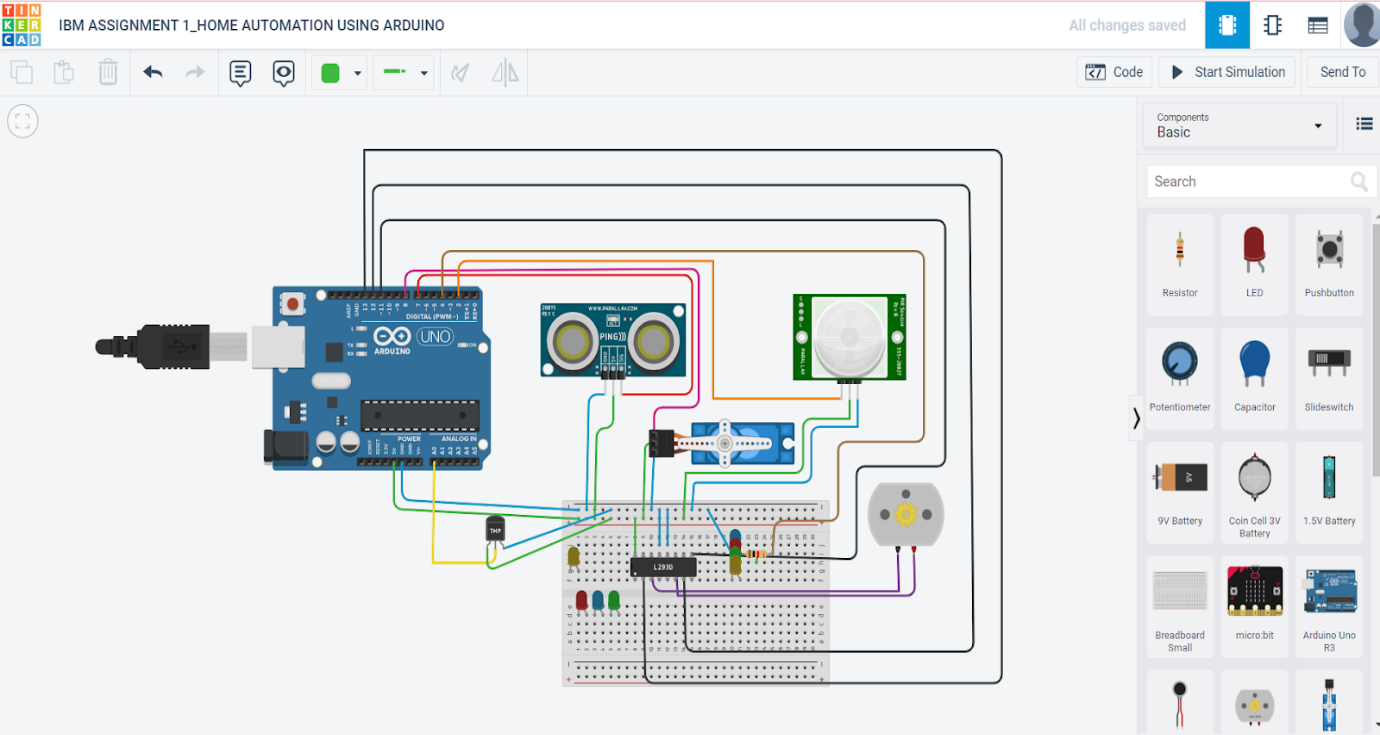
**ASSIGNMENT 1**

**SMART HOME IN TINKERCAD**

**TEAM MMEMBER-1:S Banumithra**

**REGISTER NUMBER:720319104302**

**SCHEMATIC:-**



CODE:

#include<Servo.h>

const int pingPin = 7;

int servoPin = 8;

Servo servo1;

void setup() {

  // initialize serial communication:

  Serial.begin(9600);

  servo1.attach(servoPin);

  pinMode(2,INPUT);

  pinMode(4,OUTPUT);

  pinMode(11,OUTPUT);

  pinMode(12,OUTPUT);

  pinMode(13,OUTPUT);

  pinMode(A0,INPUT);

  digitalWrite(2,LOW);

  digitalWrite(11,HIGH);

}

void loop() {

  long duration, inches, cm;

  pinMode(pingPin, OUTPUT);

  digitalWrite(pingPin, LOW);

  delayMicroseconds(2);

  digitalWrite(pingPin, HIGH);

  delayMicroseconds(5);

  digitalWrite(pingPin, LOW);

  // The same pin is used to read the signal from the PING))): a HIGH pulse

  // whose duration is the time (in microseconds) from the sending of the ping

  // to the reception of its echo off of an object.

  pinMode(pingPin, INPUT);

  duration = pulseIn(pingPin, HIGH);

  // convert the time into a distance

  inches = microsecondsToInches(duration);

  cm = microsecondsToCentimeters(duration);

  //Serial.print(inches);

  //Serial.print("in, ");

  //Serial.print(cm);

  //Serial.print("cm");

  //Serial.println();

  //delay(100);

  servo1.write(0);

  if(cm < 40)

  {

    servo1.write(90);

    delay(2000);

  }

  else

  {

    servo1.write(0);

  }

  // PIR with LED starts

  int pir = digitalRead(2);

  if(pir == HIGH)

  {

    digitalWrite(4,HIGH);

    delay(1000);

  }

  else if(pir == LOW)

  {

    digitalWrite(4,LOW);

  }

  //temp with fan

  float value=analogRead(A0);

  float temperature=value\*0.48;

  Serial.println("temperature");

  Serial.println(temperature);

  if(temperature > 20)

  {

    digitalWrite(12,HIGH);

    digitalWrite(13,LOW);

  }

  else

  {

    digitalWrite(12,LOW);

    digitalWrite(13,LOW);

  }

}

long microsecondsToInches(long microseconds) {

  return microseconds / 74 / 2;

}

long microsecondsToCentimeters(long microseconds) {

  return microseconds / 29 / 2;

}