LEMBAR KERJA PROYEK
MIKROPROSESOR DAN MIKROKONTROLER
KODE MATA KULIAH: CCE61307

SISTEM KEAMANAN CERDAS BERBASIS SENSOR RADAR ULTRASONIK DAN SENSOR SUARA UNTUK PENGAWASAN MALAM HARI

OLEH :

Abdul Hakim Sidik Lessy	235150319111001
Aqilah Akma	235150301111017
Evika Fatika Sari	235150307111018
Fawwas Aliy	235150300111009

FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG

2024

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Keamanan adalah aspek penting dalam kehidupan, terutama pada malam hari ketika risiko ancaman meningkat. Dengan meningkatnya insiden pencurian dan intrusi, dibutuhkan sistem keamanan yang efisien. Teknologi sensor modern seperti LDR, radar ultrasonik, dan sensor suara memungkinkan pengembangan sistem yang dapat mendeteksi keberadaan dan kebisingan, serta memberikan peringatan visual dan audio. Sistem ini dirancang untuk beroperasi otomatis dalam kondisi gelap, sehingga memberikan perlindungan yang optimal dan meningkatkan rasa aman bagi pengguna.

1.2 Konsep Sistem

1.2.1 Penjelasan Konsep Sistem

Sistem keamanan ini dirancang untuk beroperasi pada malam hari dengan memanfaatkan sensor LDR (Light Dependent Resistor) yang berfungsi untuk mendeteksi perubahan tingkat cahaya. Ketika sensor LDR mengidentifikasi bahwa kondisi lingkungan sudah gelap, sistem akan diaktifkan dan mulai melakukan pemantauan.

Dalam sistem ini, terdapat dua jenis sensor pemantauan utama:

a. **Sensor Radar Ultrasonik**: Sensor ini bertugas untuk mendeteksi keberadaan entitas di dalam ruangan serta mengukur jarak antara sensor dan objek yang terdeteksi. Dengan kemampuan ini, sensor radar ultrasonik dapat memberikan informasi penting mengenai posisi objek dalam area yang diawasi.
b. **Sensor Suara**: Sensor ini berfungsi untuk mendeteksi suara yang berada dalam ruangan. Jika tingkat kebisingan yang terdeteksi melebihi ambang batas yang telah ditentukan, sensor suara akan memberikan sinyal ke sistem untuk menanggapi situasi tersebut.

Dalam kondisi normal atau idle, sistem keamanan akan memberikan indikasi visual bahwa ia berfungsi dengan baik. Hal ini ditunjukkan dengan lampu LED yang mati, menandakan bahwa sistem dalam keadaan siap dan aktif.

Namun, apabila salah satu sensor (baik sensor radar ultrasonik maupun sensor suara) terpicu, sistem akan memasuki kondisi peringatan (warning). Dalam situasi ini, lampu LED akan mengalami interupsi dan mulai berkedip cepat. Ini berfungsi sebagai sinyal visual yang jelas untuk menarik perhatian pengguna. Selain itu, sistem juga akan mengaktifkan buzzer sebagai alarm, memberikan sinyal suara yang menandakan adanya potensi ancaman atau situasi tidak normal di dalam ruangan.

Dengan kombinasi antara deteksi cahaya, pengukuran jarak, dan pengenalan suara, sistem keamanan ini mampu memberikan perlindungan yang efektif di malam hari, serta memberikan respons yang jelas melalui indikator visual dan audio saat terjadi kondisi yang memerlukan perhatian lebih.

1.2.2 Output Sistem

Output dari sistem keamanan yang dijelaskan mencakup beberapa elemen berikut:

- Deteksi Kegelapan: Sensor LDR akan mendeteksi kondisi pencahayaan. Ketika LDR mendeteksi bahwa kondisi sudah gelap, sistem akan aktif untuk memulai pemantauan.
- Pemantauan Entitas: Sensor radar ultrasonik akan mengukur jarak ke objek dalam ruangan. Jika ada entitas yang terdeteksi dalam jarak tertentu, sistem akan memicu tindakan lebih lanjut.
- Deteksi Suara: Sensor suara akan mendeteksi kebisingan. Jika suara terdeteksi melebihi ambang batas yang ditentukan, ini juga akan memicu respons sistem.
- Kondisi Warning: jika salah satu sensor terpicu, LED akan berkedip lebih cepat dan buzzer akan aktif, memberikan sinyal suara untuk memperingatkan pengguna tentang potensi ancaman atau ketidaknormalan di ruangan.

1.3 Tujuan

Tujuan dari sistem keamanan ini adalah untuk menyediakan perlindungan di malam hari dengan memanfaatkan teknologi sensor. Secara spesifik, tujuan sistem ini meliputi:

- Deteksi Otomatis: Menggunakan sensor LDR untuk secara otomatis mendeteksi kondisi gelap dan mengaktifkan sistem keamanan.
- Indikasi Sistem Normal: Memberikan indikasi visual yang jelas melalui LED yang berkedip, menunjukkan bahwa sistem dalam kondisi siap dan berfungsi dengan baik.
- Peringatan Cepat: Mengaktifkan alarm dan memberikan sinyal visual melalui LED yang berkedip lebih cepat saat ada potensi ancaman, sehingga pengguna dapat segera merespons situasi darurat.
- Menampilan serial output pada terminal di device dan antarmuka pengguna grafis (GUI) yang menampilkan hasil dari radar ultrasonik.

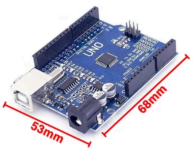
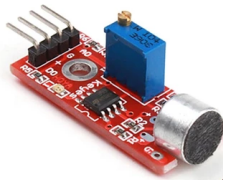
1.4 Perlengkapan Software

Berikut ini adalah daftar perangkat lunak yang diperlukan beserta penjelasan singkat mengenai masing-masing perangkat lunak tersebut:

1. *Arduino IDE 2.3.3*: perangkat lunak untuk memprogram mikrokontroler Arduino. IDE ini menawarkan antarmuka pengguna yang intuitif, fitur pemformatan kode otomatis, dan dukungan untuk berbagai papan Arduino. Pengguna dapat menulis, mengedit, dan mengunggah kode, serta memantau port serial untuk proyek berbasis sensor dan aktuator.
2. *Processing-4.3*: lingkungan pemrograman open-source yang dirancang untuk seniman dan desainer. Dengan fokus pada visualisasi data dan interaksi, ia menyediakan alat untuk membuat grafik, animasi, dan aplikasi multimedia.
3. *Fritzing 0.9.3*: Perangkat lunak ini digunakan untuk merancang dan membuat diagram rangkaian elektronika secara visual. Dengan Fritzing, pengguna dapat menggambar rangkaian pada breadboard dan menyusun skematik yang detail, sehingga memudahkan dalam memahami serta mendokumentasikan proyek elektronik.

1.5 Komponen Hardware

Berikut adalah tabel yang mencantumkan nama dan spesifikasi perangkat keras.

NO	NAMA	TIPE	CONTOH GAMBAR	JUMLAH & HARGA SATUAN	TOTAL HARGA (Rupiah)
1	Mikrokontroller + USB Cable	ARDUINO UNO R3 ATMEGA32 P		1x 57.900	57.900
	https://www.tokopedia.com/cncstorebandung/arduino-uno-r3-atmega328p-smd-ch340-compatible-board-plus-usb-cable				
2	Sensor Ultrasonic	HC-SR04		1x 12.000	12.000
	https://www.tokopedia.com/starlectric/hc-sr04-sensor-ultrasonik-ultrasonic-ping				
3	Sensor Cahaya	SEN-0012		1x 5.800	5.800
	https://www.tokopedia.com/cncstorebandung/cnc-photosenstive-light-ldr-resistance-sensor-cahaya-module				
4	Sensor Suara	KY-037		1x 6.000	6.000
	https://www.tokopedia.com/best-ay/sensor-suara-sound-detection-whistle-activated-siulan-fc-04-arduino				
5	Micro Servo	MTR-0002		1x 15.500	15.500

	https://www.tokopedia.com/cncstorebandung/cnc-towerpro-motor-servo-sg90-sg-90-9g				
6	Breadboard	Half Size 400 Lubang		2x 7.800	15.600
	https://www.tokopedia.com/cncstorebandung/cnc-breadboard-mini-solderless-400-400p				
7	40 pcs Jumper Cable	20CM M to M & F to M		1x 17.000	17.000
	https://www.tokopedia.com/cncstorebandung/40pcs-kabel-jumper-cable-40cm-male-male-female-female-male-female-female-t-female-3e81c				
8	10 pcs Resistor	220 Ohm		1x 1000	1000
	https://www.tokopedia.com/cncstorebandung/cnc-10x-resistor-220ohm-220r-220-14w-1-metal-film				
9	Buzzer	Aktif 5V		1x 1.500	1.500
	https://www.tokopedia.com/cncstorebandung/cnc-buzzer-speaker-active-5v-for-arduino-uno-mega-mini-nano				
10	10 pcs LED	Red 3mm		1x 1.500	1.500
	https://www.tokopedia.com/cncstorebandung/10pcs-led-3mm-f3-super-bright-red-merah-diffused				
Total					133.800

Pada tabel diatas merupakan bahan perangkat keras yang dibutuhkan. Berikut ini adalah penjelasan serta fungsi dari setiap perangkat keras tersebut:

1. **Mikrokontroler + USB Cable**: Mikrokontroler yang terhubung melalui kabel USB mentransfer data dan daya antara mikrokontroler dan komputer. USB berfungsi sebagai jalur komunikasi untuk pemrograman dan monitoring data secara langsung dari komputer. Saat terhubung, komputer mengirimkan instruksi program melalui kabel USB ke mikrokontroler, yang kemudian menyimpan dan menjalankan instruksi tersebut. Mikrokontroler mengontrol sensor dan aktuator berdasarkan program yang diunggah, dan dapat mengirimkan data kembali ke komputer, memungkinkan monitoring atau pengaturan ulang sistem sesuai kebutuhan.
2. **Sensor Ultrasonic**: Sensor ultrasonik bekerja dengan mengirimkan gelombang suara frekuensi tinggi dan mengukur waktu yang dibutuhkan gelombang tersebut untuk memantul kembali setelah mengenai objek. Berdasarkan waktu yang ditempuh gelombang, sensor menghitung jarak antara sensor dan objek tersebut. Dalam sistem keamanan, sensor ini mendeteksi keberadaan entitas dalam ruangan dengan mengukur jarak objek di area yang diawasi dan memberi sinyal jika objek terdeteksi dalam jarak tertentu.
3. **Sensor Cahaya**: Sensor cahaya, atau LDR (Light Dependent Resistor), bekerja dengan mengubah resistansi berdasarkan intensitas cahaya yang diterimanya. Dalam sistem keamanan ini, LDR mendeteksi kondisi pencahayaan sekitar, contohnya ketika lingkungan menjadi gelap, resistansi meningkat, dan ini memicu sistem untuk aktif, memulai pemantauan otomatis pada malam hari.
4. **Sensor Suara**: Sensor suara bekerja dengan mendeteksi tingkat kebisingan. Ketika suara melebihi ambang batas, sensor ini mengirim sinyal ke mikrokontroler untuk memicu peringatan, seperti LED berkedip cepat dan buzzer berbunyi, guna meningkatkan keamanan di area tersebut.
5. **Micro Servo**: Micro servo adalah motor kecil yang dapat memutar sudut tertentu berdasarkan sinyal PWM (Pulse Width Modulation) yang diterimanya dari mikrokontroler. Ketika sinyal dikirim, servo mengatur posisinya sesuai dengan durasi pulsa, memungkinkan kontrol akurat atas gerakan, seperti membuka atau menutup pintu otomatis dalam sistem.
6. **Breadboard**: Breadboard alat untuk merakit dengan memasukkan komponen resistornya ke dalam lubang-lubang yang terhubung secara elektrik,dan kabel jumper untuk menghubungkan komponen.
7. **Jumper Cable**: Jumper cable digunakan untuk menghubungkan berbagai komponen dalam sirkuit pada breadboard.
8. **Resistor**: Resistor adalah komponen yang membatasi aliran arus listrik dalam sirkuit untuk melindungi komponen dari arus yang berlebih. Dengan menjaga kinerja sirkuit agar tetap stabil dan aman.
9. **Buzzer**: Buzzer diaktifkan oleh sinyal dari mikrokontroler ketika sensor radar ultrasonik atau suara mendeteksi ancaman. Saat aktif, buzzer mengubah sinyal listrik menjadi gelombang suara, dengan menghasilkan alarm yang memberitahukan peringatan kepada user.
10. **LED**: Berfungsi untuk menunjukkan status operasional, ketika kondisi normal semua LED akan padam, menandakan sistem berfungsi dengan baik. Jika salah satu sensor terpicu, LED akan berkedip lebih cepat sebagai sinyal peringatan, memberitahukan user terhadap potensi ancaman.

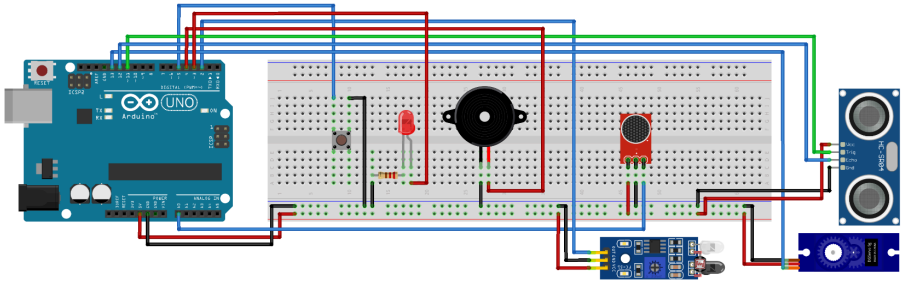
BAB II
METODE

2.1 Rangkaian Elektronik

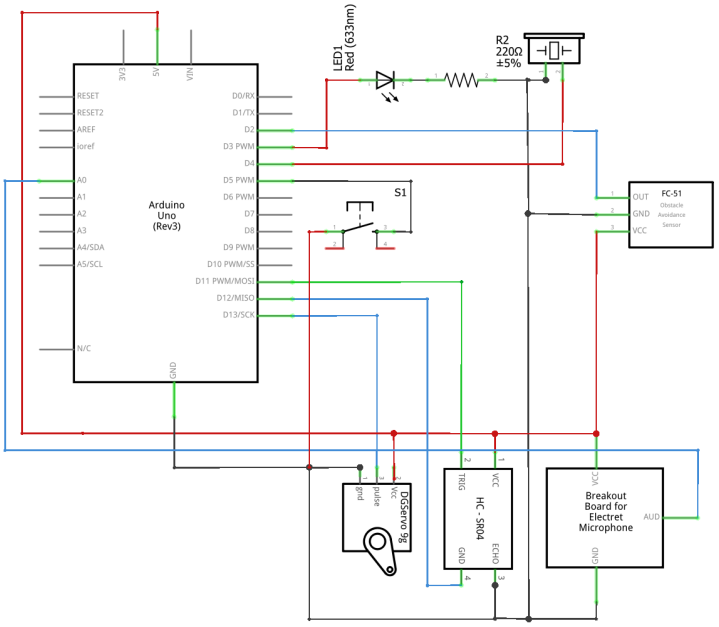
Dalam merancang rangkaian elektronika yang digunakan dalam sistem ini, kami memanfaatkan perangkat lunak Fritzing versi 0.9.3. Perangkat lunak ini memungkinkan untuk membuat representasi visual dari rangkaian yang akan digunakan. Melalui Fritzing, kami dapat menggambarkan rangkaian pada breadboard serta menyusun skematik secara detail.

Berikut ini adalah representasi dari rangkaian tersebut dalam bentuk diagram breadboard dan skematik yang dihasilkan menggunakan Fritzing 0.9.3.

- Circuit Breadboard



- Skematik



Rangkaian ini dirancang untuk mengintegrasikan berbagai sensor dan perangkat output dengan Arduino Uno. Berikut adalah penjelasan mengenai rangkaian tersebut:

- **Koneksi LDR**: LDR terhubung ke salah satu pin digital pada Arduino (3) untuk membaca nilai resistansi yang berubah sesuai dengan tingkat cahaya. Saat kondisi gelap, nilai resistansi LDR akan meningkat, yang akan dideteksi oleh Arduino dan mengaktifkan sistem.
- **Koneksi Sensor Ultrasonik**: Sensor ultrasonik terhubung ke pin digital Arduino (12 & 11) untuk mengirim dan menerima gelombang ultrasonik. Arduino mengukur waktu yang dibutuhkan gelombang untuk kembali, lalu menghitung jarak objek.
- **Koneksi Sensor Suara**: Sensor suara terhubung ke pin analog Arduino (A0), tergantung pada jenis sensor yang digunakan. Ketika suara terdeteksi melebihi ambang batas yang ditentukan, sensor ini mengirimkan sinyal ke Arduino.
- **Koneksi LED dan Buzzer**: LED terhubung ke pin digital Arduino (2 & 4) dengan resistor untuk membatasi arus. Buzzer juga terhubung ke pin digital Arduino, dan diaktifkan ketika ada kondisi peringatan.
- **Koneksi Button**: Button terhubung ke pin digital Arduino (5) untuk mematikan mode warning.
- **Sumber Daya**: Rangkaian ini menggunakan 5V Arduino untuk menyediakan daya ke seluruh sistem dari daya yang tersambung pada device (laptop/pc).

2.2 Kode Program Mikrokontroler

Kode program memainkan peran yang sangat penting dalam sistem ini, berfungsi sebagai otak yang mengendalikan seluruh operasi dan penghubung dengan device. Kode tersebut akan disimpan dalam mikrokontroler (ATmega), yang kemudian akan mengatur dan mengkoordinasikan fungsi-fungsi sistem.

Berikut kode program yang akan disimpan pada ATmega.

main.c

```
1   #include <Servo.h>
2
3   const int trigPin = 11;
4   const int echoPin = 12;
5   const int soundPin = A0;
6   const int ldrPin = 2;
7   const int ledPin = 3;
8   const int buzzerPin = 4;
9   const int buttonPin = 5;
10
11  Servo myServo;
12
13  long distance = 0;
14  long sound;
15  unsigned long startTime = 0;
16  const unsigned long WARNING_TIMEOUT = 300000;
17  bool isTimerActive = false;
18  const int LED_INTERVAL = 100;
19  const int MIN_FREQ = 500;
20  const int MAX_FREQ = 2000;
21  int freqStep = 100;
22
23  void setup()
```

```
{
    Serial.begin(9600);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(ldrPin, INPUT);
    pinMode(soundPin, INPUT);
    pinMode(ledPin, OUTPUT);
    pinMode(buzzerPin, OUTPUT);
    pinMode(buttonPin, INPUT_PULLUP);
    myServo.attach(13);
    startTime = millis();
    isTimerActive = true;
}

void loop()
{
    for (int i = 15; i <= 115; i++)
    {
        checkTimer();
        if (digitalRead(ldrPin) == HIGH)
            moveServoCheck(i);
        else
            off();
    }
    for (int i = 115; i > 15; i--)
    {
        checkTimer();
        if (digitalRead(ldrPin) == HIGH)
            moveServoCheck(i);
        else
            off();
    }
}

void checkTimer()
{
    if (isTimerActive)
    {
        unsigned long currentTime = millis();
        unsigned long elapsedTime = currentTime -
startTime;
        if (elapsedTime >= WARNING_TIMEOUT)
            triggerTimeoutWarning(true);
    }
}

void off()
{
    Serial.println("OFF");
    myServo.detach();
    digitalWrite(ledPin, LOW);
    noTone(buzzerPin);
}

void moveServoCheck(int angle)
```

```
{
    myServo.attach(13);
    myServo.write(angle);
    delay(25);
    distance = calculateDistance();
    sound = analogRead(soundPin);
    Serial.print("ON | Angle: ");
    Serial.print(angle);
    Serial.print(" | Distance: ");
    Serial.print(distance);
    Serial.print(" | Sound: ");
    Serial.println(sound);
    if ((distance <= 20 && distance > 0) || sound >=
1023)
        triggerWarning(false);
}

int calculateDistance()
{
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    return pulseIn(echoPin, HIGH) * 0.034 / 2;
}

void triggerWarning(bool timer)
{
    myServo.detach();
    if (timer)
        Serial.println("TIMEOUT WARNING");
    else
        Serial.println("WARNING");
    unsigned long lastLEDChange = 0;
    unsigned long lastBuzzerChange = 0;
    int currentFreq = MIN_FREQ;
    bool ledState = false;
    while (1)
    {
        unsigned long currentMillis = millis();
        if (currentMillis - lastLEDChange >=
LED_INTERVAL)
        {
            lastLEDChange = currentMillis;
            ledState = !ledState;
            digitalWrite(ledPin, ledState);
        }
        if (currentMillis - lastBuzzerChange >= 50)
        {
            lastBuzzerChange = currentMillis;
            tone(buzzerPin, currentFreq);
            if (timer)
                currentFreq += freqStep * 2;
            else
```

```
                currentFreq += freqStep;
            if (currentFreq >= MAX_FREQ || currentFreq <=
MIN_FREQ)
                freqStep = -freqStep;
        }
        if (digitalRead(buttonPin) == LOW)
        {
            if (timer)
                startTime = millis();
            break;
        }
    }
    digitalWrite(ledPin, LOW);
    noTone(buzzerPin);
}
```

Berikut beberapa penjelasan kode program diatas:

- `#include <Servo.h>`: Kode ini mengimpor pustaka Servo yang memungkinkan pengguna untuk mengontrol motor servo. Pustaka ini menyediakan fungsi dan metode untuk mengatur posisi servo berdasarkan sudut tertentu.
- `const int trigPin = 11;`: Pin untuk mengirim sinyal ultrasonik.
- `const int echoPin = 12;`: Pin untuk menerima sinyal ultrasonik yang dipantulkan.
- `const int ldrPin = 2;`: Pin untuk sensor LDR yang mengukur cahaya.
- `const int soundPin = A0;`: Pin untuk sensor suara.
- `const int ledPin = 3;`: Pin untuk LED yang berfungsi sebagai indikator.
- `const int buzzerPin = 4;`: Pin untuk buzzer yang memberikan peringatan suara.
- `const int buttonPin = 5;`: Pin untuk menonaktifkan alarm warning.
- `long duration;`: Variabel untuk menyimpan durasi waktu sinyal ultrasonik.
- `int distance;`: Variabel untuk menyimpan jarak yang dihitung oleh sensor ultrasonik.
- `Servo myServo;`: Objek untuk mengontrol motor servo, yang dapat dipindahkan ke sudut tertentu.

Pada kode program ini terdapat beberapa fungsi, yaitu:

- `void setup()`: Fungsi ini dieksekusi sekali saat program dimulai. Di dalamnya, pin-pin diatur sebagai input atau output sesuai kebutuhan. Komunikasi serial dimulai untuk memudahkan debugging, dan objek servo dihubungkan dengan pin tertentu.
- `void loop()`: Fungsi ini berulang kali dijalankan setelah fungsi `setup()`. Nilai dari sensor LDR dibaca. Jika nilai LDR adalah HIGH menandakan keadaan gelap dan sistem akan aktif. Jika false maka memanggil fungsi `off()`.
- `void checkTimer()`: Fungsi ini mengecek apakah timer ini sudah sampai timer yang dikonfigurasi oleh user. Jika true maka akan trigger warning mode timeout.
- `void off()` : Fungsi ini untuk mematikan sistem keamanan menandakan bahwa waktu pada siang hari.
- `void moveServoCheck(int angle)`: Fungsi ini menggerakkan servo ke sudut yang diberikan parameter angle dan menunggu sebentar agar servo mencapai posisi tersebut.

- `int calculateDistance()`: Fungsi ini menghitung jarak menggunakan sensor ultrasonik. Ia mengirimkan sinyal, menunggu hingga sinyal kembali, dan menghitung durasi waktu tersebut. Jarak dihitung dengan menggunakan rumus berdasarkan kecepatan suara dan hasilnya dikembalikan.
- `void triggerWarning()`: Fungsi ini dipanggil ketika sistem mendeteksi ancaman. LED dan buzzer dinyalakan selama 100 milidetik untuk memberikan sinyal peringatan. Mode warning akan keluar ketika tombol ditekan.

2.3 Kode Radar Processing

Kode program ini berfungsi sebagai antarmuka pengguna grafis (GUI) untuk menampilkan hasil dari radar ultrasonik.

Berikut kode program radar Processing.

radar.java

```
1   import processing.serial.*;
2   import java.awt.event.KeyEvent;
3   import java.io.IOException;
4
5   Serial myPort;
6   String angle = "";
7   String distance = "";
8   String data = "";
9   String noObject;
10  float pixsDistance;
11  int iAngle, iDistance;
12  int index1 = 0;
13  int index2 = 0;
14  PFont orcFont;
15
16  void setup() {
17      size(1200, 700);
18      smooth();
19      myPort = new Serial(this, "COM5", 9600);
20      myPort.bufferUntil('.');
21  }
22
23  void draw() {
24      fill(98, 245, 31);
25      noStroke();
26      fill(0, 4);
27      rect(0, 0, width, height - height * 0.065);
28      fill(98, 245, 31);
29      drawRadar();
30      drawLine();
31      drawObject();
32      drawText();
33  }
34
35  void serialEvent (Serial myPort) {
36      data = myPort.readStringUntil('.');
```

```
37      data = data.substring(0, data.length() - 1);
38      index1 = data.indexOf(",");
39      angle = data.substring(0, index1);
40      distance = data.substring(index1 + 1, data.length());
41      iAngle = int(angle);
42      iDistance = int(distance);
43  }
44
45  void drawRadar() {
46      pushMatrix();
47      translate(width / 2, height - height * 0.074);
48      noFill();
49      strokeWeight(2);
50      stroke(98, 245, 31);
51      arc(0, 0, (width - width * 0.0625), (width - width *
    0.0625), PI, TWO_PI);
52      arc(0, 0, (width - width * 0.27), (width - width *
    0.27), PI, TWO_PI);
53      arc(0, 0, (width - width * 0.479), (width - width *
    0.479), PI, TWO_PI);
54      arc(0, 0, (width - width * 0.687), (width - width *
    0.687), PI, TWO_PI);
55      line(-width / 2, 0, width / 2, 0);
56      line(0, 0, (-width / 2) * cos(radians(30)), (-width /
    2) * sin(radians(30)));
57      line(0, 0, (-width / 2) * cos(radians(60)), (-width /
    2) * sin(radians(60)));
58      line(0, 0, (-width / 2) * cos(radians(90)), (-width /
    2) * sin(radians(90)));
59      line(0, 0, (-width / 2) * cos(radians(120)), (-width
    / 2) * sin(radians(120)));
60      line(0, 0, (-width / 2) * cos(radians(150)), (-width
    / 2) * sin(radians(150)));
61      line((-width / 2) * cos(radians(30)), 0, width / 2,
    0);
62      popMatrix();
63  }
64
65  void drawObject() {
66      pushMatrix();
67      translate(width / 2, height - height * 0.074);
68      strokeWeight(9);
69      stroke(255, 10, 10);
70      pixsDistance = iDistance * ((height - height *
    0.1666) * 0.025);
71      if (iDistance < 40) {
72          line(pixsDistance * cos(radians(iAngle)),
    -pixsDistance * sin(radians(iAngle)),
73                  (width - width * 0.505) *
    cos(radians(iAngle)), -(width - width * 0.505) *
    sin(radians(iAngle)));
74      }
75      popMatrix();
76  }
77
```

```
78  void drawLine() {
79      pushMatrix();
80      strokeWeight(9);
81      stroke(30, 250, 60);
82      translate(width / 2, height - height * 0.074);
83      line(0, 0, (height - height * 0.12) *
    cos(radians(iAngle)), -(height - height * 0.12) *
    sin(radians(iAngle)));
84      popMatrix();
85  }
86
87  void drawText() {
88      pushMatrix();
89      if (iDistance > 40) {
90          noObject = "Out of Range";
91      } else {
92          noObject = "In Range";
93      }
94      fill(0, 0, 0);
95      noStroke();
96      rect(0, height - height * 0.0648, width, height);
97      fill(98, 245, 31);
98      textSize(25);
99      text("10cm", width - width * 0.3854, height - height
    * 0.0833);
100     text("20cm", width - width * 0.281, height - height *
    0.0833);
101     text("30cm", width - width * 0.177, height - height *
    0.0833);
102     text("40cm", width - width * 0.0729, height - height
    * 0.0833);
103     textSize(40);
104     text("Simple Circuits ", width - width * 0.875,
    height - height * 0.0277);
105     text("Angle: " + iAngle + " °", width - width * 0.48,
    height - height * 0.0277);
106     text("Dist.:    ", width - width * 0.26, height -
    height * 0.0277);
107     if (iDistance < 40) {
108         text("        " + iDistance + " cm", width -
    width * 0.225, height - height * 0.0277);
109     }
110     textSize(25);
111     fill(98, 245, 60);
112     translate((width - width * 0.4994) + width / 2 *
    cos(radians(30)),
113               (height - height * 0.0907) - width / 2 *
    sin(radians(30)));
114     rotate(-radians(-60));
115     text("30°", 0, 0);
116     resetMatrix();
117     translate((width - width * 0.503) + width / 2 *
    cos(radians(60)),
118               (height - height * 0.0888) - width / 2 *
    sin(radians(60)));
```

```
119     rotate(-radians(-30));
120     text("60°", 0, 0);
121     resetMatrix();
122     translate((width - width * 0.507) + width / 2 *
    cos(radians(90)),
123               (height - height * 0.0833) - width / 2 *
    sin(radians(90)));
124     rotate(radians(0));
125     text("90°", 0, 0);
126     resetMatrix();
127     translate(width - width * 0.513 + width / 2 *
    cos(radians(120)),
128               (height - height * 0.07129) - width / 2 *
    sin(radians(120)));
129     rotate(radians(-30));
130     text("120°", 0, 0);
131     resetMatrix();
132     translate((width - width * 0.5104) + width / 2 *
    cos(radians(150)),
133               (height - height * 0.0574) - width / 2 *
    sin(radians(150)));
134     rotate(radians(-60));
135     text("150°", 0, 0);
136     popMatrix();
137 }
```

2.4 Cara Kerja

Diagram alir adalah representasi visual dari langkah-langkah dalam suatu proses atau sistem. Menggunakan simbol-simbol standar untuk menunjukkan berbagai jenis tindakan dan urutan kerja, diagram alir memudahkan pemahaman dan analisis alur kerja secara sistematis.

Pada penjelasan cara kerja kami menggunakan diagram flowchart. Berikut diagram flowchart dari cara kerja program pada sistem keamanan ini.

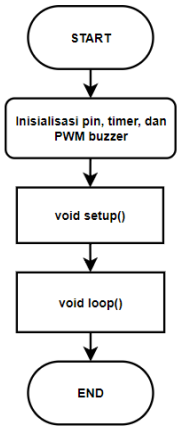
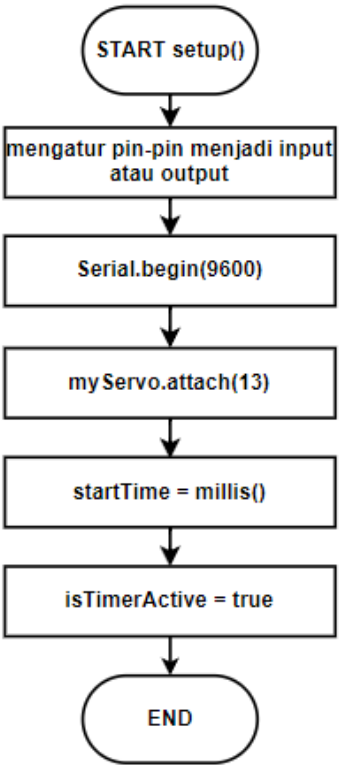
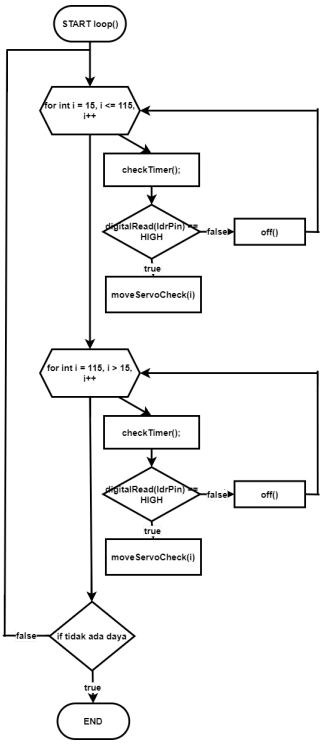


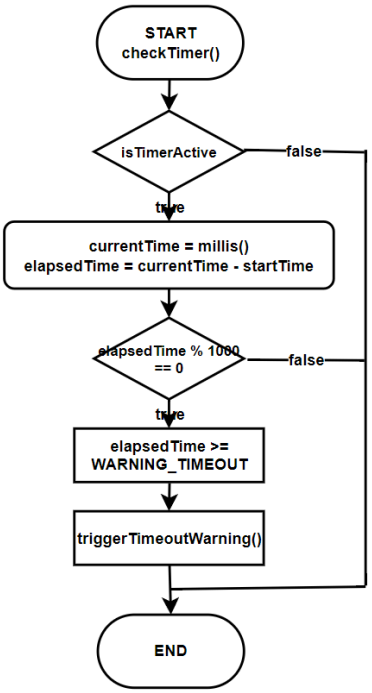
Diagram dimulai dengan proses inisialisasi variabel-variabel penting seperti `trigPin`, `echoPin`, `ldrPin`, `soundSensorPin`, `ledPin`, `buzzerPin`, `duration`, `distance`, dan objek `myServo`. Setelah inisialisasi, program masuk ke dalam fungsi `setup()`, di mana pin-pin pada Arduino diatur sebagai input atau output, komunikasi serial dimulai, dan servo disambungkan ke pin tertentu. Selanjutnya, program masuk ke dalam fungsi `loop()`, yang merupakan inti dari program dan dieksekusi berulang kali.



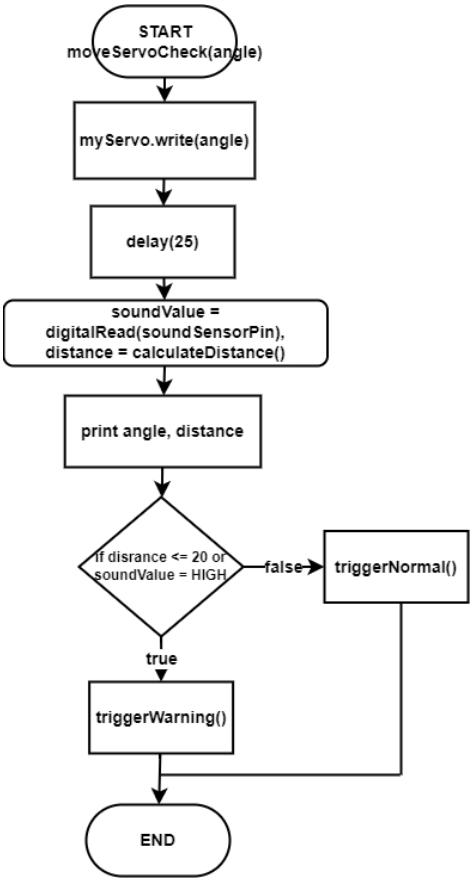
Fungsi `setup()` adalah bagian dari program Arduino yang dijalankan sekali pada awal eksekusi untuk mengatur konfigurasi awal perangkat keras yang terhubung. Dalam fungsi ini, beberapa pin pada Arduino dikonfigurasi menggunakan `pinMode()`. Pin `trigPin` diatur sebagai output untuk mengirim sinyal trigger ke sensor ultrasonik, sedangkan `echoPin` diatur sebagai input untuk menerima pantulan sinyal dari sensor ultrasonik. Pin `ldrPin` dan `soundSensorPin` diatur sebagai input untuk membaca nilai dari sensor LDR dan sensor suara. Pin `ledPin` dan `buzzerPin` diatur sebagai output untuk mengendalikan LED dan buzzer. Selain itu, `buttonPin` diatur sebagai input dengan pull-up resistor internal untuk mendeteksi penekanan tombol. Fungsi ini juga menginisialisasi komunikasi serial dengan kecepatan 9600 baud menggunakan `Serial.begin(9600)` untuk debugging dan menampilkan data, serta menyambungkan objek servo `myServo` ke pin digital 13 menggunakan `myServo.attach(13)`. Kemudian waktu timer dimulai.



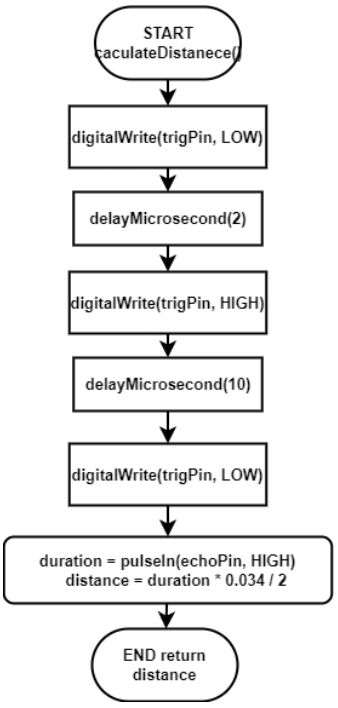
Fungsi `loop()` adalah inti dari program Arduino yang dijalankan berulang kali. Di dalam `loop()`, nilai dari sensor LDR dibaca menggunakan `digitalRead(ldrPin)`, dan nilai ini disimpan dalam variabel `ldrValue`. Jika nilai `ldrValue` adalah HIGH, menandakan kondisi gelap, maka dua loop `for` akan dijalankan untuk menggerakkan servo dari sudut 15 derajat ke 115 derajat, dan kembali dari 115 derajat ke 15 derajat. Pada setiap iterasi dalam kedua loop tersebut, fungsi `moveServoCheck(i)` dipanggil untuk memutar servo dan mengecek jarak serta nilai sensor suara. Jika nilai `ldrValue` adalah LOW maka menandakan terang dan memanggil fungsi `off()` untuk mematikan sistem keamanan.



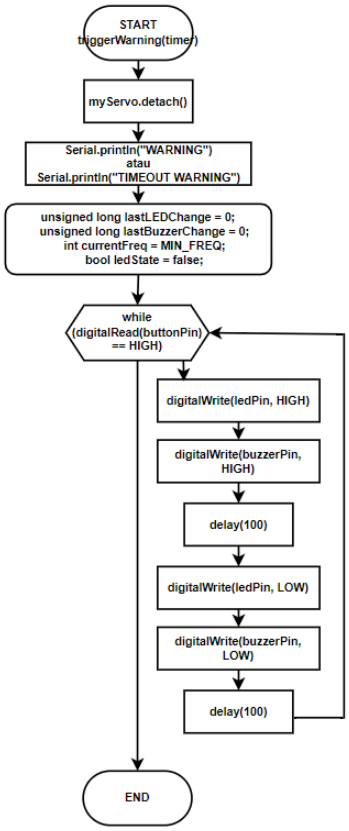
Fungsi `checkTimer()` dalam kode Arduino ini berfungsi untuk memantau apakah waktu yang telah berlalu sejak timer dimulai (variabel `startTime`) telah mencapai batas waktu peringatan (`WARNING_TIMEOUT`), yang diatur ke 300000 milidetik atau 5 menit. Pada setiap iterasi dari loop utama, fungsi ini dipanggil untuk memeriksa apakah timer aktif (`isTimerActive`). Jika timer aktif, fungsi ini menghitung waktu yang telah berlalu dengan mengurangi waktu saat ini (`millis()`) dengan waktu saat timer dimulai (`startTime`). Jika waktu yang telah berlalu melebihi atau sama dengan `WARNING_TIMEOUT`, maka fungsi `triggerTimeoutWarning(true)` akan dipanggil untuk mengaktifkan peringatan timeout, yang mungkin melibatkan mengeluarkan sinyal peringatan melalui LED dan buzzer serta menampilkan pesan peringatan pada serial monitor.



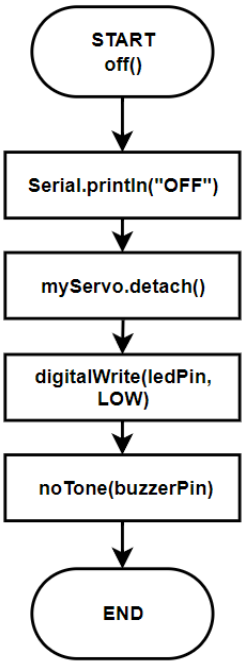
Fungsi `moveServoCheck(int angle)` digunakan untuk menggerakkan servo ke sudut tertentu dan memeriksa kondisi sensor pada setiap langkah pergerakan. Pertama, fungsi ini mengatur posisi servo ke sudut yang diberikan (`angle`) menggunakan `myServo.write(angle)`, kemudian menunggu selama 25 milidetik untuk memberi waktu bagi servo untuk mencapai posisi tersebut. Setelah itu, jarak dihitung dengan memanggil fungsi `calculateDistance()` dan disimpan dalam variabel `distance`, sedangkan nilai dari sensor suara dibaca dengan `digitalRead(soundSensorPin)` dan disimpan dalam variabel `soundValue`. Nilai sudut servo dan jarak dicetak ke komunikasi serial untuk tujuan debugging. Selanjutnya, kondisi jarak dan nilai sensor suara diperiksa: jika jarak kurang dari atau sama dengan 20 atau nilai sensor suara adalah `HIGH`, fungsi `triggerWarning()` dipanggil untuk mengaktifkan mode peringatan.



Fungsi `calculateDistance()` digunakan untuk mengukur jarak menggunakan sensor ultrasonik. Pertama, pin `trigPin` disetel ke `LOW` selama 2 mikrodetik untuk memastikan bahwa tidak ada sinyal tertinggal. Kemudian, pin `trigPin` disetel ke HIGH selama 10 mikrodetik untuk mengirimkan pulsa ultrasonik. Setelah itu, pin `trigPin` disetel kembali ke `LOW`. Fungsi `pulseIn(echoPin, HIGH)` kemudian digunakan untuk mengukur durasi waktu (dalam mikrodetik) yang dibutuhkan oleh pulsa untuk melakukan perjalanan ke objek terdekat dan kembali ke sensor. Durasi ini disimpan dalam variabel `duration`. Jarak dihitung dengan mengalikan durasi dengan 0.034 (kecepatan suara dalam cm/µs) dan membaginya dengan 2 (karena perjalanan bolak-balik), dan hasilnya disimpan dalam variabel `distance`. Nilai `distance` ini kemudian dikembalikan oleh fungsi.



Fungsi `triggerWarning()` bertujuan untuk menghentikan pergerakan servo dan memasuki mode peringatan, di mana LED dan buzzer berkedip secara cepat hingga tombol dilepas. Pertama, fungsi ini memutus kendali servo dengan memanggil `myServo.detach()`. Kemudian, fungsi ini memasuki loop `while` yang terus berulang selama tombol belum ditekan (nilai `buttonPin` tetap `HIGH`). Di dalam loop ini, LED dan buzzer diaktifkan (dinyalakan) selama 100 milidetik, kemudian dimatikan selama 100 milidetik, menciptakan efek berkedip. Ketika tombol ditekan (nilai `buttonPin` menjadi `LOW`), loop `while` berhenti. Setelah keluar dari loop, LED dan buzzer dimatikan dengan menulis nilai `LOW` pada `ledPin` dan `buzzerPin`, memastikan bahwa kedua perangkat tersebut tidak lagi berkedip. Akhirnya, kontrol servo dilanjutkan dengan memanggil `myServo.attach(13)`, yang menghubungkan kembali servo ke pin 13 untuk melanjutkan pergerakan normal.



Fungsi `off()` dalam kode Arduino ini berfungsi untuk menonaktifkan sistem keamanan dengan menghentikan semua operasi yang sedang berjalan dan mengembalikan perangkat ke keadaan standby. Ketika fungsi ini dipanggil, pertama-tama menampilkan pesan "OFF" pada serial monitor untuk indikasi status. Kemudian, fungsi ini melepaskan kontrol servo motor dengan memanggil `myServo.detach()`, yang menghentikan pergerakan servo. Selanjutnya, fungsi ini mematikan LED dengan mengatur pin `ledPin` ke kondisi LOW dan menghentikan suara buzzer dengan memanggil `noTone(buzzerPin)`. Dengan demikian, fungsi ini memastikan bahwa tidak ada sinyal peringatan yang aktif dan semua perangkat keras terkait berada dalam kondisi nonaktif.

2.5 Penerapan Mikrokontroler dan Mikroprosesor

Sensor dan aktuator merupakan materi dalam mata kuliah Mikrokontroler dan Mikroprosesor. Sensor adalah perangkat yang mendeteksi perubahan fisik di lingkungan, seperti cahaya, suhu, atau jarak, dan mengubahnya menjadi sinyal yang dapat diproses oleh mikrokontroler. Sementara itu, aktuator adalah komponen yang menerima sinyal dari mikrokontroler dan mengubahnya menjadi aksi fisik.

Berikut penerapan dari sensor dan aktuator pada program sistem ini:

1. **LDR (Light Dependent Resistor)**: Digunakan untuk mendeteksi tingkat cahaya di lingkungan sekitar. Nilai analog dibaca dari pin `A0`.
2. **Ultrasonic Sensor (HC-SR04)**: Menggunakan pin `trigPin` dan `echoPin` untuk menghitung jarak dengan menggunakan gelombang ultrasonik.
3. **Sound Sensor**: Dibaca dengan `analogRead(soundPin)`, digunakan untuk mendeteksi suara.
4. **Servo Motor**: Digunakan untuk menggerakkan mekanisme dari sudut 15 hingga 165 derajat. Aktuator ini dikendalikan oleh objek `myServo`.
5. **LED**: Digunakan untuk memberikan indikasi status (kedip dalam mode normal atau peringatan).
6. **Buzzer**: Mengeluarkan suara sebagai tanda peringatan ketika ada ancaman terdeteksi (misalnya suara atau jarak terlalu dekat).

Penerapan lain yang digunakan dalam sistem ini adalah timer/counter, PWM, ADC, dan UART. Timer/counter berfungsi untuk mengatur waktu atau menghitung peristiwa yang terjadi dalam interval tertentu. PWM (Pulse Width Modulation) adalah teknik mengatur daya keluaran dengan mengubah lebar pulsa pada frekuensi tetap. Sementara ADC (Analog-to-Digital Converter) digunakan untuk mengubah sinyal analog, seperti dari sensor, menjadi data digital yang dapat diproses oleh mikrokontroler. UART (Universal Asynchronous

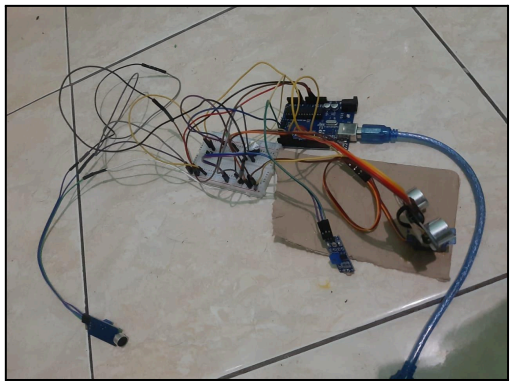
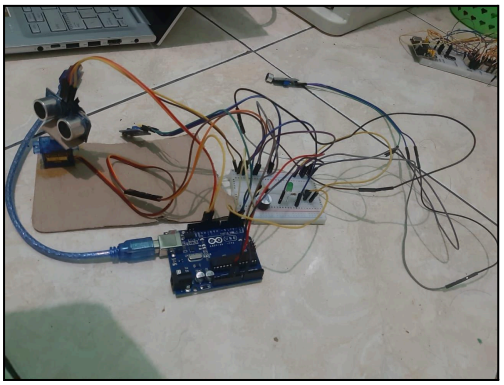
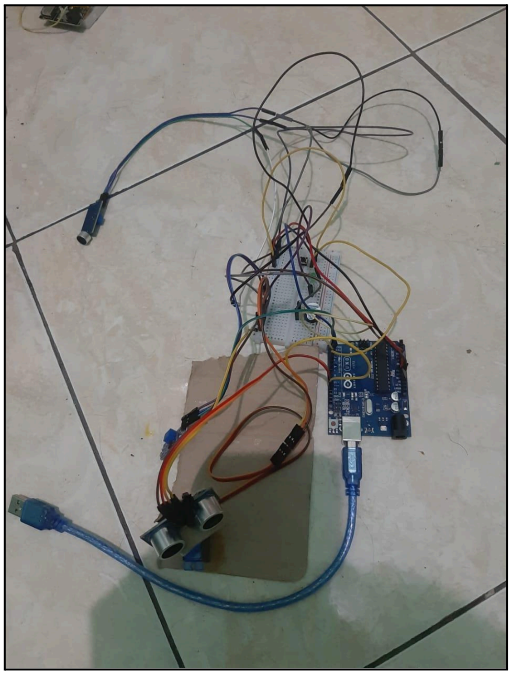
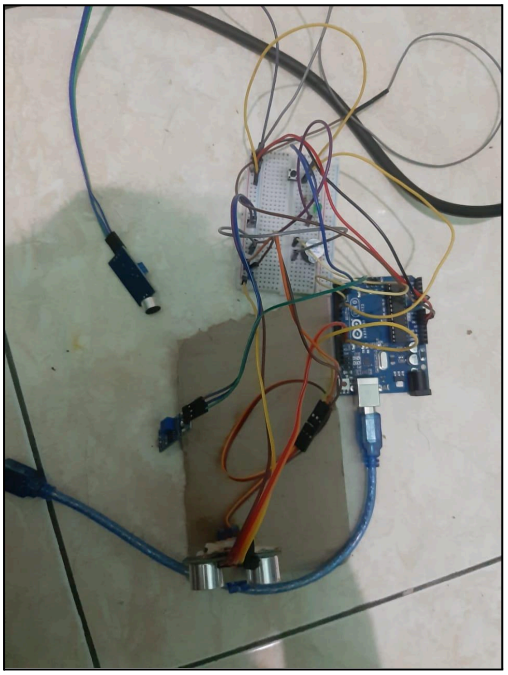
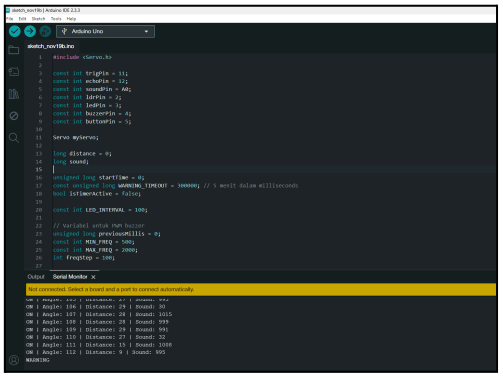
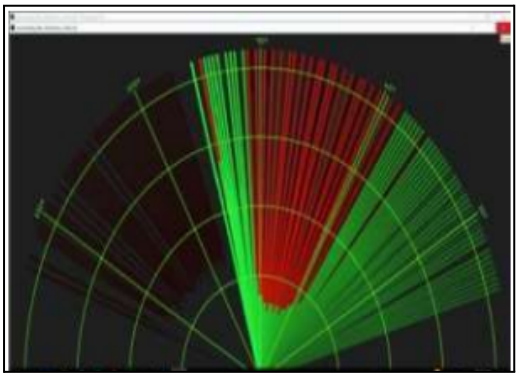
Receiver-Transmitter) memungkinkan mikrokontroler untuk berkomunikasi dengan perangkat lain seperti komputer melalui antarmuka serial

Berikut merupakan penerapan Timer/Counter, PWM dan ADC:

- **Timer/Counter**: Implementasi timer/counter pada kode menggunakan fungsi `millis()` untuk menghitung waktu yang telah berlalu sejak program dimulai, di mana sistem menginisialisasi `startTime` saat setup dan secara kontinyu memeriksa waktu yang berlalu melalui fungsi `checkTimer()`. Jika waktu yang berlalu mencapai `WARNING_TIMEOUT` (5 menit atau 300000 milliseconds), sistem akan memicu `triggerTimeoutWarning()` yang mengaktifkan LED berkedip setiap 100ms dan buzzer dengan PWM frekuensi yang berubah lebih cepat (500-2000Hz), kemudian sistem dapat direset dengan menekan tombol yang akan memperbarui `startTime` ke waktu saat ini, dan semua penghitungan waktu dilakukan secara non-blocking sehingga tidak mengganggu fungsi scanning servo dan deteksi sensor lainnya.
- **PWM:** Implementasi PWM (Pulse Width Modulation) pada kode hanya diterapkan pada buzzer menggunakan fungsi tone(), di mana frekuensi buzzer berubah secara bertahap dari `MIN_FREQ` (500Hz) hingga `MAX_FREQ` (2000Hz) dengan step perubahan (freqStep) sebesar 100Hz setiap 50ms. Ketika mencapai batas maksimum atau minimum, arah perubahan dibalik dengan mengubah nilai `freqStep` menjadi negatif, menciptakan efek suara yang naik-turun secara halus. Sementara untuk LED tidak menggunakan PWM melainkan menggunakan digital on/off biasa (`digitalWrite()`) dengan interval kedip 100ms yang diatur menggunakan `millis()` untuk timing non-blocking.
- **ADC**: Sound Sensor adalah sensor yang mengeluarkan nilai analog. Fungsi `analogRead(soundPin)` digunakan untuk membaca nilai dari Sound Sensor yang terhubung ke pin analog `A0`. Nilai yang dibaca oleh Sound Sensor kemudian digunakan untuk mengecek apakah terdapat suara bising yang menyebabkan warning mode aktif.
- **UART**: diterapkan melalui fungsi `Serial` bawaan Arduino, yang memungkinkan mikrokontroler untuk berkomunikasi dengan perangkat lain seperti komputer melalui antarmuka serial. Dalam fungsi `setup()`, `Serial.begin(9600)` menginisialisasi komunikasi serial dengan baud rate 9600 bps (bits per second). Selama eksekusi program, berbagai fungsi seperti `Serial.print()` dan `Serial.println()` digunakan untuk mengirim data ke komputer, yang dapat berupa informasi status, seperti "OFF", "ON | Angle: ", "Distance: ", "Sound: ", dan pesan peringatan lainnya. Data ini bisa dipantau di Serial Monitor pada Arduino IDE untuk debugging dan pemantauan sistem.

BAB III
HASIL

3.1 Dokumentasi Sistem

3.2 Hasil Percobaan

https://www.youtube.com/watch?v=4PF1Kgk9h3k

BAB IV
PENUTUP

4.1 Kesimpulan

Sistem keamanan cerdas berbasis sensor radar ultrasonik dan sensor suara ini merupakan solusi efektif untuk meningkatkan perlindungan di malam hari. Dengan mengintegrasikan sensor LDR untuk mendeteksi cahaya, sensor radar ultrasonik untuk mengukur jarak objek, dan sensor suara untuk mendeteksi kebisingan, sistem ini dapat memberikan respons otomatis terhadap potensi ancaman. Penggunaan LED dan buzzer sebagai indikator visual dan audio memberikan peringatan yang jelas kepada pengguna, sehingga dapat segera mengambil tindakan jika terjadi keadaan darurat. Kombinasi teknologi ini menjadikan sistem keamanan ini efisien dan mudah dioperasikan.

Selain itu, penerapan mikrokontroler dalam sistem ini, khususnya menggunakan Arduino, memungkinkan pengolahan data secara real-time dan kontrol terhadap perangkat keras. Kode program yang disusun memastikan bahwa setiap sensor bekerja dengan baik, dan peringatan diberikan saat kondisi berbahaya terdeteksi. Dengan demikian, sistem ini tidak hanya memberikan perlindungan yang lebih baik di malam hari, tetapi juga memanfaatkan teknologi sensor dan mikrokontroler untuk menciptakan sistem keamanan yang lebih responsif dan dapat diandalkan.