Terdeteksi
8	Terdeteksi
9	Tidak Terdeteksi

Berdasarkan tabel diatas, hasil pengujian menunjukkan bahwa, modul *bluetooth* dapat menerima sinyal *trigger* dimulai dari jarak ± 1m sampai dengan jarak ± 8 m namun kesulitan dan bahkan tidak dapat menerima sinyal *trigger* lewat dari jarak 9 m.

4.3.3 *Motor Wiper*

Pengujian ini bertujuan untuk mengetahui bahwa *motor wiper* yang digunakan dapat berfungsi dengan baik, pengujian ini dilakukan dengan cara menghubungkan *motor wiper* dengan modul *relay* yang sudah terhubung dengan arduino dan modul *bluetooth* kemudian motor diuji melalui *smartphone android* yang sudah terhubung ke modul *bluetooth*.

Gambar 4.5 berikut merupakan gambar skema *motor wiper* yang telah dipasangkan pada modul *relay* dan modul *bluetooth*:



Gambar 4. 6 Skema Pemasangan *Motor Wiper* ke *Module Relay* dan *Arduino*

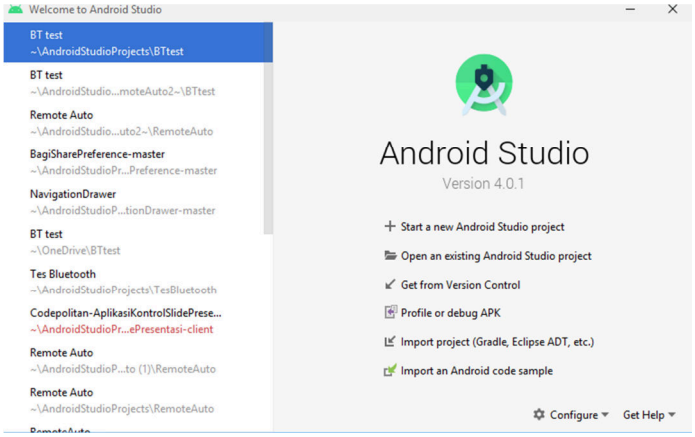
4.4 Aplikasi *Remote Controller*

Setelah bagian *hardware* telah dibuat, maka dilanjutkan dengan proses pembuatan aplikasi *android* pengendali nirkabel dengan menggunakan *software Android Studio IDE.*

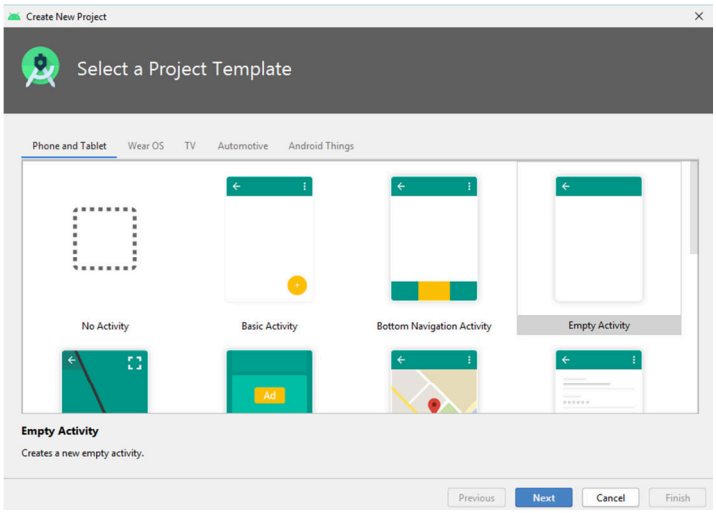
Berikut merupakan langkah-langkah yang harus dilakukan untuk membuat aplikasi tersebut.

1. *Install software Android Studio* di laptop/komputer dengan spesifikasi RAM minimal 3GB, namun disarankan untuk memakai laptop/komputer dengan RAM 8GB atau lebih.

2. Setelah *software* ter-*install*, silahkan buka *software* tersebut kemudian klik *"Start a new Android Studio Project"*.

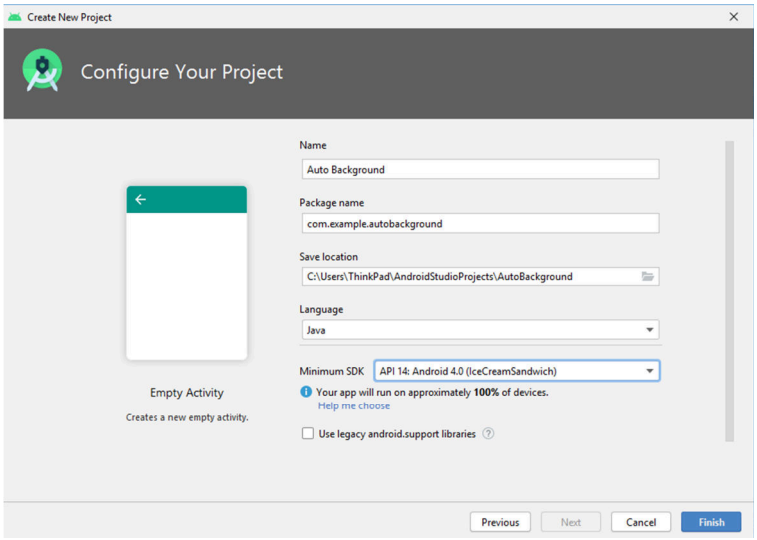


3. Lalu pilih "*Empty Activity*" kemudian klik "*Next*".

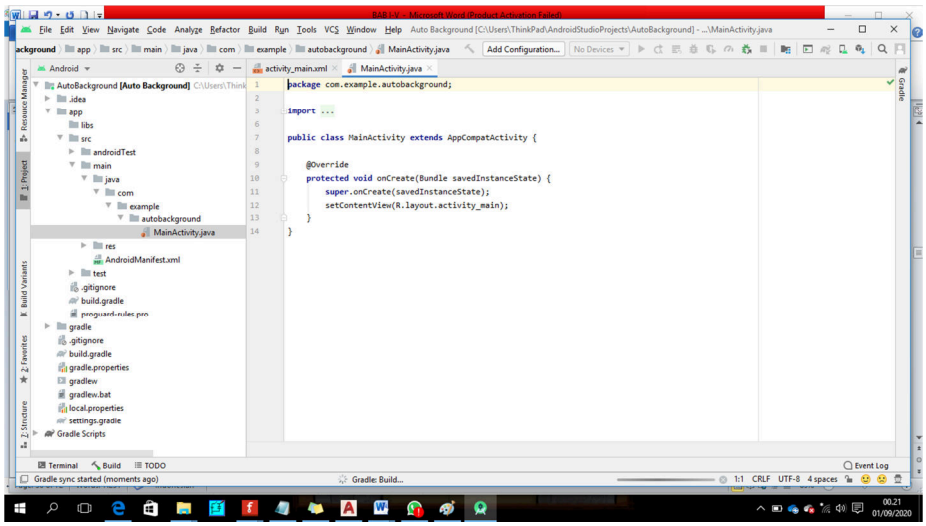


4. Ganti nama *proje*ct menjadi *Auto Background*, pilih *Java* pada kolom *Language*, lalu ganti *minimum* SDK menjadi API 14: *Android* 4.0 (*IceCreamSandwich*) yang merupakan versi *android* paling rendah saat ini.

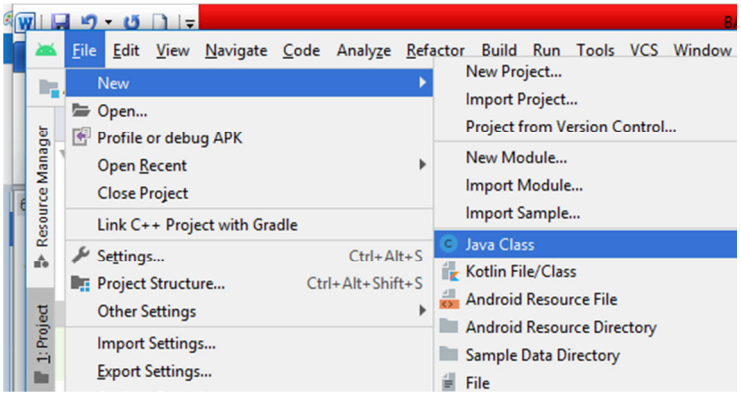
Hal ini bertujuan agar aplikasi ini dapat digunakan pada versi android manapun. Lalu klik "*Finish*".



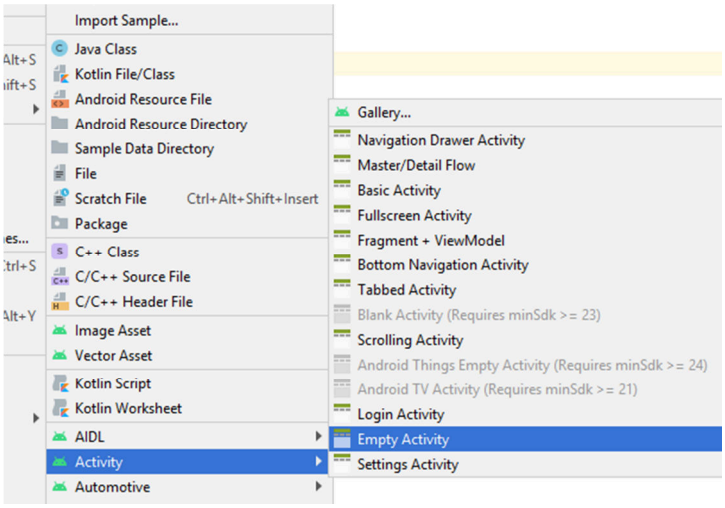
5. Tunggu proses *build gradle* selesai, kemudian ikuti program yang ada pada LAMPIRAN 2.



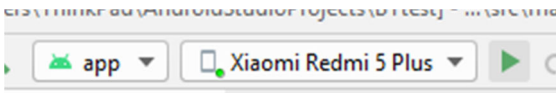
6. Untuk membuat *Java Class* baru, klik *File* > *New* > *Java Class*



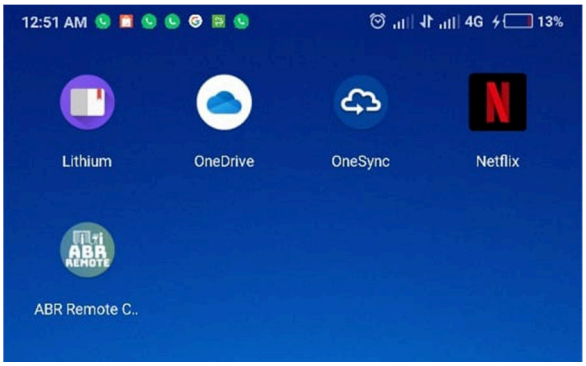
7. Untuk membuat *Activity* baru, klik *File* > *New* > *Activity* > *Empty Activity*.



8. Setelah mengikuti program yang ada di LAMPIRAN 2, sambungkan perangkat *android* menggunakan kabel USB. Namum sebelumnya perangkat *android* harus menerapkan *mode developer* agar bisa digunakan sebagai *simulator* pada *android studio*. Kemudian klik "*Run App*"(simbol kepala panah warna hijau) atau tekan *Shift+F10* pada keyboard. Lalu tunggu sampai proses *gradle build running* selesai.



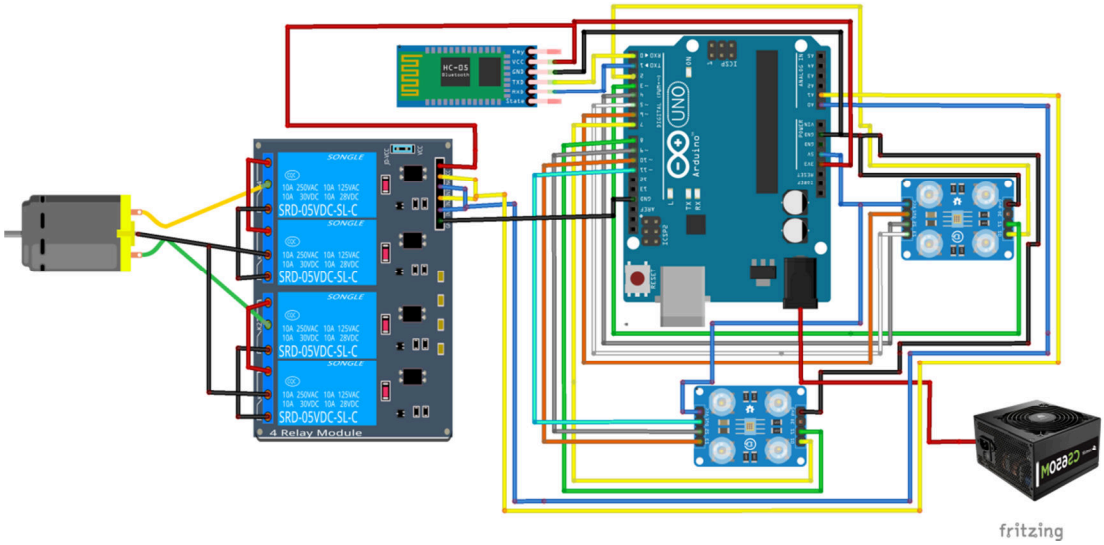
9. Kemudian pada perangkat *android* akan otomatis terpasang aplikasi tersebut dengan nama "ABR *Remote Controller*"



4.5 Pengujian *Automatic Background*

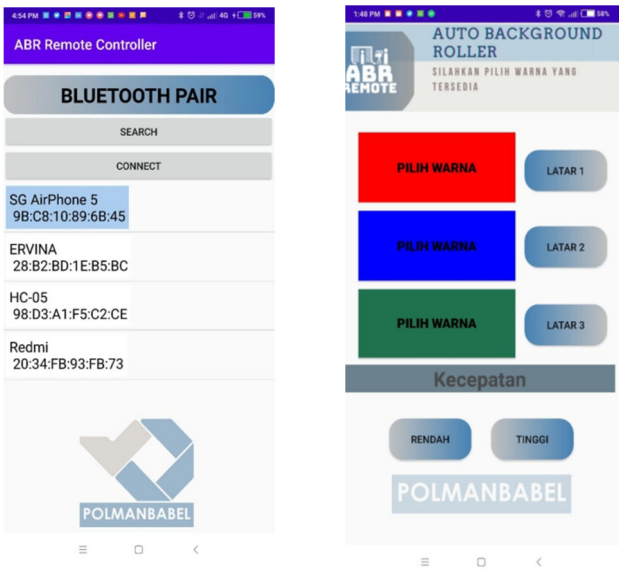
Setelah bagian *hardware* dan *software* dari *Automatic Background* telah dibuat, proses selanjutnya ialah pengujian *Automatic Background* untuk mengetahui apakah alat ini dapat bekerja sesuai dengan yang diinginkan serta untuk melihat bagaimana hasil yang akan didapatkan setelah selesai melakukan proses perancangan, pembuatan dan pengujian pada masing-masing *hardware* maupun *software.*

Berikut adalah gambar skema keseluruhan *hardware* dari *Automatic Background* pada gambar 4.7 dibawah ini.



Gambar 4. 7 Skema *Wiring* keseluruhan *Automatic Background*

Kemudian berikut adalah gambar dari *user interface remote control* dari *Automatic Background* dan hasil konstruksi dari alat ini ialah seperti pada gambar 4.8 dibawah ini.



Gambar 4. 8 *User interface remote control Automatic Background*

Lalu dibawah ini adalah gambar hasil konstruksi mekanis dari *Automatic Background* yang telah dibuat seperti pada gambar 4.9 dibawah ini.

Gambar 4. 9 Hasil Konstruksi Mekanis dari *Automatic Background.*

Kemudian, untuk mengetahui apakah alat ini dapat bekerja lebih efisien dalam kecepatan setiap pergantian warna latar dibandingkan dengan *background* konvensional dan *background semi* otomatis maka dilakukan pengujian kecepatan perpindahan pada tiap warna latar yang hasilnya dapat dilihat dari tabel dibawah:

Tabel 4.5 (a) Hasil Pengujian Kecepatan pada *Background* Konvensional

Warna Latar	**Kecepatan (s)**
Merah	58
Biru	59
Hijau	58

Tabel 4.5 (b) Hasil Pengujian Kecepatan pada *Background Semi* Otomatis

Warna Latar	**Kecepatan (s)**
Merah	34
Biru	33
Hijau	34

Tabel 4.5 (c) Hasil Pengujian Kecepatan *Automatic Background*

Warna Latar	**Kecepatan (s)**					
	Jarak 2m		**Jarak 5m**		**Jarak 7m**	
	Rendah	**Tinggi**	**Rendah**	**Tinggi**	**Rendah**	**Tinggi**
Merah	120	62	120	63	118	125
Biru	40	20	41	19	41	20
Hijau	37	15	37	15	37	16

Berdasarkan hasil data diatas, dapat dilihat bahwa untuk *background* konvensional memakan waktu kurang lebih 1 menit untuk tiap pergantian warna lalu untuk *background* semi otomatis memakan waktu kurang lebih 34 detik dalam satu kali pergantian warna kemudian

untuk *automatic background* warna latar hijau dan biru memakan waktu kurang lebih 20 detik dengan menggunakan kecepatan maksimum sedangkan untuk warna merah memakan waktu lebih lama yakni kurang lebih 63 detik menggunakan kecepatan maksimum.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisa terhadap fungsi alat pada Proyek Akhir dengan judul *"Automatic Background Roller dengan Bluetooth Control System"* ini, maka dapat ditarik kesimpulan bahwa:

1. Alat ini dibuat dengan metode otomatis, dimana pada alat ini menggunakan Arduino sebagai media prosesnya dan Bluetooth sebagai komunikasi datanya.
2. *Automatic Background* dilengkapi dengan modul *relay* untuk mengatur kecepatan putaran motor dan aplikasi *android* sebagai pengendali dari jarak jauh dengan jarak maksimal ± 8 m.
3. Dalam segi kinerja, alat ini dapat bekerja lebih cepat dengan waktu ± 40 detik (kecepatan maksimum) dalam satu kali pergantian warna dibandingkan dengan *background* konvensional dan ± 10 detik (kecepatan maksimum) lebih cepat dalam satu kali pergantian warna dibandingkan dengan *background semi* otomatis .
4. Sensor warna lebih sensitif dengan warna biru, yang mengakibatkan apabila sensor tersebut mendeteksi warna biru maka motor akan langsung berhenti (dalam kondisi *button* apapun).
5. Dalam segi kepraktisan, alat ini lebih praktis sebab dapat dipindah-pindahkan tanpa merusak dinding ruangan dan dapat memuat lebih banyak warna latar dibandingkan dengan *background* semi otomatis.

5.2 Saran

Apabila alat ini akan dikembangkan lebih lanjut, fungsi yang perlu diperbaiki dan dikembangkan adalah memodifikasikan rancangan konstruksi agar alat ini dapat dibuat menjadi lebih ringkas dan pada pengendalinya ditambahkan fitur untuk menambah banyak jumlah tombol dan *display* warna, kemudian

sehubungan dengan sensor TCS-230 yang lebih sensitif dengan satu warna dibandingkan warna lain maka akan lebih baik jika sensor tersebut diganti dengan sensor lain yang dapat bekerja sesuai dengan yang diharapkan.

DAFTAR PUSTAKA

[1] **Linarti, Lusi.** *Aplikasi Bluetooth Pada Pengontrol Alat Elektronik Rumah Tangga Dengan Smartphone Android.* s.l. : Politeknik Negeri Sriwijaya, 2014.

[2] **Daeng, Intan Trivena Maria, kalesaran, edmon r dan Kalesaran, Edmon R.** *Penggunaan Smartphone Dalam Menunjang Aktivitas Perkuliahan.***.** : Acta Diurna Universitas Sam ratulangi, 2017, Vol. 1.

[3] **STIKOM Surabaya.** Repositori Universitas Dinamika. Diakses pada tanggal 9 Juli 2020. http://repository.dinamika.ac.id/id/eprint/2535/4/BAB_II.pdf.

[4] **Kadir, Abdul.** *From Zero to A Pro.* Yogyakarta : ANDI, 2015.

[5] Sensor TCS230. Diakses pada tanggal 16 Agustus 2020. http://eprints.polsri.ac.id/175/3/BAB%20II.pdf.

[6] **Prasetyo, Adi, Hafizah, Putri Nur dan Dkk.***Monitoring Suhu Tubuh Pasien Demam Berdarah Menggunakan Bluetooth yang Diintegrasikan ke Personal Komputer.* 2015, Prosiding SENTIA, hal. 8.

[7] **Setiawan, Rudi dan Bornok, Mardohar Batu.** Research Report - Humanities and Social Science. *Journal Unpar.* 2015. Diakses pada tanggal 09 Juli 2020: http://journal.unpar.ac.id/index.php/Sosial/article/view/1468/1409.

LAMPIRAN

LAMPIRAN 1

DAFTAR RIWAYAT HIDUP

DAFTAR RIWAYAT HIDUP

1. Data Pribadi

Nama lengkap	: Dwi Arta Putra
Tempat & tanggal lahir	: Belinyu, 09 Februari 2000
Alamat rumah	: Jl. Pemuda
Hp	: 0858 3229 6816
Email	: dwiartaputra12@gmail.com
Jenis kelamin	: Laki-Laki
Agama	: Islam

2. Riwayat Pendidikan

SD Negeri 14 Belinyu	2005 - 2006
SD Negeri 09 Belinyu	2006 - 2009
SD Negeri 2 Pemali	2009 - 2011
SMP Negeri 1 Pemali	2011 - 2014
SMK YAPENSU	2014 - 2017
Politeknik Manufaktur Negeri Bangka Belitung	2017 - 2020

3. Pendidikan Non Formal

-

Sungailiat, 19 Agustus 2020

Dwi Arta Putra

DAFTAR RIWAYAT HIDUP

1. Data Pribadi

Nama lengkap	: Ervina Sugianti
Tempat & tanggal lahir	: Sungailiat, 16 November 1999
Alamat rumah	: Jl. Jendral Sudirman Gg Kelud No.15 Paritpadang, Sungailiat
Hp	: 0896 2532 8554
Email	: ervinnasugianti@gmail.com
Jenis kelamin	: Perempuan
Agama	: Islam

2. Riwayat Pendidikan

SD Negeri 8 Sungailiat	2005-2009
SD Negeri 8 Airbara	2009 - 2011
SMP Negeri 3 Airgegas	2011 - 2014
SMA Negeri 1 Sungailiat	2014 - 2017
Politeknik Manufaktur Negeri Bangka Belitung	2017 - 2020

3. Pendidikan Non Formal

-

Sungailiat, 19 Agustus 2020

Ervina Sugianti

LAMPIRAN 2

Program Java Android Studio:

Automatic Background Roller Remote Controller

Class Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="55dp"
        android:text="BLUETOOTH PAIR"
        android:layout_centerInParent="true"
        android:layout_marginTop="16dp"
        android:textColor="@android:color/black"
        android:background="@drawable/custom3"
        android:gravity="center"
        android:textSize="28sp"
        android:textStyle="bold"/>
    <Button
        android:id="@+id/search"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Search" />
    <Button
        android:id="@+id/connect"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Connect" />
    <ListView
        android:id="@+id/listview"
        android:layout_width="wrap_content"
        android:layout_height="326dp" />
    <ImageView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:alpha="0.4"
        android:contentDescription="polmanbabel"
        android:src="@drawable/polman1"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />
</LinearLayout>
```

Class Activity_controlling.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/remote_auto"
```

```
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_centerInParent="true"
tools:context=".Controllin">
<TextView
    android:id="@+id/header"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true"
    android:layout_marginTop="4dp"
    android:background="@color/grey25"
    android:gravity="center"
    android:text="Silahkan Pilih Warna Latar yang Tersedia"
    android:textColor="@android:color/black"
    android:textSize="28sp"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
<Button
    android:id="@+id/bg1"
    android:layout_width="225dp"
    android:layout_height="95dp"
    android:layout_centerInParent="true"
    android:layout_marginStart="20dp"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="76dp"
    android:background="@drawable/custom2"
    android:gravity="center"
    android:text="Pilih Warna"
    android:textColor="@android:color/black"
    android:textSize="18sp"
    android:textStyle="bold"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/header" />
<Button
    android:id="@+id/bg2"
    android:background="@drawable/custom2"
    android:layout_width="225dp"
    android:layout_height="95dp"
    android:layout_centerInParent="true"
    android:layout_marginStart="20dp"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="12dp"
    android:gravity="center"
    android:text="Pilih Warna"
    android:textColor="@android:color/black"
    android:textSize="18sp"
    android:textStyle="bold"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/bg1" />
<Button
    android:id="@+id/bg3"
    android:background="@drawable/custom2"
    android:layout_width="225dp"
```

```
    android:layout_height="95dp"
    android:layout_centerInParent="true"
    android:layout_marginStart="20dp"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="12dp"
    android:gravity="center"
    android:text="Pilih Warna"
    android:textColor="@android:color/black"
    android:textSize="18sp"
    android:textStyle="bold"
    app:layout_constraintBottom_toTopOf="@+id/textView"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/bg2"
    app:layout_constraintVertical_bias="0.0" />
<Button
    android:id="@+id/on"
    android:layout_width="118dp"
    android:layout_height="56dp"
    android:layout_marginTop="100dp"
    android:layout_marginEnd="16dp"
    android:layout_marginRight="16dp"
    android:background="@drawable/custom1"
    android:text="Latar 1"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/header" />
<Button
    android:id="@+id/on1"
    android:layout_width="118dp"
    android:layout_height="56dp"
    android:layout_marginTop="48dp"
    android:layout_marginEnd="16dp"
    android:layout_marginRight="16dp"
    android:background="@drawable/custom1"
    android:text="Latar 2"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/on" />
<Button
    android:id="@+id/on2"
    android:layout_width="118dp"
    android:layout_height="56dp"
    android:layout_marginTop="52dp"
    android:layout_marginEnd="16dp"
    android:layout_marginRight="16dp"
    android:background="@drawable/custom1"
    android:text="Latar 3"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/on1" />
<Button
    android:id="@+id/on3"
    android:layout_width="118dp"
    android:layout_height="56dp"
    android:layout_marginStart="64dp"
    android:layout_marginLeft="64dp"
    android:layout_marginBottom="116dp"
    android:background="@drawable/custom3"
    android:text="Pelan"
```

```
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="parent" />
<Button
    android:id="@+id/on4"
    android:layout_width="118dp"
    android:layout_height="56dp"
    android:layout_marginEnd="64dp"
    android:layout_marginRight="64dp"
    android:layout_marginBottom="116dp"
    android:background="@drawable/custom3"
    android:text="Ngebut"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent" />
<TextView
    android:id="@+id/textView"
    android:backgroundTint="@color/biru"
    android:alpha="0.7"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginEnd="4dp"
    android:layout_marginRight="4dp"
    android:layout_marginBottom="208dp"
    android:background="@color/grey25"
    android:gravity="center"
    android:text="Kecepatan"
    android:textColor="@android:color/black"
    android:textSize="28sp"
    android:textStyle="bold"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent" />
<ImageView
    android:id="@+id/imageView"
    android:layout_width="258dp"
    android:layout_height="131dp"
    android:layout_marginStart="77dp"
    android:layout_marginLeft="77dp"
    android:layout_marginEnd="76dp"
    android:layout_marginRight="76dp"
    android:layout_marginBottom="4dp"
    android:alpha="0.2"
    android:contentDescription="polmanbabel"
    android:src="@drawable/polman"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent" />
<ImageView
    android:id="@+id/imageView1"
    android:layout_width="425dp"
    android:layout_height="152dp"
    android:layout_marginStart="77dp"
    android:layout_marginLeft="77dp"
    android:layout_marginEnd="76dp"
    android:layout_marginRight="76dp"
    android:layout_marginBottom="588dp"
```

```
        android:contentDescription="polman"
        android:src="@drawable/picc"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.461"
        app:layout_constraintStart_toStartOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Class MainActivity.java

```
package com.example.autobackground;
import android.annotation.SuppressLint;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.ActivityInfo;
import android.graphics.Color;
import android.os.AsyncTask;
import android.os.Bundle;
import android.preference.PreferenceManager;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import java.util.ArrayList;
import java.util.List;
import java.util.Set;
import java.util.UUID;
public class MainActivity extends AppCompatActivity {
    private Button search;
    private Button connect;
    private ListView listView;
    private BluetoothAdapter mBTAdapter;
    private static final int BT_ENABLE_REQUEST = 10; // This is
the code we use for BT Enable
    private static final int SETTINGS = 20;
    private UUID mDeviceUUID = UUID.fromString("00001101-0000-
1000-8000-00805F9B34FB");
    private int mBufferSize = 50000; //Default
    public static final String DEVICE_EXTRA =
"com.example.autobackground.SOCKET";
    public static final String DEVICE_UUID =
"com.example.autobackground.uuid";
    private static final String DEVICE_LIST =
"com.example.autobackground.devicelist";
```

```
    private static final String DEVICE_LIST_SELECTED =
"com.example.autobackground.devicelistselected";
    public static final String BUFFER_SIZE =
"com.example.autobackground.buffersize";
    private static final String TAG = "BlueTest5-MainActivity";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        search = (Button) findViewById(R.id.search);
        connect = (Button) findViewById(R.id.connect);
        listView = (ListView) findViewById(R.id.listview);
        if (savedInstanceState != null) {
            ArrayList<BluetoothDevice> list =
savedInstanceState.getParcelableArrayList(DEVICE_LIST);
            if (list != null) {
                initList(list);
                MyAdapter adapter = (MyAdapter)
listView.getAdapter();
                int selectedIndex =
savedInstanceState.getInt(DEVICE_LIST_SELECTED);
                if (selectedIndex != -1) {
                    adapter.setSelectedIndex(selectedIndex);
                    connect.setEnabled(true);
                }
            } else {
                initList(new ArrayList<BluetoothDevice>());
            }
        } else {
            initList(new ArrayList<BluetoothDevice>());
        }
        search.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View arg0) {
                mBTAdapter = BluetoothAdapter.getDefaultAdapter();
                if (mBTAdapter == null) {
                    Toast.makeText(getApplicationContext(),
"Bluetooth not found", Toast.LENGTH_SHORT).show();
                } else if (!mBTAdapter.isEnabled()) {
                    Intent enableBT = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                    startActivityForResult(enableBT,
BT_ENABLE_REQUEST);
                } else {
                    new SearchDevices().execute();
                }
            }
        });
        connect.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View arg0) {
                BluetoothDevice device = ((MyAdapter)
(listView.getAdapter())).getSelectedItem();
                Intent intent = new
Intent(getApplicationContext(), Controllin.class);
                intent.putExtra(DEVICE_EXTRA, device);
```

```
                intent.putExtra(DEVICE_UUID,
mDeviceUUID.toString());
                intent.putExtra(BUFFER_SIZE, mBufferSize);
                startActivity(intent);
            }
        });
    }
    protected void onPause() {
// TODO Auto-generated method stub
        super.onPause();
    }
    @Override
    protected void onStop() {
// TODO Auto-generated method stub
        super.onStop();
    }
    @Override
    protected void onActivityResult(int requestCode, int
resultCode, Intent data) {
        switch (requestCode) {
            case BT_ENABLE_REQUEST:
                if (resultCode == RESULT_OK) {
                    msg("Bluetooth Enabled successfully");
                    new SearchDevices().execute();
                } else {
                    msg("Bluetooth couldn't be enabled");
                }
                break;
            case SETTINGS: //If the settings have been updated
                SharedPreferences prefs =
PreferenceManager.getDefaultSharedPreferences(this);
                String uuid = prefs.getString("prefUuid", "Null");
                mDeviceUUID = UUID.fromString(uuid);
                Log.d(TAG, "UUID: " + uuid);
                String bufSize = prefs.getString("prefTextBuffer",
"Null");
                mBufferSize = Integer.parseInt(bufSize);
                String orientation =
prefs.getString("prefOrientation", "Null");
                Log.d(TAG, "Orientation: " + orientation);
                if (orientation.equals("Landscape")) {
setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE)
;
                } else if (orientation.equals("Portrait")) {
setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
                } else if (orientation.equals("Auto")) {
setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_FULL_SENSO
R);
                }
                break;
            default:
                break;
        }
        super.onActivityResult(requestCode, resultCode, data);
    }
    private void msg(String str) {
```

```
        Toast.makeText(getApplicationContext(), str,
Toast.LENGTH_SHORT).show();
    }
    private void initList(List<BluetoothDevice> objects) {
        final MyAdapter adapter = new
MyAdapter(getApplicationContext(), R.layout.list_item,
R.id.lstContent, objects);
        listView.setAdapter(adapter);
        listView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View
view, int position, long id) {
                adapter.setSelectedIndex(position);
                connect.setEnabled(true);
            }
        });
    }
    @SuppressLint("StaticFieldLeak")
    private class SearchDevices extends AsyncTask<Void, Void,
List<BluetoothDevice>> {
        @Override
        protected List<BluetoothDevice> doInBackground(Void...
params) {
            Set<BluetoothDevice> pairedDevices =
mBTAdapter.getBondedDevices();
            List<BluetoothDevice> listDevices = new
ArrayList<BluetoothDevice>();
            for (BluetoothDevice device : pairedDevices) {
                listDevices.add(device);
            }
            return listDevices;
        }
        @Override
        protected void onPostExecute(List<BluetoothDevice>
listDevices) {
            super.onPostExecute(listDevices);
            if (listDevices.size() > 0) {
                MyAdapter adapter = (MyAdapter)
listView.getAdapter();
                adapter.replaceItems(listDevices);
            } else {
                msg("No paired devices found, please pair your
serial BT device and try again");
            }
        }
    }
    private class MyAdapter extends ArrayAdapter<BluetoothDevice>
{
        private int selectedIndex;
        private Context context;
        private int selectedColor = Color.parseColor("#abcdef");
        private List<BluetoothDevice> myList;
        public MyAdapter(Context ctx, int resource, int
textViewResourceId, List<BluetoothDevice> objects) {
            super(ctx, resource, textViewResourceId, objects);
```

```
            context = ctx;
            myList = objects;
            selectedIndex = -1;
        }
        public void setSelectedIndex(int position) {
            selectedIndex = position;
            notifyDataSetChanged();
        }
        public BluetoothDevice getSelectedItem() {
            return myList.get(selectedIndex);
        }
        @Override
        public int getCount() {
            return myList.size();
        }
        @Override
        public BluetoothDevice getItem(int position) {
            return myList.get(position);
        }
        @Override
        public long getItemId(int position) {
            return position;
        }
        private class ViewHolder {
            TextView tv;
        }
        public void replaceItems(List<BluetoothDevice> list) {
            myList = list;
            notifyDataSetChanged();
        }
        public List<BluetoothDevice> getEntireList() {
            return myList;
        }
        @Override
        public View getView(int position, View convertView,
ViewGroup parent) {
            View vi = convertView;
            ViewHolder holder;
            if (convertView == null) {
                vi =
LayoutInflater.from(context).inflate(R.layout.list_item, null);
                holder = new ViewHolder();
                holder.tv = (TextView)
vi.findViewById(R.id.lstContent);
                vi.setTag(holder);
            } else {
                holder = (ViewHolder) vi.getTag();
            }
            if (selectedIndex != -1 && position == selectedIndex)
{
                holder.tv.setBackgroundColor(selectedColor);
            } else {
                holder.tv.setBackgroundColor(Color.WHITE);
            }
            BluetoothDevice device = myList.get(position);
            holder.tv.setText(device.getName() + "\n " +
```

```
device.getAddress());
            return vi;
        }
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is
present.
        //getMenuInflater().inflate(R.menu.homescreen, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_settings:
                Intent intent = new Intent(MainActivity.this,
PreferencesActivity.class);
                startActivityForResult(intent, SETTINGS);
                break;
        }
        return super.onOptionsItemSelected(item);
    }
}
```

Class Controlling.java

```
package com.example.autobackground;
import android.app.Activity;
import android.app.ProgressDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
import java.io.IOException;
import java.io.InputStream;
import java.util.ArrayList;
import java.util.UUID;
import petrov.kristiyan.colorpicker.ColorPicker;
public class Controllin extends Activity {
    private static final String TAG = "BlueTest5-Controlling";
    private int mMaxChars = 50000;//Default//change this to
string.....
    private UUID mDeviceUUID;
    private BluetoothSocket mBTSocket;
    private ReadInput mReadThread = null;
    private boolean mIsUserInitiatedDisconnect = false;
    private boolean mIsBluetoothConnected = false;
    private Button mBtnDisconnect;
    private BluetoothDevice mDevice;
```

```
    final static String on="11";//latar1
    final static String on1="12";//latar2
    final static String on2="13";//latar3
    final static String on3="14";//pelan
    final static String on4="15";//ngebut
    private ProgressDialog progressDialog;
    Button btnOn,btnOn1,btnOn2,btnOn3,btnOn4, bg1, bg2, bg3;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_controllin);
        ActivityHelper.initialize(this);
        // mBtnDisconnect = (Button)
findViewById(R.id.btnDisconnect);
        btnOn=(Button)findViewById(R.id.on);
        btnOn1=(Button)findViewById(R.id.on1);
        btnOn2=(Button)findViewById(R.id.on2);
        btnOn3=(Button)findViewById(R.id.on3);
        btnOn4=(Button)findViewById(R.id.on4);
        bg1 = findViewById(R.id.bg1);
        bg2 = findViewById(R.id.bg2);
        bg3 = findViewById(R.id.bg3);
        final Button bg1=(Button) findViewById(R.id.bg1);
        final Button bg2=(Button) findViewById(R.id.bg2);
        final Button bg3=(Button) findViewById(R.id.bg3);
        Intent intent = getIntent();
        Bundle b = intent.getExtras();
        mDevice = b.getParcelable(MainActivity.DEVICE_EXTRA);
        mDeviceUUID =
UUID.fromString(b.getString(MainActivity.DEVICE_UUID));
        mMaxChars = b.getInt(MainActivity.BUFFER_SIZE);
        Log.d(TAG, "Ready");
        bg1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                opencolorpicker1();
            }
        });
        bg2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                opencolorpicker2();
            }
        });
        bg3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                opencolorpicker3();
            }
        });
        btnOn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
// TODO Auto-generated method stub
                try {
```

```
mBTSocket.getOutputStream().write(on.getBytes());
                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }});
        btnOn1.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v) {
// TODO Auto-generated method stub
                try {

mBTSocket.getOutputStream().write(on1.getBytes());
                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }});
        btnOn2.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v) {
// TODO Auto-generated method stub
                try {

mBTSocket.getOutputStream().write(on2.getBytes());
                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }});
        btnOn3.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v) {
// TODO Auto-generated method stub
                try {

mBTSocket.getOutputStream().write(on3.getBytes());

                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }});
        btnOn4.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v) {
// TODO Auto-generated method stub
                try {

mBTSocket.getOutputStream().write(on4.getBytes());
                } catch (IOException e) {
                    // TODO Auto-generated catch block
```

```
                    e.printStackTrace();
                }
            }});
    }
    public void opencolorpicker1() {
        final ColorPicker colorPicker = new ColorPicker(this);
        ArrayList<String> colors = new ArrayList<>();
        colors.add("#FFFF0000");
        colors.add("#FF0000FF");
        colors.add("#20724f");
        colors.add("#6a3ab2");
        colorPicker.setColors(colors)
                .setColumns(5)
                .setRoundColorButton(true)
                .setOnChooseColorListener(new
ColorPicker.OnChooseColorListener() {
                    @Override
                    public void onChooseColor(int position, int
color) {
                        bg1.setBackgroundColor(color);
                    }
                    @Override
                    public void onCancel() {
                    }
                }).show();
    }
    public void opencolorpicker2() {
        final ColorPicker colorPicker = new ColorPicker(this);
        ArrayList<String> colors = new ArrayList<>();
        colors.add("#FFFF0000");
        colors.add("#FF0000FF");
        colors.add("#20724f");
        colors.add("#6a3ab2");
        colorPicker.setColors(colors)
                .setColumns(5)
                .setRoundColorButton(true)
                .setOnChooseColorListener(new
ColorPicker.OnChooseColorListener() {
                    @Override
                    public void onChooseColor(int position, int
color) {
                        bg2.setBackgroundColor(color);
                    }
                    @Override
                    public void onCancel() {
                    }
                }).show();
    }
    public void opencolorpicker3() {
        final ColorPicker colorPicker = new ColorPicker(this);
        ArrayList<String> colors = new ArrayList<>();
        colors.add("#FFFF0000");
        colors.add("#FF0000FF");
        colors.add("#20724f");
        colors.add("#6a3ab2");
        colorPicker.setColors(colors)
```

```
                .setColumns(5)
                .setRoundColorButton(true)
                .setOnChooseColorListener(new
ColorPicker.OnChooseColorListener() {
                    @Override
                    public void onChooseColor(int position, int
color) {
                        bg3.setBackgroundColor(color);
                    }
                    @Override
                    public void onCancel() {
                    }
                }).show();
    }
    private class ReadInput implements Runnable {
        private boolean bStop = false;
        private Thread t;
        public ReadInput() {
            t = new Thread(this, "Input Thread");
            t.start();
        }
        public boolean isRunning() {
            return t.isAlive();
        }
        @Override
        public void run() {
            InputStream inputStream;
            try {
                inputStream = mBTSocket.getInputStream();
                while (!bStop) {
                    byte[] buffer = new byte[256];
                    if (inputStream.available() > 0) {
                        inputStream.read(buffer);
                        int i = 0;
                        for (i = 0; i < buffer.length && buffer[i]
!= 0; i++) {
                        }
                        final String strInput = new String(buffer,
0, i);

                    }
                    Thread.sleep(500);
                }
            } catch (IOException e) {
// TODO Auto-generated catch block
                e.printStackTrace();
            } catch (InterruptedException e) {
// TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
        public void stop() {
            bStop = true;
        }
    }
    private class DisConnectBT extends AsyncTask<Void, Void, Void>
```

```
{
        @Override
        protected void onPreExecute() {
        }
        @Override
        protected Void doInBackground(Void... params) {//cant
inderstand these dotss
            if (mReadThread != null) {
                mReadThread.stop();
                while (mReadThread.isRunning())
                    ; // Wait until it stops
                mReadThread = null;
            }
            try {
                mBTSocket.close();
            } catch (IOException e) {
// TODO Auto-generated catch block
                e.printStackTrace();
            }
            return null;
        }
        @Override
        protected void onPostExecute(Void result) {
            super.onPostExecute(result);
            mIsBluetoothConnected = false;
            if (mIsUserInitiatedDisconnect) {
                finish();
            }
        }
    }
    private void msg(String s) {
        Toast.makeText(getApplicationContext(), s,
Toast.LENGTH_SHORT).show();
    }
    @Override
    protected void onPause() {
        if (mBTSocket != null && mIsBluetoothConnected) {
            new DisConnectBT().execute();
        }
        Log.d(TAG, "Paused");
        super.onPause();
    }
    @Override
    protected void onResume() {
        if (mBTSocket == null || !mIsBluetoothConnected) {
            new ConnectBT().execute();
        }
        Log.d(TAG, "Resumed");
        super.onResume();
    }
    @Override
    protected void onStop() {
        Log.d(TAG, "Stopped");
        super.onStop();
    }
    @Override
```

```
    protected void onSaveInstanceState(Bundle outState) {
// TODO Auto-generated method stub
        super.onSaveInstanceState(outState);
    }
    private class ConnectBT extends AsyncTask<Void, Void, Void> {
        private boolean mConnectSuccessful = true;
        @Override
        protected void onPreExecute() {
            progressDialog = ProgressDialog.show(Controllin.this,
"Hold on", "Connecting");//
http://stackoverflow.com/a/11130220/1287554
        }
        @Override
        protected Void doInBackground(Void... devices) {

            try {
                if (mBTSocket == null || !mIsBluetoothConnected) {
                    mBTSocket =
mDevice.createInsecureRfcommSocketToServiceRecord(mDeviceUUID);
BluetoothAdapter.getDefaultAdapter().cancelDiscovery();
                    mBTSocket.connect();
                }
            } catch (IOException e) {
// Unable to connect to device`
                // e.printStackTrace();
                mConnectSuccessful = false;
            }
            return null;
        }
        @Override
        protected void onPostExecute(Void result) {
            super.onPostExecute(result);
            if (!mConnectSuccessful) {
                Toast.makeText(getApplicationContext(), "Could not
connect to device.Please turn on your Hardware",
Toast.LENGTH_LONG).show();
                finish();
            } else {
                msg("Connected to device");
                mIsBluetoothConnected = true;
                mReadThread = new ReadInput(); // Kick off input
reader
            }
            progressDialog.dismiss();
        }
    }

    @Override
    protected void onDestroy() {
        // TODO Auto-generated method stub
        super.onDestroy();
    }
}
```

LAMPIRAN 3

Program Arduino UNO: *Automatic Background Roller dengan Bluetooth Control System*

```
#include <SoftwareSerial.h>
SoftwareSerial Blue(0, 1);
#define RED 0
#define GREEN 1
#define BLUE 2
#define red 0
#define green 1
#define blue 2
int c = 0;
int x = 0;
int a = 0;
int W = 0;
int d = 0;
long data ;
long int pb1 = 11; //latar 1 merah
long int pb2 = 12; //latar 2 biru
long int pb3 = 13; //latar 3 hijau
long int pb4 = 14; //pelan
long int pb5 = 15; //ngebut
char state = 0;
const char s0 = 2;
const char s1 = 3;
const char s2 = 4;
const char s3 = 5;
const char out = 6;
const char Sw0 = 7;
const char Sw1 = 8;
```

```
const char Sw2 = 9;
const char Sw3 = 10;
const char sensorOut = 11;
int rouge;
int rgbValues[3] = {};
int colorValues[3] = {};
void readColorSensor();
void colorsensor();
void setup() {
  Serial.begin(9600);
  Blue.begin(9600);
  pinMode(A0, OUTPUT);
  pinMode(A1, OUTPUT);
  pinMode(A2, OUTPUT);
  pinMode(A3, OUTPUT);
  pinMode(s0, OUTPUT);
  pinMode(s1, OUTPUT);
  pinMode(s2, OUTPUT);
  pinMode(s3, OUTPUT);
  pinMode(out, INPUT);
  pinMode(Sw0, OUTPUT);
  pinMode(Sw1, OUTPUT);
  pinMode(Sw2, OUTPUT);
  pinMode(Sw3, OUTPUT);
  pinMode(sensorOut, INPUT);
  digitalWrite(s0, HIGH);
  digitalWrite(s1, HIGH);
```

```
  digitalWrite(Sw0, HIGH);
  digitalWrite(Sw1, HIGH);
  digitalWrite(A2, HIGH);
  digitalWrite(A3, HIGH);
  digitalWrite(A0, HIGH);
  digitalWrite(A1, HIGH);
}
void loop()
{
  colorsensor();
  readColorSensor();
  while (Blue.available() == 0);
  if (Blue.available() > 0)
  {
    data = Blue.parseInt();
  }
  if (data == pb5)
  {
    x = 1;
    Serial.println("c");
  }
  while (x == 1)
  {
    Serial.println("dlm while");
    if (data == pb4)
    {
      a = 1;
```

```
    x = 0;
    digitalWrite(A2, HIGH);
    digitalWrite(A3, HIGH);
    digitalWrite(A0, HIGH);
    digitalWrite(A1, HIGH);
    Serial.println("mati");
  }
  cepat();
}
if (data == pb4)
{
  a = 1;
  Serial.println("a");
}
while (a == 1)
{
  Serial.println("dlm while pelan");
  if (data == pb5)
  {
    x = 1;
    a = 0;
    digitalWrite(A2, HIGH);
    digitalWrite(A3, HIGH);
    digitalWrite(A0, HIGH);
    digitalWrite(A1, HIGH);
    Serial.println("mati");
  }
```

```
    rendah();
  }
}
void cepat()
{
  switch (rouge)
case 1: opsi:
colorsensor();
  readColorSensor();
  while (Blue.available() == 0);
  if (Blue.available() > 0)
  {
    data = Blue.parseInt();
  }
  if (data == pb1)
  {
    c = 1;
    digitalWrite(A2, LOW);
    digitalWrite(A3, HIGH);
    digitalWrite(A0, HIGH);
    digitalWrite(A1, HIGH);
    Serial.println("latar 1"); //bg merah
    Serial.println(colorValues[red]);
    while (c == 1) {
      colorsensor();
      if (colorValues[red] < colorValues[blue] &&
colorValues[red] < colorValues[green])
      {
```

```
        data = Blue.parseInt();
        colorsensor();
        digitalWrite(A2, HIGH);
        digitalWrite(A3, HIGH);
        digitalWrite(A0, HIGH);
        digitalWrite(A1, HIGH);
        goto opsi;
      }
    }
  }
  else if (data == pb3)
  {
    d = 1;
    digitalWrite(A2, HIGH);
    digitalWrite(A3, LOW);
    digitalWrite(A0, HIGH);
    digitalWrite(A1, HIGH);
    Serial.println("latar 3"); //bg merah
    Serial.println(rgbValues[GREEN]);
    while (d == 1) {
      readColorSensor();
      if (rgbValues[GREEN] < rgbValues[BLUE] &&
rgbValues[GREEN] < rgbValues[RED])
      {

        data = Blue.parseInt();
        readColorSensor();
        digitalWrite(A2, HIGH);
```

```
        digitalWrite(A3, HIGH);
        digitalWrite(A0, HIGH);
        digitalWrite(A1, HIGH);
        goto opsi;
      }
    }
  }
    else if (data == pb2)
  {
    W = 1;
    digitalWrite(A2, HIGH);
    digitalWrite(A3, LOW);
    digitalWrite(A0, HIGH);
    digitalWrite(A1, HIGH);
    Serial.println("latar 2"); //bg merah
    Serial.println(rgbValues[BLUE] );
    while (W == 1) {
      readColorSensor();
      colorsensor();
      if ((colorValues[blue] && rgbValues[BLUE])>
(rgbValues[RED] && colorValues[green]))
      {
        data = Blue.parseInt();
        readColorSensor();
        colorsensor();
        digitalWrite(A2, HIGH);
        digitalWrite(A3, HIGH);
        digitalWrite(A0, HIGH);
```

```
        digitalWrite(A1, HIGH);
        goto opsi;
      }
    }
  }
}
////////////////////////////////////////////////////
//////////////
void rendah()
{
    switch (rouge)
case 1: opsi:
colorsensor();
  readColorSensor();
  while (Blue.available() == 0);
  if (Blue.available() > 0)
  {
    data = Blue.parseInt();
  }
  if (data == pb1)
  {
    c = 1;
    digitalWrite(A2, HIGH);
    digitalWrite(A3, HIGH);
    digitalWrite(A0, LOW);
    digitalWrite(A1, HIGH);
    Serial.println("latar 1"); //bg merah
    Serial.println(rgbValues[RED]);
```

```
    while (c == 1) {
      colorsensor();
      if (colorValues[red] < colorValues[BLUE] &&
rgbValues[RED] < rgbValues[GREEN])
      {
        data = Blue.parseInt();
        readColorSensor();
        digitalWrite(A2, HIGH);
        digitalWrite(A3, HIGH);
        digitalWrite(A0, HIGH);
        digitalWrite(A1, HIGH);
        goto opsi;
      }
    }
  }
  else if (data == pb3)
  {
    d = 1;
    digitalWrite(A2, HIGH);
    digitalWrite(A3, HIGH);
    digitalWrite(A0, HIGH);
    digitalWrite(A1, LOW);
    Serial.println("latar 3"); //bg merah
    Serial.println(rgbValues[GREEN]);
    while (d == 1) {
      readColorSensor();
      if (rgbValues[GREEN] < rgbValues[BLUE] &&
rgbValues[GREEN] < rgbValues[RED])
```

```
      {
        data = Blue.parseInt();
        readColorSensor();
        digitalWrite(A2, HIGH);
        digitalWrite(A3, HIGH);
        digitalWrite(A0, HIGH);
        digitalWrite(A1, HIGH);
        goto opsi;
      }
    }
  }
    else if (data == pb2)
  {
    W = 1;
    digitalWrite(A2, HIGH);
    digitalWrite(A3, HIGH);
    digitalWrite(A0, HIGH);
    digitalWrite(A1, LOW);
    Serial.println("latar 2"); //bg merah
    Serial.println(rgbValues[BLUE]);
    Serial.println(colorValues[blue]);
    while (W == 1) {
      readColorSensor();
      colorsensor();
      if ((colorValues[blue] && rgbValues[BLUE])>
(rgbValues[RED] && colorValues[green]))
      {
        data = Blue.parseInt();
```

```
        readColorSensor();
        colorsensor();
        digitalWrite(A2, HIGH);
        digitalWrite(A3, HIGH);
        digitalWrite(A0, HIGH);
        digitalWrite(A1, HIGH);
        goto opsi;
      }
    }
  }

}
//////////////////////////////////////////////////////
/////////////
void readColorSensor()
{
  digitalWrite(s2, LOW);
  digitalWrite(s3, LOW);
  rgbValues[RED] = pulseIn(out, HIGH);
  digitalWrite(s2, HIGH);
  digitalWrite(s3, HIGH);
  rgbValues[GREEN] = pulseIn(out, HIGH);
  digitalWrite(s2, LOW);
  digitalWrite(s3, HIGH);
  rgbValues[BLUE] = pulseIn(out, HIGH);
}
//////////////////////////////////////////////////////
////////
```

```
void colorsensor()
{
  digitalWrite(Sw2, LOW);
  digitalWrite(Sw3, LOW);
  colorValues[red] = pulseIn(sensorOut, HIGH);
  digitalWrite(Sw2, HIGH);
  digitalWrite(Sw3, HIGH);
  colorValues[green] = pulseIn(sensorOut, HIGH);
  digitalWrite(Sw2, LOW);
  digitalWrite(Sw3, HIGH);
  colorValues[blue] = pulseIn(sensorOut, HIGH);
}
```