

IEE2022 Laporan Proyek MAGER - Kelompok 1A



Kelompok 1A:

- | | |
|--|-------------------------|
| 1. Christopher Jonathan | 212100170 / IBDA |
| 2. Dilivio Cullen Lemuel Tilaar | 212100576 / IBDA |
| 3. Wesley Hakim | 212100211 / IEE |

Calvin Institute of Technology

Tahun ajaran 2022/2023

Latar Belakang Masalah

Di zaman modern ini di mana teknologi sudah berkembang dengan pesat, dunia kita menggunakan sangat banyak energi. Penggunaan energi yang tidak efisien menjadi sebuah masalah yang cukup besar karena terbatasnya sumber daya yang bisa kita ambil dan gunakan. Salah satu energi yang sering digunakan dengan tidak efisien adalah energi listrik. Energi listrik digunakan pada berbagai sektor dalam kehidupan kita, salah satunya adalah di dalam tempat tinggal. Di dalam tempat tinggal, penggunaan energi seringkali tidak efisien karena kelalaian manusia, contohnya tidak mematikan lampu setelah dipakai, ataupun lupa mematikan AC.

Penyelesaian Masalah dan Cara Kerja Alat

Solusi dari Masalah

Untuk menanggapi kurang efisiennya penggunaan energi khususnya di sektor tempat tinggal, kelompok kami memberikan sebuah solusi untuk meningkatkan efisiensi penggunaan energi di sektor tempat tinggal yaitu sistem smart room yang kami namai MAGER.

MAGER

Proyek MAGER merupakan sistem smart room yang akan memudahkan pengguna dalam mengakses dan mengatur fitur dalam kamar sesuai dengan kenyamanan, dan preferensi dari pengguna kamar tersebut. MAGER merupakan singkatan dari Monitored, Automated, Green technology, Energy Management, and Remote Accessed, yang merupakan fitur-fitur yang tersedia dalam sistem smart room MAGER. Selain dari itu tujuan utama MAGER adalah untuk mengurangi jumlah energi yang terbuang setiap harinya dengan penambahan fitur otomatisasi sehingga memudahkan pengguna untuk menghemat daya dari ruangan atau bahkan rumah yang telah terinstal perangkat ini.

Komponen-komponen dari MAGER terdiri dari 3 sensor dan 2 aktuator yaitu; PIR sensor, Ultrasonic sensor, dan temperature sensor sebagai sensor, dan lampu LED (RGB), dan AC yang digantikan dengan micro servo sebagai aktuator untuk simulasi. Sensor ultrasonic akan diletakkan di samping pintu untuk melihat apakah ada orang yang melewati atau tidak. PIR sensor akan mendeteksi gerakan yang ada dalam ruangan, diletakkan di tengah ruangan. Temperature sensor akan mendeteksi suhu ruangan dan diletakkan sembarang sesuai keinginan pengguna dalam ruangan.

Cara Kerja Alat

Cara kerja alat ini sebagai berikut:

1. Ultrasonic sensor yang mengukur jarak di depannya
2. Jika jarak terukur berada dalam rentang jarak 20 cm hingga 80 cm, maka akan dianggap orang telah melewati pintu.
3. Jika pintu dilewati maka PIR sensor akan mulai memantau apakah ada gerakan dalam jangkauan sensornya, jika terdapat gerakan maka akan dianggap orang telah memasuki ruangan. Sebaliknya jika tidak terdapat gerakan maka akan dianggap bahwa orang telah meninggalkan ruangan.
4. Sesuai dengan bacaan sebelumnya, bila terdapat orang dalam ruangan, maka lampu akan menyala otomatis atau mati otomatis jika orang meninggalkan ruangan.
5. Jika terdapat orang dalam ruangan, temperature sensor akan terus memantau suhu ruangan.
6. Jika suhu ruangan di atas 28°C maka AC akan dinyalakan. Jika suhu dibawah atau sama dengan 28°C, AC akan dimatikan.

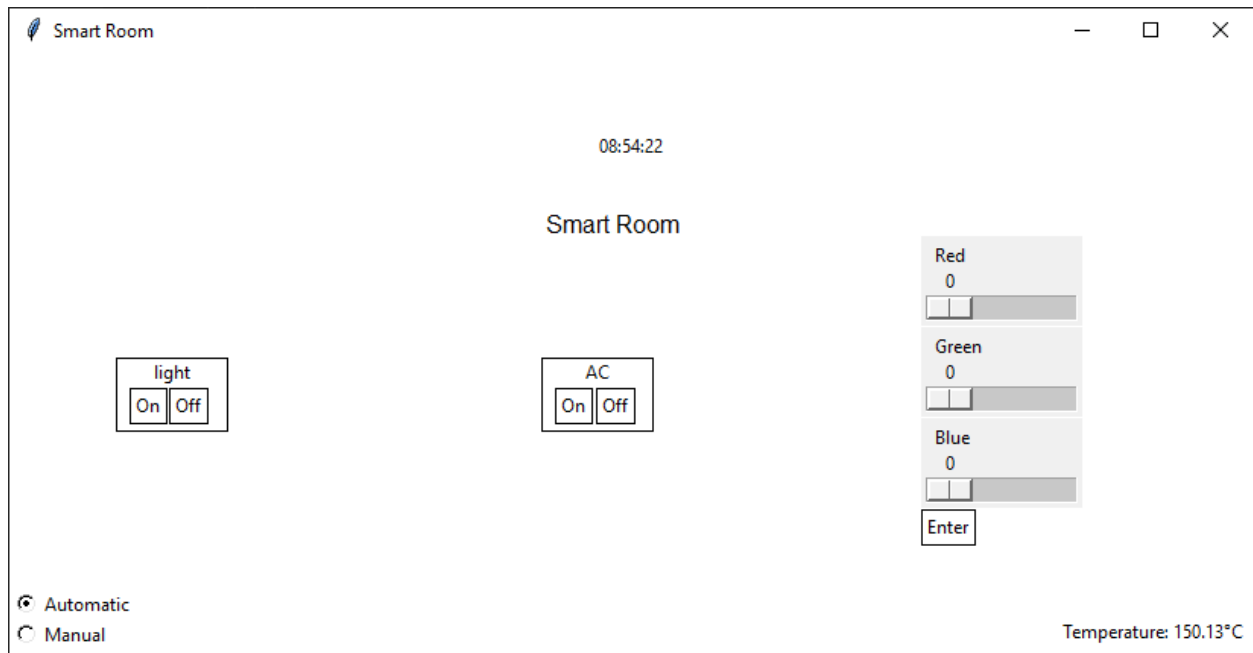
Perangkat MAGER yang kami buat ini juga terhubung dengan UI yang digunakan untuk mengubah mode dari otomatis menjadi manual, menampilkan jam, dan suhu secara langsung. Dalam mode manual kita bisa mengatur fitur aktuator yang telah tersedia sesuka hati. Sistem juga dilengkapi dengan database yang akan mencatat aktivitas sensor yang terdapat dalam kamar tersebut, orang yang masuk dan keluar dalam kamar tersebut, hasil pengukuran dari sensor, dan juga status dari aktuator yang tersedia sehingga penggunaan ruangan dapat dimonitor lebih mudah.

Komunikasi yang digunakan

Dalam sistem smart home MAGER, untuk menghubungkan dua papan pengembang Arduino Uno R3 kami menggunakan komunikasi I2C yang dimana Arduino Master terhubung kepada seluruh sensor, sedangkan Arduino Slave terhubung ke semua aktuator. Arduino Master akan mengirim data ke Arduino Slave jika terdapat bacaan tertentu sehingga aktuator dapat digerakkan sesuai bacaan sensor.













UI

Tampilan UI sistem smart room MAGER



Database

Tampilan database sistem smart room MAGER

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Date	date			No	None			 Change  Drop More
2	Time	time(6)			No	None			 Change  Drop More
3	Person	int(2)			No	None			 Change  Drop More
4	Temperature	int(5)			No	None			 Change  Drop More
5	AC	int(1)			No	None			 Change  Drop More
6	LED	int(1)			No	None			 Change  Drop More

Bill of Materials

Nama Komponen	Jumlah	Harga satuan (Rp)	Harga total (Rp)	Informasi pembelian
Arduino Uno R3	2	70,000	140,000	<a href="https://tokopedia.l
ink/oxuUYPKOlz
b">https://tokopedia.l ink/oxuUYPKOlz b
Breadboard	1	10,000	10,000	<a href="https://tokopedia.l
ink/H0KdOsQOlz
b">https://tokopedia.l ink/H0KdOsQOlz b
Kabel Jumper Male to Male	50	330	16,500	<a href="https://tokopedia.l
ink/DM2ZU9UOlz
b">https://tokopedia.l ink/DM2ZU9UOlz b
Kabel Jumper Female to Male	5	330	1,650	<a href="https://tokopedia.l
ink/UH7QFw2Olz
b">https://tokopedia.l ink/UH7QFw2Olz b
Ultrasonic Sensor HC-SR04	1	10,000	10,000	<a href="https://tokopedia.l
ink/uEpLJc5Olzb">https://tokopedia.l ink/uEpLJc5Olzb
PIR Sensor	1	13,000	13,000	<a href="https://tokopedia.l
ink/sbnobqnPlzb">https://tokopedia.l ink/sbnobqnPlzb
Temperature Sensor LM35	1	5,850	5,850	<a href="https://tokopedia.l
ink/tJje9dqPlzb">https://tokopedia.l ink/tJje9dqPlzb
Micro Servo SG90	1	14,400	14,400	<a href="https://tokopedia.l
ink/QVkhEcsPlzb">https://tokopedia.l ink/QVkhEcsPlzb
RGB LED	1	1,500	1,500	<a href="https://tokopedia.l
ink/sCrHIJuPlzb">https://tokopedia.l ink/sCrHIJuPlzb
Resistor 330Ω	3	90	270	<a href="https://tokopedia.l
ink/1fEFOCyPlzb">https://tokopedia.l ink/1fEFOCyPlzb
USB A to B cable	2	2,000	4,000	<a href="https://tokopedia.l
ink/oxuUYPKOlz
b">https://tokopedia.l ink/oxuUYPKOlz b
Jumlah pengeluaran			217,170	

Rangkaian

Foto rangkaian fisik

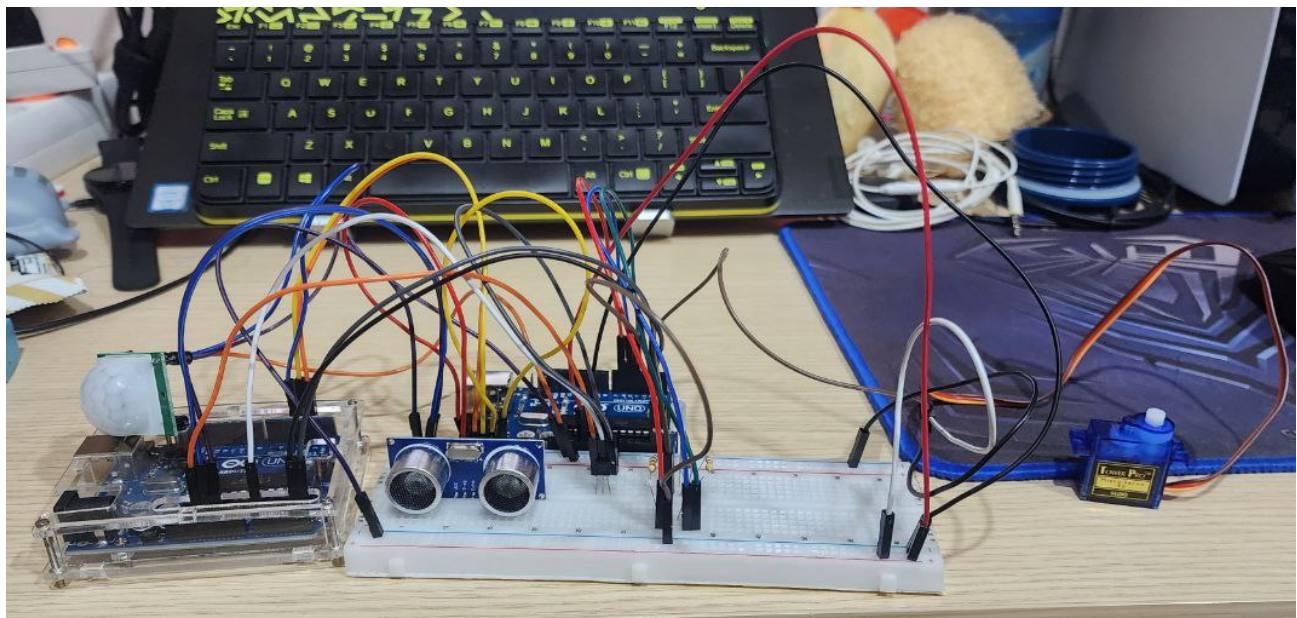
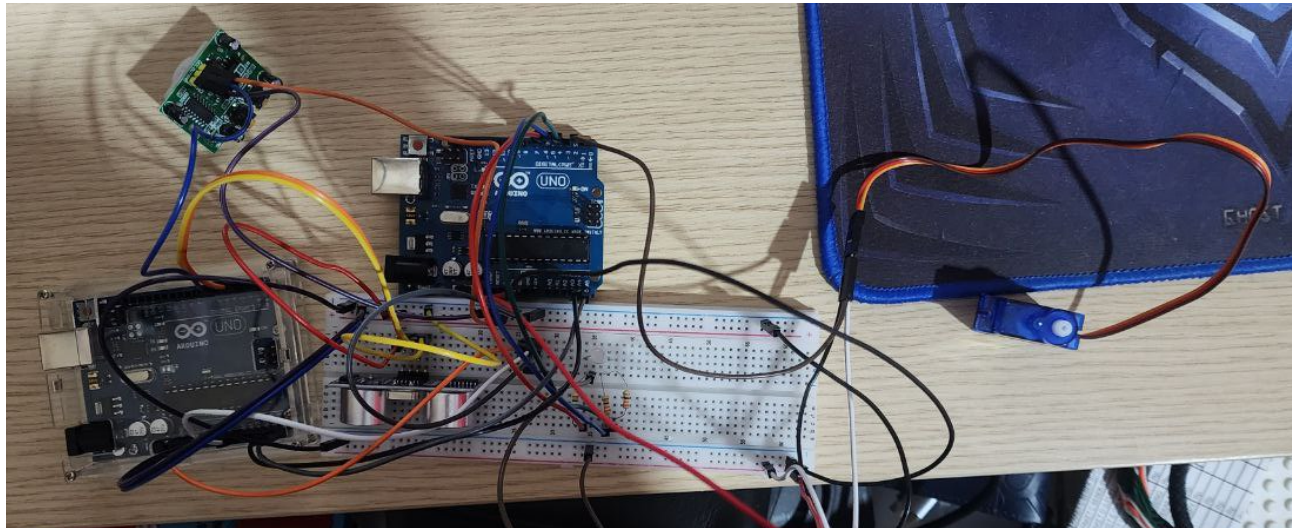
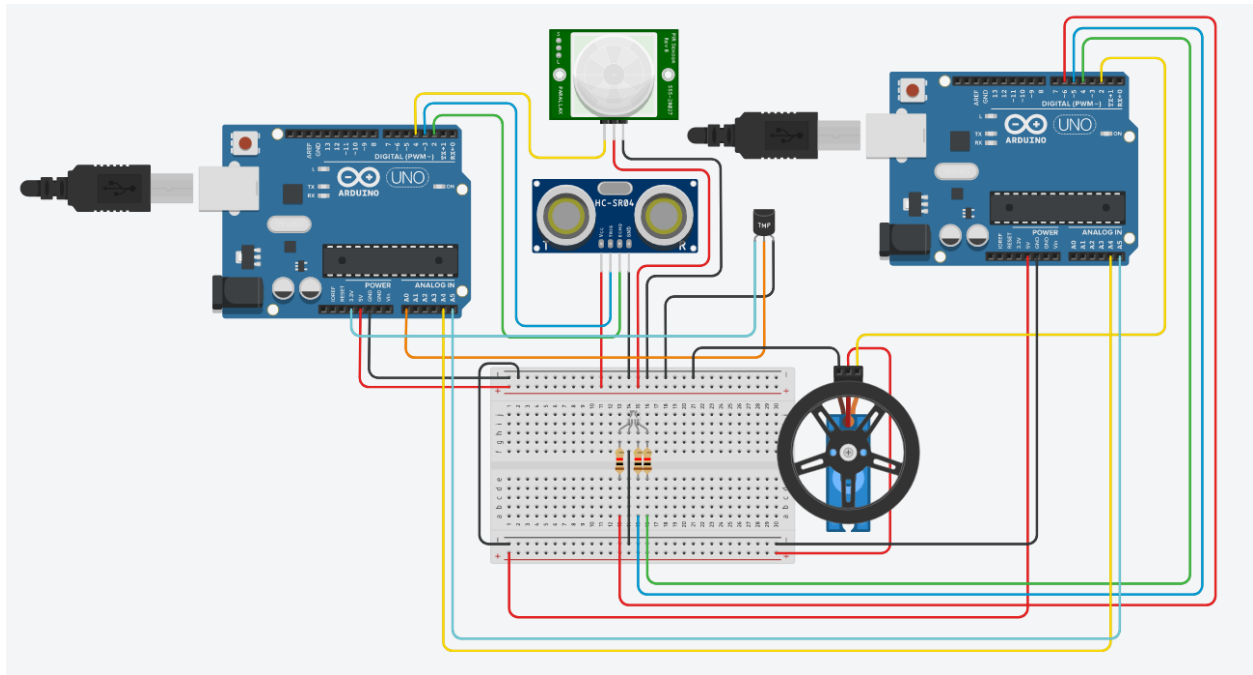


Foto rangkaian tinkercad



Tautan tinkercad

<https://www.tinkercad.com/things/0H7M4bGShf6>

Kode dan Perangkat Lunak

Dalam sistem smart room MAGER ini kami menggunakan bahasa pemrograman python dan juga C++. Software yang kami gunakan adalah Visual Studio untuk pemrograman UI dan database dalam bahasa python. Untuk UI kami menggunakan library tkinter, dan untuk database kami menggunakan library sqlite3. Untuk pemrograman Arduino kami menggunakan software Arduino IDE dan library Wire.h untuk menghubungkan antar Arduino dengan cara I2C. Library Servo.h juga digunakan untuk menggerakkan micro servo.

Kode UI dan Database

```
import tkinter as tk
from datetime import datetime
import serial
import sqlite3

ser = serial.Serial('COM4', 9600)
ser.readline()
conn = sqlite3.connect("smartroom.db")
c = conn.cursor()
root = tk.Tk()
root.title("Smart Room")
root.geometry("800x400")
root.configure(bg="white")

current_time = datetime.now().strftime('%H:%M:%S')
current_date = datetime.now().strftime('%Y-%m-%d')

#Title
title = tk.Label(root, text="Smart Room", bg="white", font=(20))
title.place(x=350, y=100)

#Clock
clock = tk.Label(root, text="", bg="white")
def update_clock():
    current_time = datetime.now().strftime("%H:%M:%S")
    clock.config(text=current_time)
    root.after(1000, update_clock)
clock.place(x=385, y=50)

#Automatic Switch
```

```

selected_option = tk.StringVar(value="Auto")
def auto_command():
    command = selected_option.get()
    if command == "Auto":
        ser.write(b"3,1\n")
        print("Automatic-mode")
    else:
        ser.write(b"3,0\n")
        print("Manual-mode")
automatic = tk.Radiobutton(root, text="Automatic",
variable=selected_option, value="Auto", bg="white", command=auto_command)
manual = tk.Radiobutton(root, text="Manual", variable=selected_option,
value="Manual", bg="white", command=auto_command)
automatic.place(x=0, y=350)
manual.place(x=0, y=370)

#Light
def l_on_button():
    n = selected_option.get()
    if n == "Manual":
        data_to_send = "1,1"
        ser.write(data_to_send.encode("utf-8"))
        print("Lights On")
    else:
        print("Auto-mode please switch if you want to manually turn on the
lights")

def l_off_button():
    n = selected_option.get()
    if n == "Manual":
        data_to_send = "1,0"
        ser.write(data_to_send.encode())
        print("Lights Off")
    else:
        print("Auto-mode please switch if you want to manually turn off the
lights")
        print(ser.readline().decode().strip())

light = tk.Label(root, text="light", bg="white", borderwidth=0.5,
relief="solid", width=10, height=3, anchor="n")

```

```

light_on = tk.Button(root, text="On", bg="white", borderwidth=0.5,
relief="solid", command=l_on_button)
light_off = tk.Button(root, text="Off", bg="white", borderwidth=0.5,
relief="solid", command=l_off_button)
light.place(x=70,y=200)
light_on.place(x=79, y=220)
light_off.place(x=105, y=220)

#AC
def ac_on_button():
    n = selected_option.get()
    if n == "Manual":
        data_to_send = "0,1"
        ser.write(data_to_send.encode())
        print("AC On")
    else:
        print("Auto-mode please switch if you want to manually turn on the
AC")

def ac_off_button():
    n = selected_option.get()
    if n == "Manual":
        data_to_send = "0,0"
        ser.write(data_to_send.encode())
        print("AC Off")
    else:
        print("Auto-mode please switch if you want to manually turn off the
AC")

ac = tk.Label(root, text="AC", bg="white", borderwidth=0.5,
relief="solid", width=10, height=3, anchor="n")
ac_on = tk.Button(root, text="On", bg="white", borderwidth=0.5,
relief="solid", command=ac_on_button)
ac_off = tk.Button(root, text="Off", bg="white", borderwidth=0.5,
relief="solid", command=ac_off_button)
ac.place(x=350, y=200)
ac_on.place(x=359, y=220)
ac_off.place(x=386, y=220)

#RGB

```

```

def enter_button():
    r = red_scale.get()
    g = green_scale.get()
    b = blue_scale.get()
    data_to_send = f"2,{r},{g},{b}"
    ser.write(data_to_send.encode())
    print(f"red:{r}, green:{g}, blue:{b}")
rgb = tk.Label(root, text="AC", bg="white", borderwidth=0.5,
relief="solid", width=10, height=3, anchor="n")

red_scale = tk.Scale(root, from_=0, to=255, orient="horizontal",
label="Red")
green_scale = tk.Scale(root, from_=0, to=255, orient="horizontal",
label="Green")
blue_scale = tk.Scale(root, from_=0, to=255, orient="horizontal",
label="Blue")
enter = tk.Button(root, text="Enter", bg="white", borderwidth=0.5,
relief="solid", command=enter_button)

rgb.place()
red_scale.place(x=600, y=120)
green_scale.place(x=600, y=180)
blue_scale.place(x=600, y=240)
enter.place(x=600, y=300)

#Temprature
# data= (19,1)
temperature = tk.Label(root, text="", bg="white")
def update_temp():
    unit = "\u00b0C"
    # current_temp= data[0]
    data_from_arduino = ser.readline().decode().strip()
    person, temp, ac, led = data_from_arduino.split(",")
    query = "INSERT into room1 (Date, Time, Person, Temperature, AC, LED)
VALUES (?, ?, ?, ?, ?, ?)"
    c.execute(query, (current_date, current_time, person, temp, ac, led))

    conn.commit()
    try:

```

```

        current_temp = float(temp)
        temperature.config(text=f"Temperature: {current_temp}{unit}")
    except ValueError:
        temperature.config(text=f"Temperature: {current_temp}{unit}")
    except TypeError:
        temperature.config(text=f"Temperature: {current_temp}{unit}")
    root.after(5000, update_temp)
temperature.place(x=690, y=370)
print(ser.readline().decode().strip())

root.after(0, update_temp)
root.after(0, update_clock)
root.mainloop()

```

Kode Arduino Master

```

#include <Wire.h>
const int echoUltrasonic = 2;
const int trigUltrasonic = 3;
const int PIR = 4;
int tempSensor = A0;
float duration = 0;
float distanceCM = 0;
int movementExist = 0;
int personPass = 0;
int personExist = 0;
float celcius = 0;
unsigned long currentTime = 0;
unsigned long startTime = 0;
int autoMode = 1;
int ACStatus = 0;
int LEDStatus = 0;
int personExistBefore = 0;
String dataPython = "";

void setup() {
    pinMode(trigUltrasonic, OUTPUT);
    pinMode(echoUltrasonic, INPUT);
    pinMode(PIR, INPUT);
    pinMode(tempSensor, INPUT);
    Wire.begin();
}

```

```

    Serial.begin(9600);
}

int checkUltrasonic(){
    int ultrasonicStatus = 0;
    digitalWrite(trigUltrasonic, LOW);
    delayMicroseconds(2);
    digitalWrite(trigUltrasonic, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigUltrasonic, LOW);
    duration = pulseIn(echoUltrasonic, HIGH);
    distanceCM = duration*0.034/2;
    // Serial.println(distanceCM);
    if (distanceCM <= 80 && distanceCM >= 20) {
        ultrasonicStatus = 1;
    }
    return ultrasonicStatus;
}

int checkPIR(){
    startTime = millis();
    currentTime = millis();
    while ((currentTime - startTime)<= 2000){
        movementExist = digitalRead(PIR);
        if (movementExist == HIGH){
            return 1;
        }
        currentTime = millis();
    }
    return 0;
}

float getTemp(){
    float reading = analogRead(tempSensor);
    float voltage = reading * (3.3 / 1024.0);
    float roomTemperature = ((voltage - 0.5) * 100);
    return roomTemperature;
}

int printAllStatus(){

```

```

Serial.print(personExist);
Serial.print(",");
Serial.print(getTemp());
Serial.print(",");
Serial.print(ACStatus);
Serial.print(",");
Serial.print(LEDStatus);
Serial.print("\n");
}

int actuatorAC(int state){
    if (ACStatus != state){
        Wire.beginTransmission(0x22);
        Wire.write(0);
        Wire.write(state);
        Wire.endTransmission();
        ACStatus = state;
        printAllStatus();
    }
}

int actuatorLED(int state){
    if (LEDStatus != state){
        Wire.beginTransmission(0x22);
        Wire.write(1);
        Wire.write(state);
        Wire.endTransmission();
        LEDStatus = state;
        printAllStatus();
    }
}

int actuatorRGB(String string){
    Wire.beginTransmission(0x22);
    Wire.write(1);
    String value = "";
    for (int i = 0; i < string.length(); i++) {
        if (i != ","){
            value += i;
        }
    }
}

```

```

    else{
        Wire.write(value);
        value = "";
    }
}
Wire.endTransmission();
}

```

```

int checkTemp(int roomTemperature){
    if (roomTemperature >= 28){
        actuatorAC(1);
    }
    else{
        actuatorAC(0);
    }
}

```

```

int dataChecker(String string){
// Serial.println(string);
    int command = string[0];
    Serial.print("tipe aktuator");
    Serial.println(command);
    if (command == 0){
        Serial.println(string[2]);
        actuatorAC(string[2]);
        //atur ac
    }
    else if (command == 1){
        Serial.println(string[2]);
        actuatorLED(string[2]);
        //atur lampu
    }
    else if (command == 2){
// actuatorRGB(string);
        //atur rgb
    }
    else if (command == 3){
        Serial.println(string[2]);
        int autoMode = string[2];
        //atur mode
    }
}

```



```

    }
}

int receiveData() {
    if (Serial.available() > 0) { // check if data is available to read
        String dataPython = Serial.readString();
        dataChecker(dataPython);
    }
}

void loop() {
    receiveData();
    if (autoMode == 1) {
        personPass = checkUltrasonic();
        if (personPass == 1) {
            personExistBefore = personExist;
            personExist = checkPIR();
            if (personExistBefore != personExist) {
                printAllStatus();
            }
        }
        if (personExist == 1) {
            //Serial.println("Person detected");
            actuatorLED(1);
            celcius = getTemp();
            checkTemp(celcius);
        }
        else {
            actuatorLED(0);
            actuatorAC(0);
        }
    }
}
}

```

Kode Arduino Slave

```

#include <Wire.h>
#include <Servo.h>
const int rgbGreen = 4;
const int rgbBlue = 5;
const int rgbRed = 6;

```

```

int RValue = 0;
int GValue = 0;
int BValue = 0;
int actuatorType;
int state = 0;
Servo AC;

void setup()
{
  AC.attach(2);
  pinMode(rgbGreen, OUTPUT);
  pinMode(rgbBlue, OUTPUT);
  pinMode(rgbRed, OUTPUT);
  Wire.begin(0x22);
  Wire.onReceive(receiveEvent);
  Serial.begin(9600);
  Serial.println("start slave");
}

void loop()
{
  delay(100);
}

void receiveEvent(int howMany){
  actuatorType = Wire.read();
  Serial.print("actuator: ");
  Serial.print(actuatorType);
  Serial.print(", ");
  if (actuatorType == 0){
    state = Wire.read();
    Serial.println(state);
    if (state == 1){
      AC.write(45);
    }
    else{
      AC.write(90);
    }
  }
  else if (actuatorType = 1){

```

```
state = Wire.read();
Serial.println(state);
if (state == 1){
    analogWrite(rgbGreen, 255);
    analogWrite(rgbBlue, 255);
    analogWrite(rgbRed, 255);
}
else{
    analogWrite(rgbGreen, 0);
    analogWrite(rgbBlue, 0);
    analogWrite(rgbRed, 0);
}
}
else if (actuatorType = 2){
    RValue = Wire.read();
    GValue = Wire.read();
    BValue = Wire.read();
    analogWrite(rgbGreen, GValue);
    analogWrite(rgbBlue, BValue);
    analogWrite(rgbRed, RValue);
}
}
```

Seluruh kode dapat diakses dalam file .zip

Penilaian Anggota Kelompok

- Anggota kelompok 1: 212100170, Christopher Jonathan
- Anggota kelompok 2: 212100576, Dilivio Cullen Lemuel Tilaar
- Anggota kelompok 3: 212100211, Wesley Hakim

	Kontribusi menurut anggota kelompok 1	Kontribusi menurut anggota kelompok 2	Kontribusi menurut anggota kelompok 3	Rata-rata kontribusi anggota kelompok
Anggota Kelompok 1	33.34%	33.33%	33.33%	33.33%
Anggota Kelompok 2	33.33%	33.34%	33.33%	33.33%
Anggota Kelompok 3	33.33%	33.33%	33.34%	33.33%
	Total = 100%	Total = 100%	Total = 100%	Total = 100%