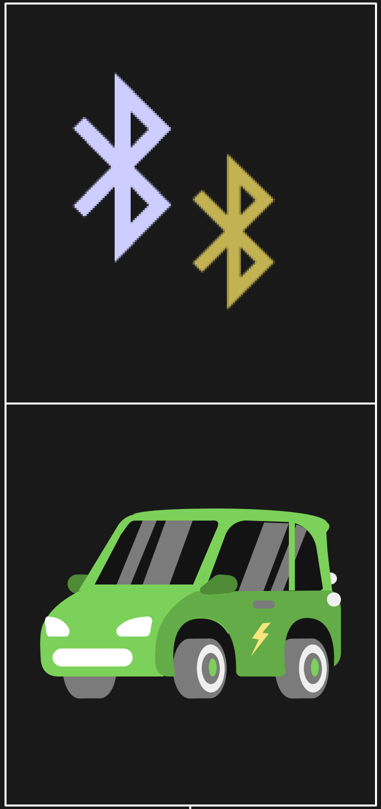
Arduino to Arduino Bluetooth Communication



29.11.24 TABLE NO-3 LAB-104

SHRIYA KANSAL 2024102075
ANUSHKA SAINI 2024102052

Table of contents

01.

Components Required

02.

Problem Statement

03.

Our Motivation

04.

Block Diagram

05.

Journey

06.

Code

07.

Photos/Videos

08.

References

Components Required

Arduino Uno (2)

Breadboard (2)

HC-05 module (2)

Jumper wires

LEDs

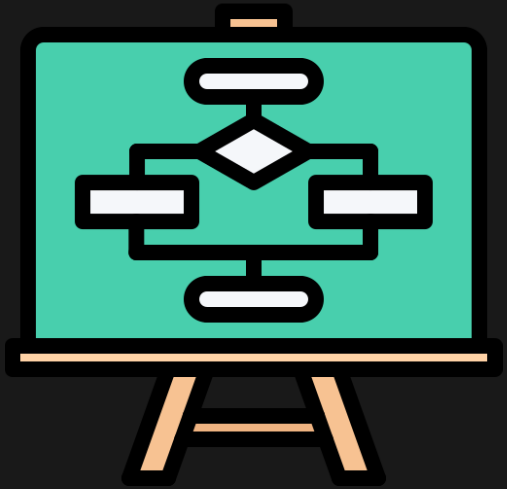
Buzzer

Problem Statement

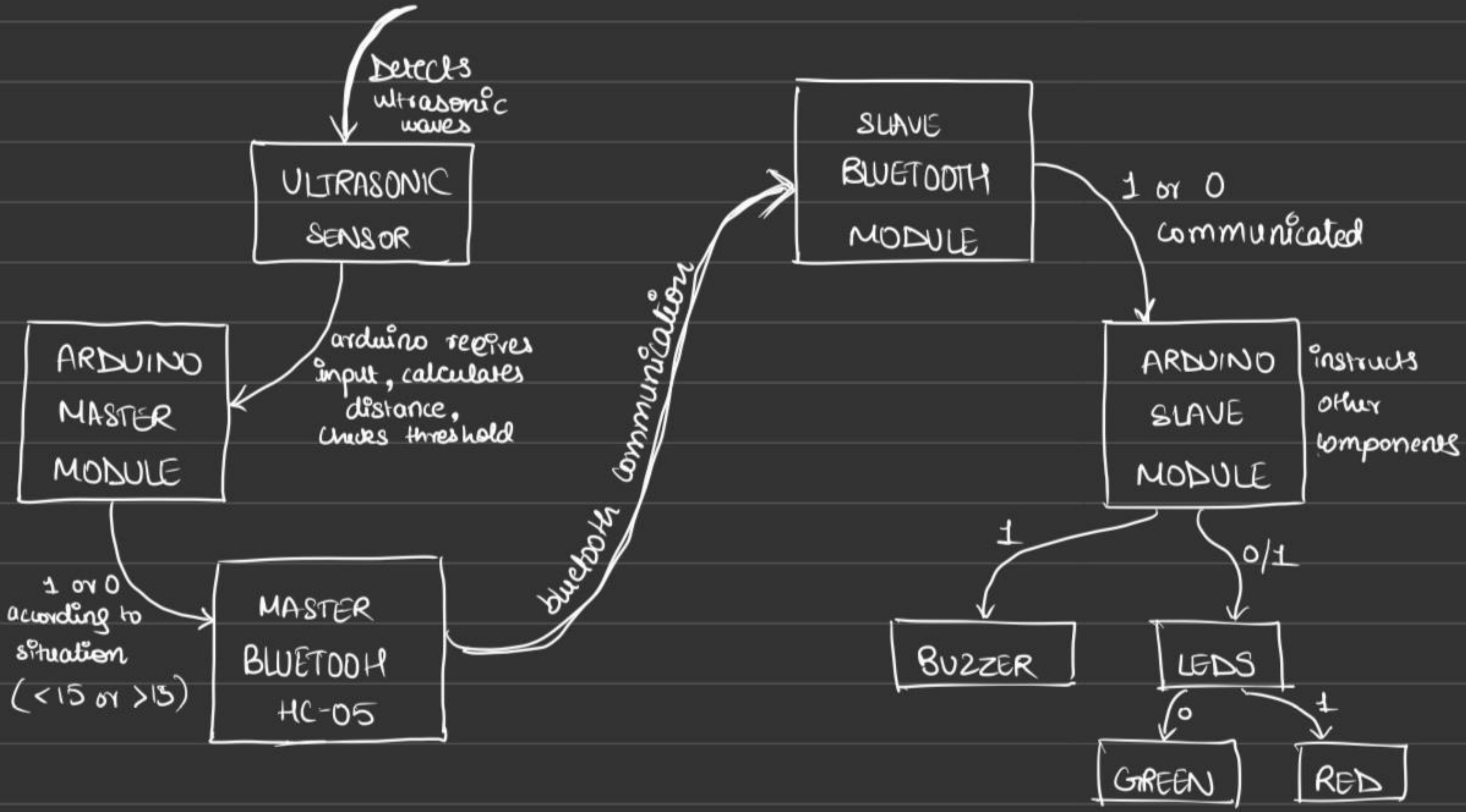
In safety-critical environments, there is a need for a cost-effective system to detect proximity and prevent collisions. Our solution uses an ultrasonic sensor and Bluetooth communication to monitor distance, classify areas as "Danger Zone" or "Safe Zone," and trigger an audible warning to ensure safety.

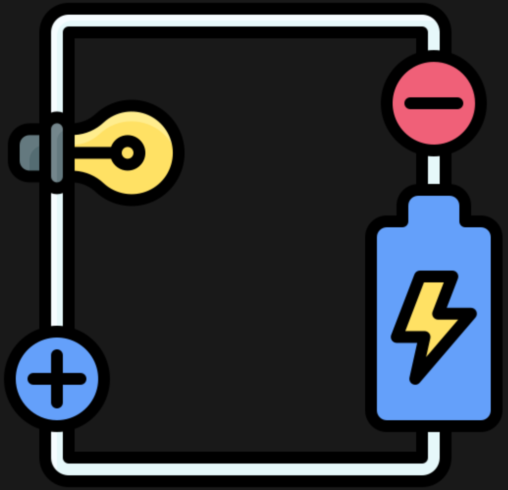
Motivation

This project was driven by the need to improve safety in critical environments such as industrial areas and parking systems. By leveraging affordable and accessible components like ultrasonic sensors and Bluetooth modules, the goal was to design a reliable system for detecting potential hazards and providing timely alerts to prevent accidents.

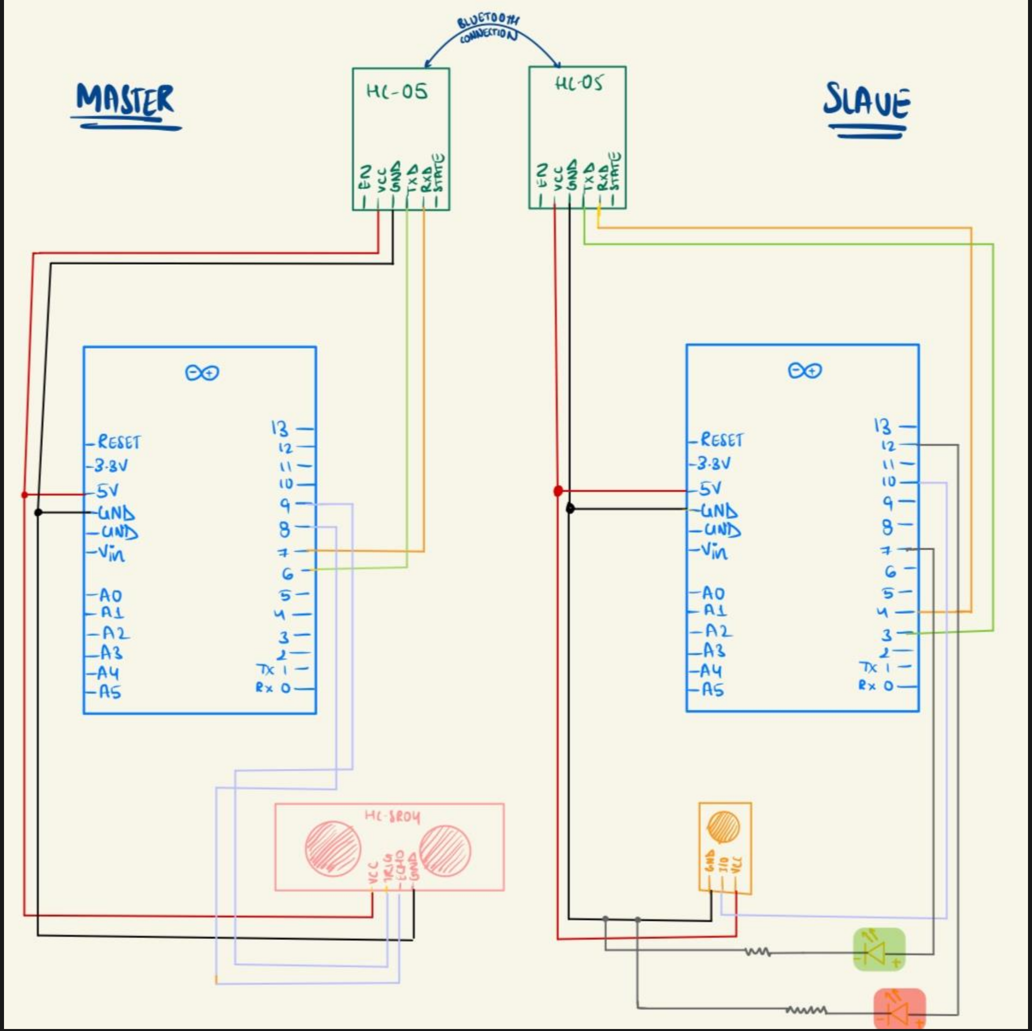


BLOCK DIAGRAM


Detects ultrasonic waves
ULTRASONIC SENSOR
arduino receives input, calculates distance, checks threshold
ARDUINO MASTER MODULE
1 or 0 according to situation (<15 or >15)
MASTER BLUETOOTH HC-05
bluetooth communication
SLAVE BLUETOOTH MODULE
1 or 0 communicated
ARDUINO SLAVE MODULE
instructs other components
1
0/1
BUZZER
LEDS
0
1
GREEN
RED



CIRCUIT DIAGRAM

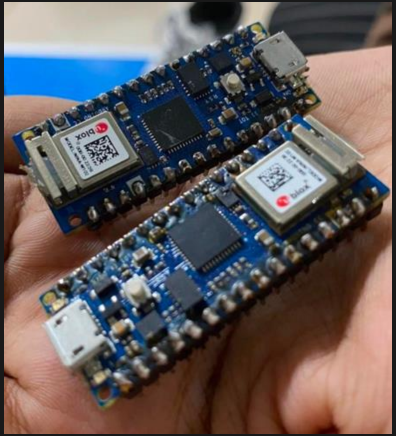

BLUETOOTH CONNECTION
MASTER
SLAVE
HC-05
EN
VCC
GND
TXD
RXD
STATE
HC-05
EN
VCC
GND
TXD
RXD
STATE
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5
13
12
11
10
9
8
7
6
5
4
3
2
TX 1
Rx 0
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5
13
12
11
10
9
8
7
6
5
4
3
2
TX 1
Rx 0
HC-SR04
VCC
TRIG
ECHO
GND
GND
I/O
VCC

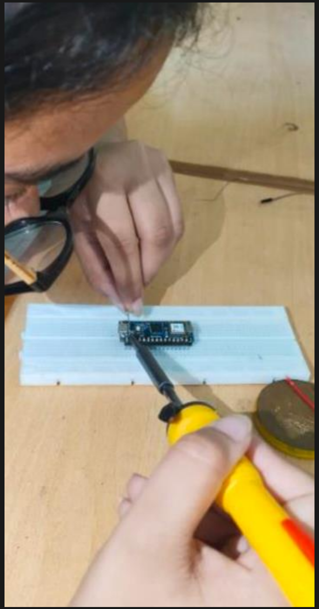
OUR JOURNEY

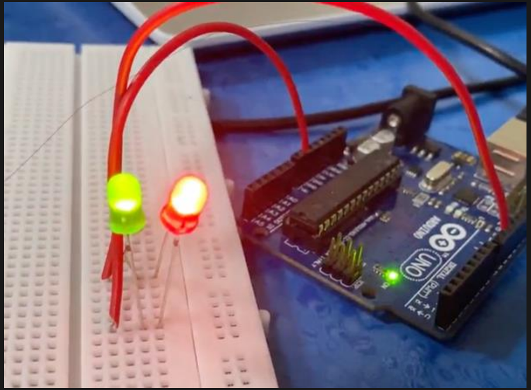
Checking of components + soldering

In this step we checked the conditions of our components (LEDs, Bluetooth modules, buzzer)

We also soldered the Arduino nanos together with pins as they were part of our initial project

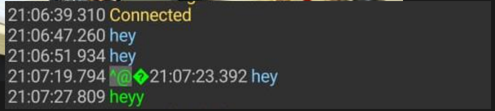






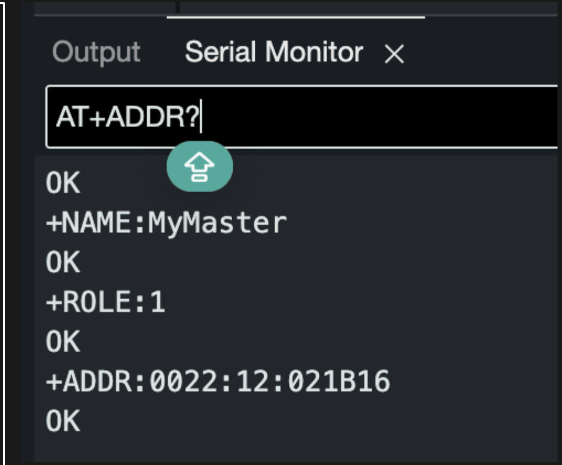
Link to videos

Setting Bluetooth modules’ roles

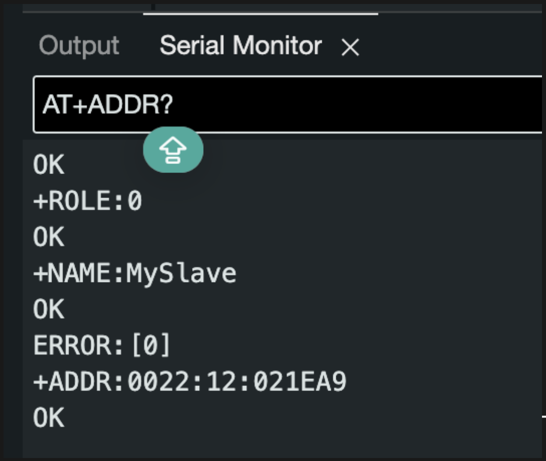


```
21:06:39.310 Connected
21:06:47.260 hey
21:06:51.934 hey
21:07:19.794 ^@�21:07:23.392 hey
21:07:27.809 heyy
```

In AT mode, the names of the HC-05 modules were changed to MyMaster and MySlave with their roles being assigned 1 and 0 respectively. We also binded the two modules together using the AT+BIND command.



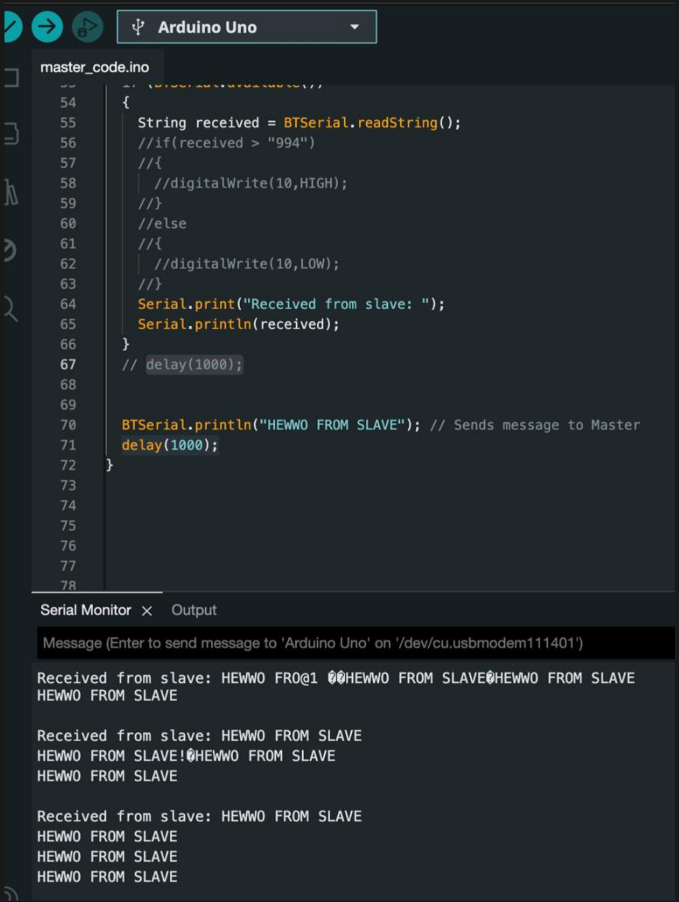
```
AT+ADDR?
OK
+NAME:MyMaster
OK
+ROLE:1
OK
+ADDR:0022:12:021B16
OK
```



```
AT+ADDR?
OK
+ROLE:0
OK
+NAME:MySlave
OK
ERROR:[0]
+ADDR:0022:12:021EA9
OK
```

Testing the communication between the modules

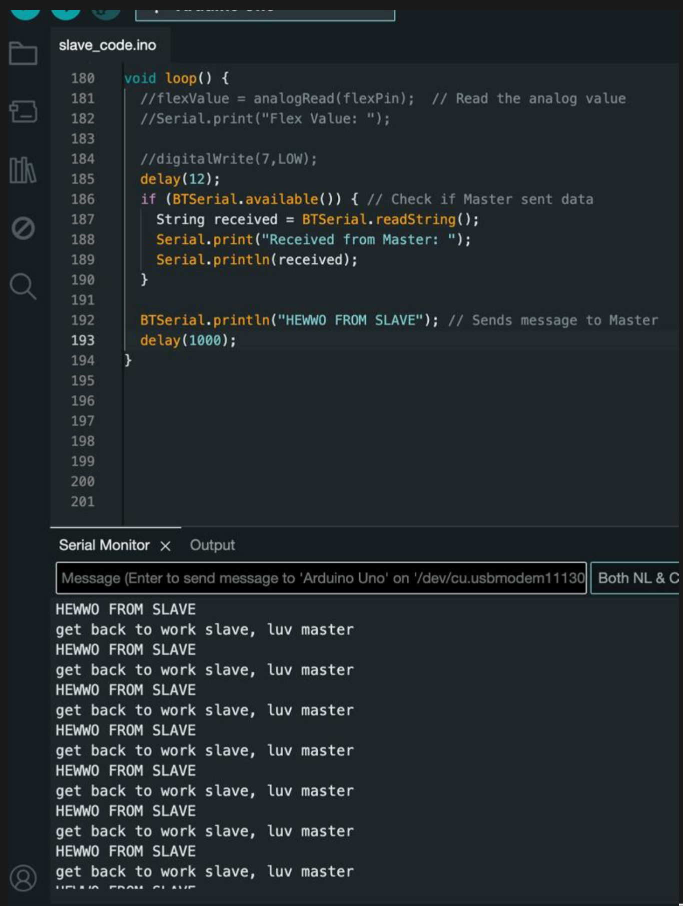
We made a simple circuit with only our Arduinos attached to the Bluetooth modules (with them paired) and checked if they were communicating properly using the serial monitor



```
  {
    String received = BTSerial.readString();
    //if(received > "994")
    //{
      //digitalWrite(10,HIGH);
    //}
    //else
    //{
      //digitalWrite(10,LOW);
    //}
    Serial.print("Received from slave: ");
    Serial.println(received);
  }
  // delay(1000);


  BTSerial.println("HEWWO FROM SLAVE"); // Sends message to Master
  delay(1000);
}
```

Message (Enter to send message to 'Arduino Uno' on '/dev/cu.usbmodem111401')

```
Received from slave: HEWWO FRO@1 ��HEWWO FROM SLAVE�HEWWO FROM SLAVE
HEWWO FROM SLAVE

Received from slave: HEWWO FROM SLAVE
HEWWO FROM SLAVE!�HEWWO FROM SLAVE
HEWWO FROM SLAVE

Received from slave: HEWWO FROM SLAVE
HEWWO FROM SLAVE
HEWWO FROM SLAVE
HEWWO FROM SLAVE
```



```
void loop() {
  //flexValue = analogRead(flexPin);  // Read the analog value
  //Serial.print("Flex Value: ");

  //digitalWrite(7,LOW);
  delay(12);
  if (BTSerial.available()) { // Check if Master sent data
    String received = BTSerial.readString();
    Serial.print("Received from Master: ");
    Serial.println(received);
  }

  BTSerial.println("HEWWO FROM SLAVE"); // Sends message to Master
  delay(1000);
}
```

Message (Enter to send message to 'Arduino Uno' on '/dev/cu.usbmodem11130

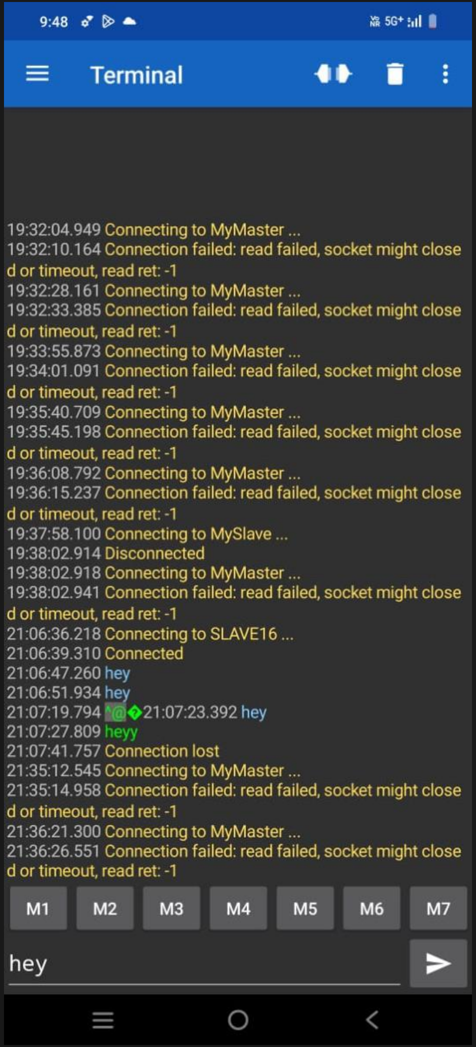
```
HEWWO FROM SLAVE
get back to work slave, luv master
HEWWO FROM SLAVE
get back to work slave, luv master
HEWWO FROM SLAVE
get back to work slave, luv master
HEWWO FROM SLAVE
get back to work slave, luv master
HEWWO FROM SLAVE
get back to work slave, luv master
HEWWO FROM SLAVE
get back to work slave, luv master
HEWWO FROM SLAVE
get back to work slave, luv master
```

Testing the communication between the modules

We used an app called Serial Bluetooth Terminal to check for the communication issues that we were experiencing

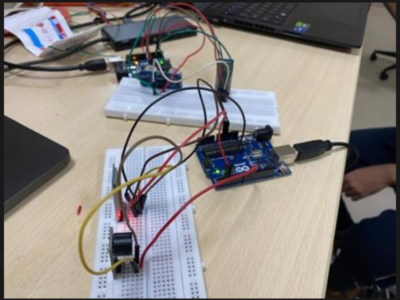
This is where we realised that the master was not communicating with us properly but the slave was. This was very confusing to us but as we looked more into it we realised the problem lay within how we were utilising the master and the slave module.

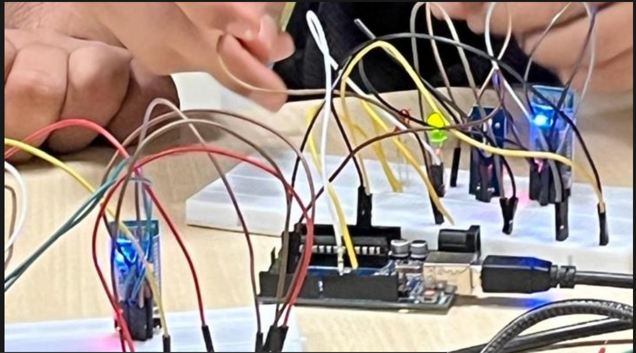
Serial Bluetooth Terminal

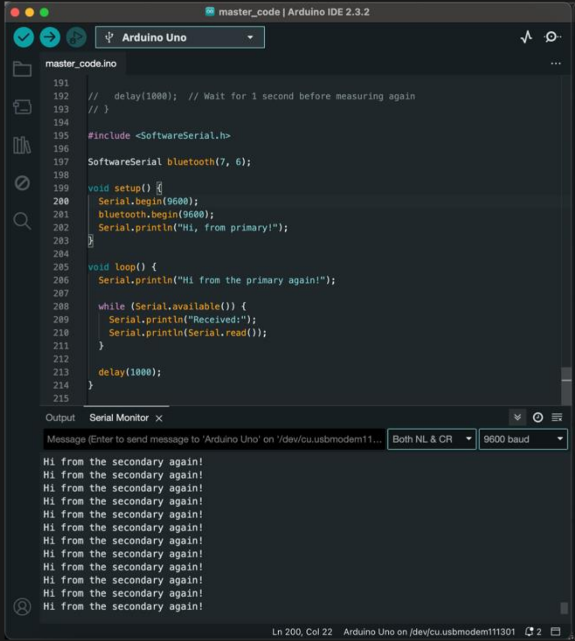


```
19:32:04.949 Connecting to MyMaster ...
19:32:10.164 Connection failed: read failed, socket might close
d or timeout, read ret: -1
19:32:28.161 Connecting to MyMaster ...
19:32:33.385 Connection failed: read failed, socket might close
d or timeout, read ret: -1
19:33:55.873 Connecting to MyMaster ...
19:34:01.091 Connection failed: read failed, socket might close
d or timeout, read ret: -1
19:35:40.709 Connecting to MyMaster ...
19:35:45.198 Connection failed: read failed, socket might close
d or timeout, read ret: -1
19:36:08.792 Connecting to MyMaster ...
19:36:15.237 Connection failed: read failed, socket might close
d or timeout, read ret: -1
19:37:58.100 Connecting to MySlave ...
19:38:02.914 Disconnected
19:38:02.918 Connecting to MyMaster ...
19:38:02.941 Connection failed: read failed, socket might close
d or timeout, read ret: -1
21:06:36.218 Connecting to SLAVE16 ...
21:06:39.310 Connected
21:06:47.260 hey
21:06:51.934 hey
21:07:19.794 ^@�21:07:23.392 hey
21:07:27.809 heyy
21:07:41.757 Connection lost
21:35:12.545 Connecting to MyMaster ...
21:35:14.958 Connection failed: read failed, socket might close
d or timeout, read ret: -1
21:36:21.300 Connecting to MyMaster ...
21:36:26.551 Connection failed: read failed, socket might close
d or timeout, read ret: -1
```

Serial Monitor Issue identified + building the final code







master_code | Arduino IDE 2.3.2

Arduino Uno
master_code.ino

```
//   delay(1000);  // Wait for 1 second before measuring again
// }

#include <SoftwareSerial.h>

SoftwareSerial bluetooth(7, 6);

void setup() {
  Serial.begin(9600);
  bluetooth.begin(9600);
  Serial.println("Hi, from primary!");
}

void loop() {
  Serial.println("Hi from the primary again!");

  while (Serial.available()) {
    Serial.println("Received:");
    Serial.println(Serial.read());
  }

  delay(1000);
}
```

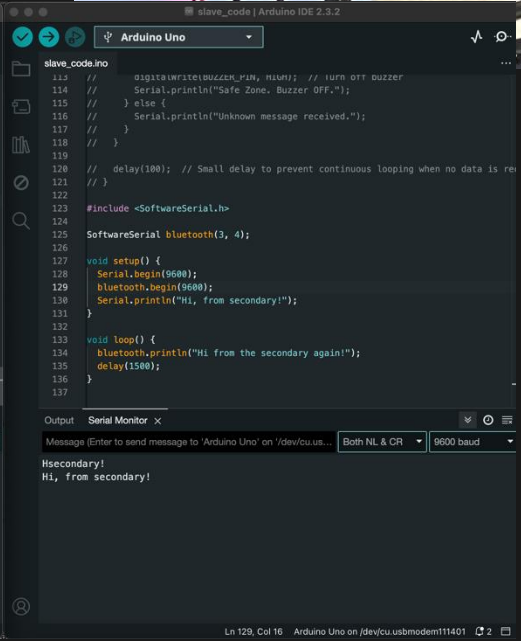
Output
Serial Monitor
Message (Enter to send message to 'Arduino Uno' on '/dev/cu.usbmodem11...

Both NL & CR

9600 baud

```
Hi from the secondary again!
Hi from the secondary again!
Hi from the secondary again!
Hi from the secondary again!
Hi from the secondary again!
Hi from the secondary again!
Hi from the secondary again!
Hi from the secondary again!
Hi from the secondary again!
Hi from the secondary again!
Hi from the secondary again!
Hi from the secondary again!
```

Ln 200, Col 22   Arduino Uno on /dev/cu.usbmodem111301   2



slave_code | Arduino IDE 2.3.2

```
//        digitalWrite(BUZZER_PIN, HIGH);  // Turn off buzzer
//        Serial.println("Safe Zone. Buzzer OFF.");
//      } else {
//        Serial.println("Unknown message received.");
//      }
//    }

//   delay(100);  // Small delay to prevent continuous looping when no data is rec
// }

#include <SoftwareSerial.h>

SoftwareSerial bluetooth(3, 4);

void setup() {
  Serial.begin(9600);
  bluetooth.begin(9600);
  Serial.println("Hi, from secondary!");
}

void loop() {
  bluetooth.println("Hi from the secondary again!");
  delay(1500);
}
```

Output
Serial Monitor
Message (Enter to send message to 'Arduino Uno' on '/dev/cu.us...

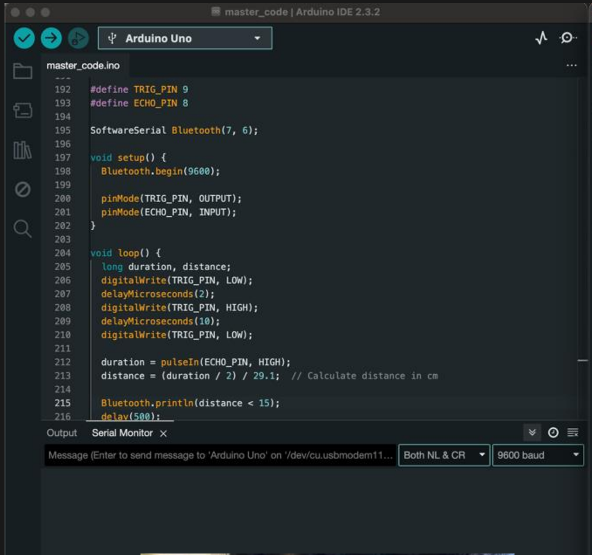
Both NL & CR

9600 baud

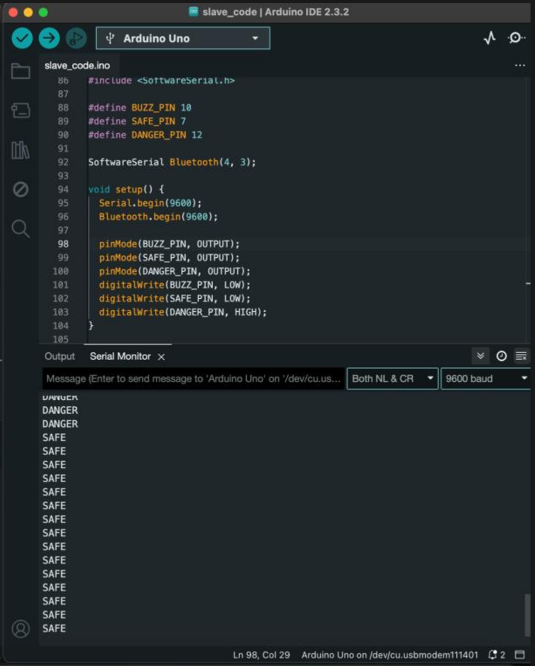
```
Hsecondary!
Hi, from secondary!
```

Ln 129, Col 16   Arduino Uno on /dev/cu.usbmodem111401   2

Final Execution of the Project


master_code | Arduino IDE 2.3.2
Arduino Uno
master_code.ino
Output
Serial Monitor
Message (Enter to send message to 'Arduino Uno' on '/dev/cu.usbmodem11...
Both NL & CR
9600 baud

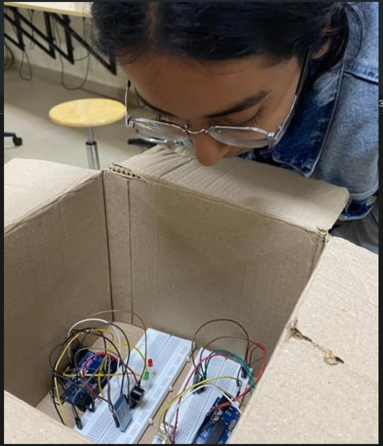
```
#define TRIG_PIN 9
#define ECHO_PIN 8

SoftwareSerial Bluetooth(7, 6);

void setup() {
  Bluetooth.begin(9600);

  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
}

void loop() {
  long duration, distance;
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  duration = pulseIn(ECHO_PIN, HIGH);
  distance = (duration / 2) / 29.1;  // Calculate distance in cm

  Bluetooth.println(distance < 15);
  delay(500);
```


slave_code | Arduino IDE 2.3.2
Arduino Uno
slave_code.ino
Output
Serial Monitor
Message (Enter to send message to 'Arduino Uno' on '/dev/cu.us...
Both NL & CR
9600 baud
Ln 98, Col 29
Arduino Uno on /dev/cu.usbmodem111401
2

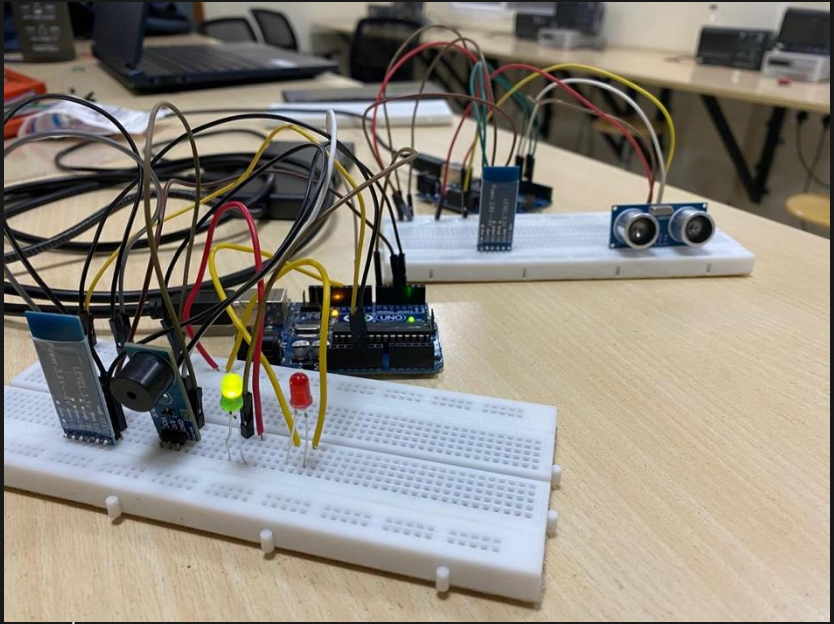
```
#include <SoftwareSerial.h>

#define BUZZ_PIN 10
#define SAFE_PIN 7
#define DANGER_PIN 12

SoftwareSerial Bluetooth(4, 3);

void setup() {
  Serial.begin(9600);
  Bluetooth.begin(9600);

  pinMode(BUZZ_PIN, OUTPUT);
  pinMode(SAFE_PIN, OUTPUT);
  pinMode(DANGER_PIN, OUTPUT);
  digitalWrite(BUZZ_PIN, LOW);
  digitalWrite(SAFE_PIN, LOW);
  digitalWrite(DANGER_PIN, HIGH);
}

```

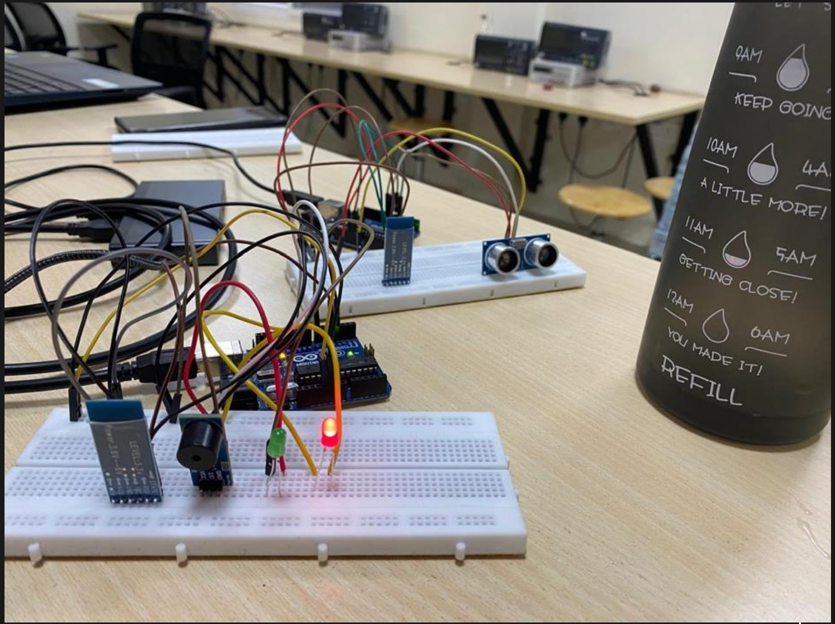
```
DANGER
DANGER
DANGER
SAFE
SAFE
SAFE
SAFE
SAFE
SAFE
SAFE
SAFE
SAFE
SAFE
SAFE
SAFE
SAFE
SAFE
SAFE
```

After building up the circuit bit by bit ensuring each component worked and testing it regularly, our project gave the desired output



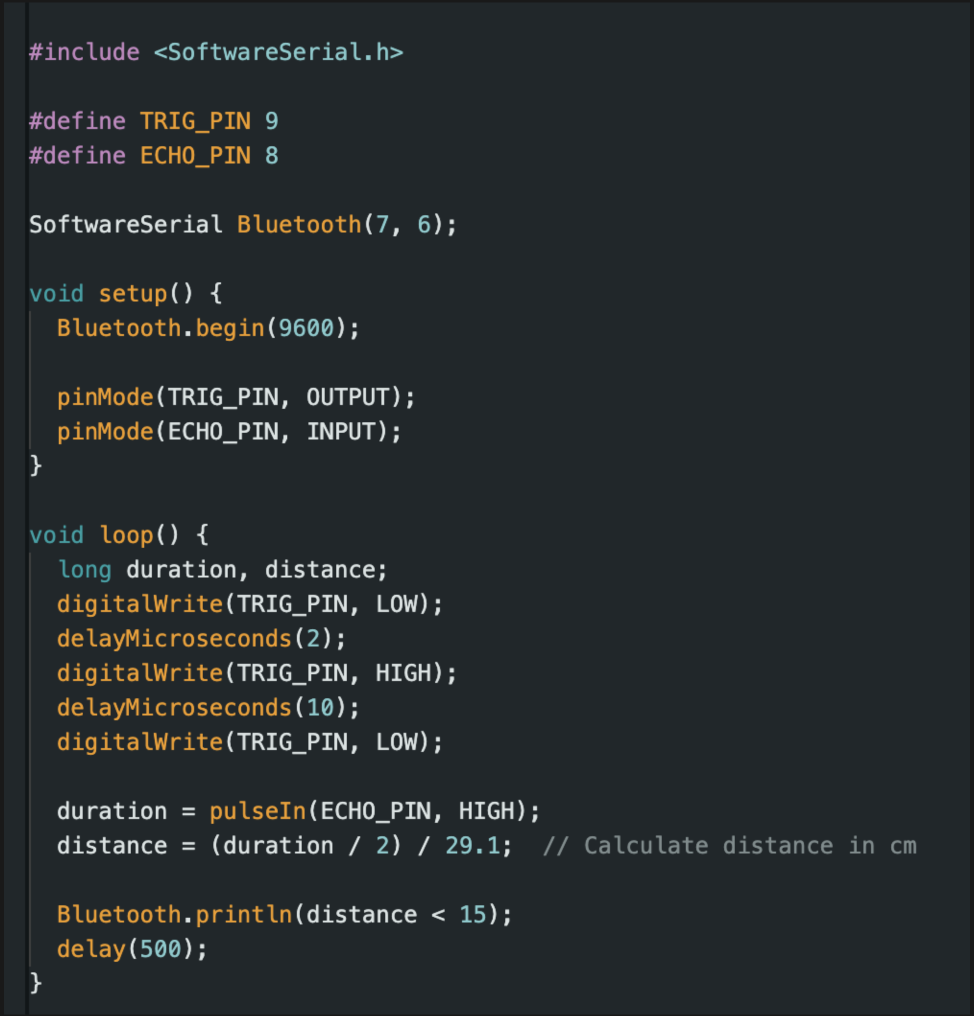
Execution of project


UNO


9AM
KEEP GOING
10AM
A LITTLE MORE!
4A
11AM
GETTING CLOSE!
5AM
12AM
YOU MADE IT!
6AM
REFILL

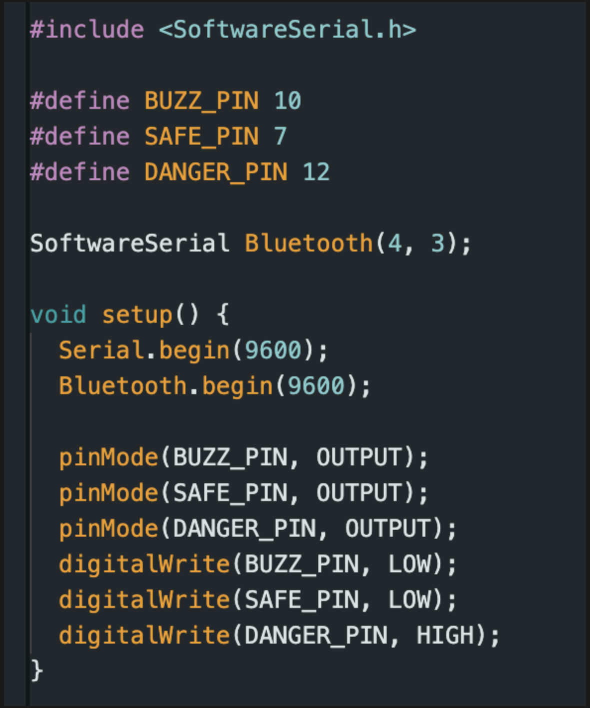
CODES

MASTER MODULE CODE

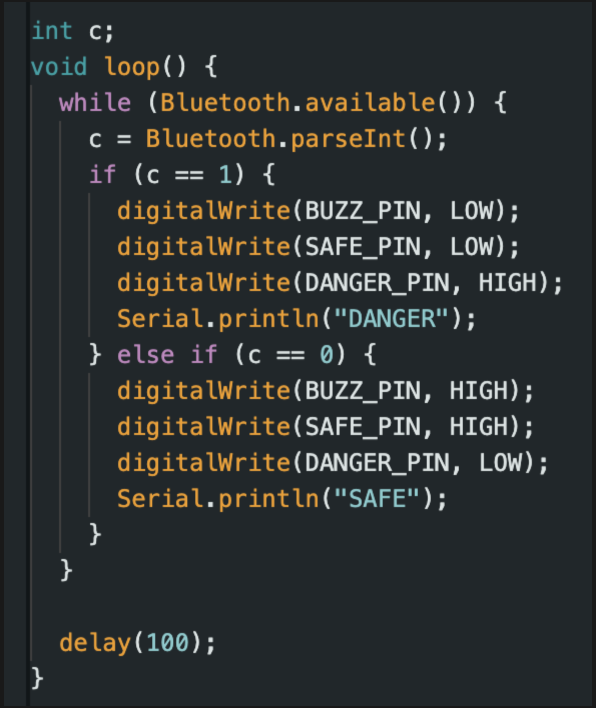


```
#include <SoftwareSerial.h>

#define TRIG_PIN 9
#define ECHO_PIN 8

SoftwareSerial Bluetooth(7, 6);

void setup() {
  Bluetooth.begin(9600);

  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
}

void loop() {
  long duration, distance;
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  duration = pulseIn(ECHO_PIN, HIGH);
  distance = (duration / 2) / 29.1;  // Calculate distance in cm

  Bluetooth.println(distance < 15);
  delay(500);
}
```

SLAVE MODULE CODE



```
#include <SoftwareSerial.h>

#define BUZZ_PIN 10
#define SAFE_PIN 7
#define DANGER_PIN 12

SoftwareSerial Bluetooth(4, 3);

void setup() {
  Serial.begin(9600);
  Bluetooth.begin(9600);

  pinMode(BUZZ_PIN, OUTPUT);
  pinMode(SAFE_PIN, OUTPUT);
  pinMode(DANGER_PIN, OUTPUT);
  digitalWrite(BUZZ_PIN, LOW);
  digitalWrite(SAFE_PIN, LOW);
  digitalWrite(DANGER_PIN, HIGH);
}
```



```
int c;
void loop() {
  while (Bluetooth.available()) {
    c = Bluetooth.parseInt();
    if (c == 1) {
      digitalWrite(BUZZ_PIN, LOW);
      digitalWrite(SAFE_PIN, LOW);
      digitalWrite(DANGER_PIN, HIGH);
      Serial.println("DANGER");
    } else if (c == 0) {
      digitalWrite(BUZZ_PIN, HIGH);
      digitalWrite(SAFE_PIN, HIGH);
      digitalWrite(DANGER_PIN, LOW);
      Serial.println("SAFE");
    }
  }

  delay(100);
}
```

REFERENCES

https://www.instructables.com/Arduino-Two-Way-.Communication-Via-Bluetooth-HC-05/

https://store.arduino.cc/products/Arduino-nano?srsltid=AfmBOoqMXS-OT83Ite00M5UoF72z1QEjqCbHkS9IXC7pg9WwZ4XywnaX

https://www.youtube.com/watch?v=I2qFXSe0W3w

https://stackoverflow.com/questions/18376468/arduino-using-serial-and-software-serial-with-bluetooth-module