

LAMPIRAN 1 :

List Program alat

```
//===== Libraries ======
```

```
#include <Ultrasonic.h>
#include <Servo.h>
#include <LiquidCrystal.h>
#include <SoftwareSerial.h>
#include <EEPROM.h>
```

#define SLOT_WARNA	30
#define MAX_DISTANCE	20
#define TIME_SCAN_ULTRA	2000
#define COUNTER_SCAN_ULTRA	5
#define SCAN_IR	500
#define NEXT_SCAN_IR	3000
#define OPEN_DURATION	5000
#define PASSWORD	2264

```
//===== Static Parameter =====
```

#define S0_COLOR	14
#define S1_COLOR	13
#define S2_COLOR	12
#define S3_COLOR	11
#define COLOR_OUT	10
#define LED_ORG	17
#define LED_NON_ORG	16

#define SERVO_ORG	3
#define SERVO_NON_ORG	4
#define TRIG_ORG	6
#define ECHO_ORG	7
#define TRIG_NON_ORG	8
#define ECHO_NON_ORG	9
#define IR	5
#define PROX	2
#define RS	18
#define EN	19
#define DB4	20
#define DB5	21
#define DB6	22
#define DB7	23
#define SIM_TX	0
#define SIM_RX	1

//===== Objects ======

```
Ultrasonic      ultra_org(TRIG_ORG, ECHO_ORG); //trig,  
echo  
  
Ultrasonic      ultra_non_org(TRIG_NON_ORG,  
ECHO_NON_ORG); //trig, echo  
  
Servo          servo_org;  
  
Servo          servo_non_org;  
  
LiquidCrystal    lcd(RS, EN, DB4, DB5, DB6, DB7);  
  
SoftwareSerial  SIM800L(SIM_TX, SIM_RX); // RX | TX
```

```

//===== Variable =====

String lokasiSampah = "Universitas 17 Agustus 1945";
uint8_t numberKeypad[4][4] = {
    {1, 2, 3, 12},
    {4, 5, 6, 13},
    {7, 8, 9, 14},
    {10, 0, 11, 15},
};

uint8_t keypadRowPin[4] = {24, 25, 26, 27};
uint8_t keypadColPin[4] = {28, 29, 30, 31};
uint32_t currMillis = 0;
uint32_t prevMillis [5];
uint16_t distanceOrg = 0, distanceNonOrg
= 0,
        counterOrgPenuh = 0, counterNonOrgPenuh
= 0;
uint8_t ERROR_TOLERANCE = 4;
uint8_t sColorPin[4] = {S0_COLOR, S1_COLOR,
S2_COLOR, S3_COLOR};
uint8_t senseColor[3][2] = {
    {0, 0},
    {1, 1},
    {0, 1}
};

```

```

        uint8_t red[SLOT_WARNA];
        uint8_t green[SLOT_WARNA];
        uint8_t blue[SLOT_WARNA];

        bool stateOrg      = false, stateNonOrg      = false,
             stateOrgFull   = false, stateNonOrgFull   = false,
             stateNotifOrg   = false, stateNotifNonOrg   = false,
             stateScanIr     = false;

//===== Main program ======
void setup() {
    //Initialize LCD as a main display.
    lcd.begin(16, 2);
    printHomePage();

    //Initialize millis.
    for(uint8_t i=0;i<5;i++) prevMillis[i] = 0;

    //Initialize Keypad Pins.
    for(uint8_t i=0;i<4;i++){
        pinMode(keypadRowPin[i], OUTPUT);
        digitalWrite(keypadRowPin[i], HIGH);
        pinMode(keypadColPin[i], INPUT_PULLUP);
    }
}

```

```
//Initialize Servo motors.  
servo_org.attach(3);  
servo_non_org.attach(4);  
servo_org.write(90);  
servo_non_org.write(90);  
  
//Restore data from ROM to RAM.  
restoreData();  
  
//Initialize color sensor I/O.  
pinMode(COLOR_OUT, INPUT);  
for(uint8_t i=0;i<4;i++) pinMode(sColorPin[i], OUTPUT);  
digitalWrite(sColorPin[0], HIGH);  
digitalWrite(sColorPin[1], LOW);  
  
//Initialize IR, PROXIMITY, and LED pin.  
pinMode(IR, INPUT);  
pinMode(PROX, INPUT);  
pinMode(LED_ORG, OUTPUT);  
pinMode(LED_NON_ORG, OUTPUT);  
digitalWrite(LED_ORG, HIGH);  
digitalWrite(LED_NON_ORG, HIGH);  
  
//Initialize GSM Module.  
SIM800L.begin(9600);  
SIM800L.println("AT+CMGF=1");
```

```
SIM800L.println("AT+CNMI=2,2,0,0,0");
}

void loop() {
    //Get timer value.
    currMillis = millis();

    //Reading keypad.
    if(getKeypadData() == 10){ // Bintang
        uint8_t index = 16, number = 0, stateSetting = 0, totalPass = 0,
        totalPassPast = 0;
        uint32_t pinNumber = 0;
        String pass = "";

        lcd.clear();
        lcd.setCursor(0,0); lcd.print("Insert PIN:");
        while(stateSetting == 0){
            index = getKeypadData();

            if(index < 10){
                if(totalPass < 4){
                    pinNumber = (pinNumber * 10) +
index;
                    totalPass++;
                }
            }
        }
    }
}
```

```
else if(index == 14){  
    if(totalPass > 0){  
        pinNumber = pinNumber / 10;  
        totalPass--;  
    }  
}  
  
else if(index == 11){  
    if(pinNumber == PASSWORD) stateSetting = 1;  
    index = 16;  
}  
  
else if(index == 13){  
    goto skip;  
}  
  
if(totalPass != totalPassPast){  
    totalPassPast = totalPass;  
    pass = "";  
    for(uint8_t i=0;i<totalPass;i++) pass = pass + "*";  
    lcd.setCursor(0,1); lcd.print(" ");  
    lcd.setCursor(0,1); lcd.print(pass);  
}  
delay(250);  
}  
  
lcd.clear();
```

```

lcd.setCursor(0,0); lcd.print("Index Warna:");

while(index != 11){ // Pagar

    index = getKeypadData();

    if(index < 10){ // Nomor

        if(number < 10) number = (number * 10) + index;

        lcd.setCursor(0,1); lcd.print(number);

    }else if(index == 15){ // Clear EEPROM

        clearEEPROM();

        goto skip;

    }else if(index == 14){ // Backspace

        if(number > 0) number = number / 10;

        lcd.setCursor(0,1); lcd.print(" ");

        lcd.setCursor(0,1); lcd.print(number);

    }else if(index == 13){ // Back to Main Menu

        goto skip;

    }

    else if(index == 12){

        if(number <= 10) {

            ERROR_TOLERANCE = number;

            EEPROM.write(100, number);

            lcd.clear();

            lcd.home();

            lcd.print("Tolerance set :");

            lcd.setCursor(0,1); lcd.print(String(number));

            delay(500);

            goto skip;

        }
    }
}

```

```
        }

    }

delay(200);

}

if(index == 11){

if(number <= 30){

    scanColor(number);

    lcd.clear();

    lcd.setCursor(0,0); lcd.print("Release item!");

    delay(NEXT_SCAN_IR);

}else{

    lcd.clear();

    lcd.setCursor(0,0); lcd.print("Maximum 30 Slots");

}

skip:

printHomePage();

//Reset timer for Ultrasonic and Next Scan IR.

prevMillis[0] = currMillis;

prevMillis[1] = currMillis;

prevMillis[2] = currMillis;

}
```

```

//Check ultrasonic sensor every TIME_SCAN_ULTRA.

if(currMillis - prevMillis[0] >= TIME_SCAN_ULTRA){

    distanceOrg      = ultra_org.read();
    distanceNonOrg = ultra_non_org.read();

    (distanceOrg < MAX_DISTANCE) ?
        counterOrgPenuh++ : counterOrgPenuh = 0;
    (distanceNonOrg < MAX_DISTANCE) ?
        counterNonOrgPenuh++ : counterNonOrgPenuh = 0;
    if(counterOrgPenuh == COUNTER_SCAN_ULTRA)
        if(stateOrgFull == false)
            stateOrgFull = true;
    if(counterNonOrgPenuh == COUNTER_SCAN_ULTRA)
        if(stateNonOrgFull == false)
            stateNonOrgFull = true;
    prevMillis[0] = currMillis;
}

```

```

//Check IR Sensor.

if(stateScanIr == false){

    if(currMillis - prevMillis[1] >= SCAN_IR){

        if(digitalRead(IR) == LOW){ //Metal
            stateScanIr = true;
            delay(2000);
            if(digitalRead(PROX) == HIGH){
                checkNonOrgGarbage();
            }else{

```

```

        uint8_t checkCounter = 0;

        //((checkColor() == true) ?
checkOrgGarbage() : checkNonOrgGarbage();

                for(uint8_t i=0;i<20;i++)
if(checkColor() == true) checkCounter++;

                (checkCounter >= 17) ?
checkOrgGarbage() : checkNonOrgGarbage();

        }

        prevMillis[2] = currMillis;

    }

        prevMillis[1] = currMillis;

    }

}

//Reset interval time of IR to scan the next period.

if(currMillis - prevMillis[2] >= NEXT_SCAN_IR){

        //prevMillis[1] = currMillis;

        prevMillis[2] = currMillis;

        stateScanIr = false;

}

//The garbage will open with their own duration.

if(stateNonOrg){

        if(currMillis - prevMillis[3] >= OPEN_DURATION){

                closeTrashCan(2);

                //servo_non_org.write(90);

                stateNonOrg = false;

}

```

```

        printHomePage();

    }

}

//The garbage will open with their own duration.

if(stateOrg){

if(currMillis - prevMillis[4] >= OPEN_DURATION){

    closeTrashCan(1);

    //servo_org.write(90);

    stateOrg = false;

    printHomePage();

}

}

//Check garbage condition, it's full or not.

checkGarbageCondition();

}

//===== Function =====

void closeTrashCan(uint8_t index){

    for (int i = 0; i < 90; i++) {

        (index == 1) ? servo_org.write(i) :

servo_non_org.write(i);

        delay(10);

    }

}

void openTrashCan(uint8_t index){

    for (int i = 90; i > 0; i--) {

```

```

        (index == 1) ? servo_org.write(i) :
servo_non_org.write(i);

        delay(10);

    }

}

uint8_t getKeypadData(){

    uint8_t result = 16;

    for(uint8_t i=0;i<4;i++){

        digitalWrite(keypadRowPin[i], LOW);

        for(uint8_t j=0;j<4;j++){

            if(digitalRead(keypadRowPin[i]) ==
digitalRead(keypadColPin[j])){

                result = numberKeypad[i][j];

                digitalWrite(keypadRowPin[i],
HIGH);

                return result;
            }
        }

        digitalWrite(keypadRowPin[i], HIGH);
    }

    return result;
}

void clearEEPROM(){

    lcd.clear();
}

```

```

lcd.setCursor(0,0); lcd.print("Erasing EEPROM..");

for (uint16_t i=0;i<EEPROM.length();i++) EEPROM.write(i,
0);

delay(500);

lcd.clear();

lcd.setCursor(0,0); lcd.print("Task Complete");

delay(500);

}

void scanColor(uint8_t preset){

if(preset >= 0 && preset < SLOT_WARNA){

    uint16_t counter = 0, r = 0, g = 0, b = 0;

    lcd.clear();

    lcd.setCursor(0,0); lcd.print("Scanning...");

    while(counter < 10){

        counter++;

        r += readColor(0); delay(100);

        g += readColor(1); delay(100);

        b += readColor(2); delay(100);

    }

    red[preset]      = r / counter;

    EEPROM.write(preset, red[preset]);

    green[preset]    = g / counter;
}

```

```

EEPROM.write(preset + SLOT_WARNA,
green[preset]);

blue[preset]      = b / counter;
EEPROM.write(preset + (SLOT_WARNA * 2),
blue[preset]);

lcd.clear();
lcd.home();
lcd.print("Nilai Warna :");
lcd.setCursor(0,1);
lcd.print("R:" + String(red[preset]) + "G:" +
String(green[preset]) + "B:" + String(blue[preset]));
delay(2000);
}

}

bool checkColor(){
    uint32_t      colorResult[3];
    uint16_t      sample = 100, counter = 0,
stateColor[SLOT_WARNA];

for(uint8_t i=0;i<3;i++) colorResult[i] = 0;

//Initialize value of state var.
for(uint8_t i=0;i<SLOT_WARNA;i++) stateColor[i] = 1;

```

```

//Find average value of sensor reading.

for(uint8_t i=0;i<3;i++){
    for(uint8_t j=0;j<sample;j++){
        colorResult[i] += readColor(i);
    }
    colorResult[i] = colorResult[i] / sample;
}

lcd.clear();
lcd.home();
lcd.print("Nilai Warna :");
lcd.setCursor(0,1);

lcd.print("R:" + String(colorResult[0]) + "G:" + String(colorResult[1])
+ "B:" + String(colorResult[2]));

delay(10);

//Match the color with local database.

for(uint8_t i=0;i<SLOT_WARNA;i++){

    (colorResult[0] >= red[i] - ERROR_TOLERANCE      &&
    colorResult[0] < red[i] + ERROR_TOLERANCE) ?

    stateColor[i] *= 1 : stateColor[i] *= 0;

    (colorResult[1] >= green[i] - ERROR_TOLERANCE &&
    colorResult[1] < green[i] + ERROR_TOLERANCE) ?

    stateColor[i] *= 1 : stateColor[i] *= 0;

    (colorResult[2] >= blue[i] - ERROR_TOLERANCE && colorResult[2]
    < blue[i] + ERROR_TOLERANCE) ?

    stateColor[i] *= 1 : stateColor[i] *= 0;
}

```

```

    }

//Check if it match or not.

for(uint8_t i=1;i<SLOT_WARNA;i++){

    if(stateColor[i] == 1) return true;

}

return false;

}

uint16_t readColor(uint8_t color){

    uint16_t frequency = 0;

    digitalWrite(sColorPin[2], senseColor[color][0]);

    digitalWrite(sColorPin[3], senseColor[color][1]);

    frequency = pulseIn(COLOR_OUT, LOW);

    if(frequency <= 255){

        return frequency;

    }else{

        return 255;

    }

}

void checkOrgGarbage(){

    if(stateOrgFull == false){

        //openTrashCan(1);

        servo_org.write(0);

        stateOrg = true;

    }

}

```

```

        printJenisSampah(0);
        prevMillis[4] = currMillis;
    }else{
        printSampahPenuh(0);
        delay(2000);
        printHomePage();
    }
}

void checkNonOrgGarbage(){
    if(stateNonOrgFull == false){
        //openTrashCan(2);
        servo_non_org.write(0);
        stateNonOrg = true;
        printJenisSampah(1);
        prevMillis[3] = currMillis;
    }else{
        printSampahPenuh(1);
        delay(2000);
        printHomePage();
    }
}

void printHomePage(){
    lcd.clear();
    lcd.setCursor(0,0); lcd.print("Smart Trash Can");
    lcd.setCursor(0,1); lcd.print("Andreas Dwimas S");
}

```

```
}
```

```
void printJenisSampah(uint8_t jenis){  
    lcd.clear();  
    lcd.setCursor(0,0); lcd.print("Jenis Sampah :");  
    lcd.setCursor(0,1);  
    (jenis == 0) ? lcd.print("Organik") : lcd.print("Anorganik");  
}
```

```
void printSampahPenuh(uint8_t jenis){  
    lcd.clear();  
    lcd.setCursor(0,0); lcd.print("Maaf Sampah");  
    lcd.setCursor(0,1);  
    (jenis == 0) ? lcd.print("Organik Penuh") :  
    lcd.print("Anorganik Penuh");  
}
```

```
void checkGarbageCondition(){  
    if(stateOrgFull == true){  
        digitalWrite(LED_ORG, 0);  
        if(stateNotifOrg == false){  
            stateNotifOrg = true;  
            SendMessage("[INFO Tempat Sampah  
Pemilah]\n Lokasi : " +  
                        lokasiSampah +  
"\nJenis : Organik \n Status : Penuh");  
        }  
    }  
}
```

```

}else{
    digitalWrite(LED_ORG, 1);
}

if(stateNonOrgFull == true){
    digitalWrite(LED_NON_ORG, 0);
    if(stateNotifNonOrg == false){
        stateNotifNonOrg = true;
        SendMessage("[INFO Tempat Sampah
Pemilah]\n Lokasi : " +
                    lokasiSampah +
"\nJenis : Anorganik \n Status : Penuh");
    }
}else{
    digitalWrite(LED_NON_ORG, 1);
}

void SendMessage(String message){
    SIM800L.println("AT+CMGF=1");
    delay(10);
    SIM800L.println("AT+CMGS=\"083833394997\"\r");
    delay(10);
    SIM800L.println(message);
    delay(10);
    SIM800L.println((char)26);
}

```

```
delay(10);
}

void restoreData(){
    ERROR_TOLERANCE = EEPROM.read(100);
    for(uint8_t i=0;i<SLOT_WARNA;i++){
        red[i]      = EEPROM.read(i);
        green[i]     = EEPROM.read(i +
SLOT_WARNA);
        blue[i]   = EEPROM.read(i + (SLOT_WARNA * 2));
    }
}
```