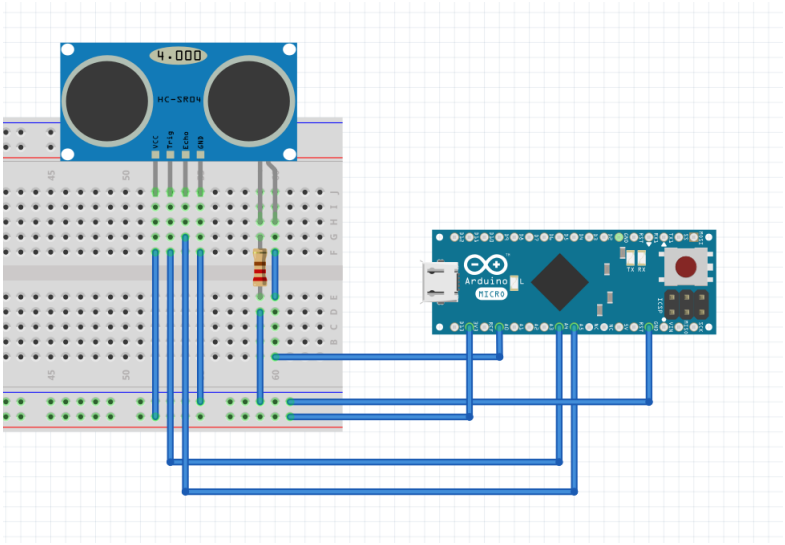
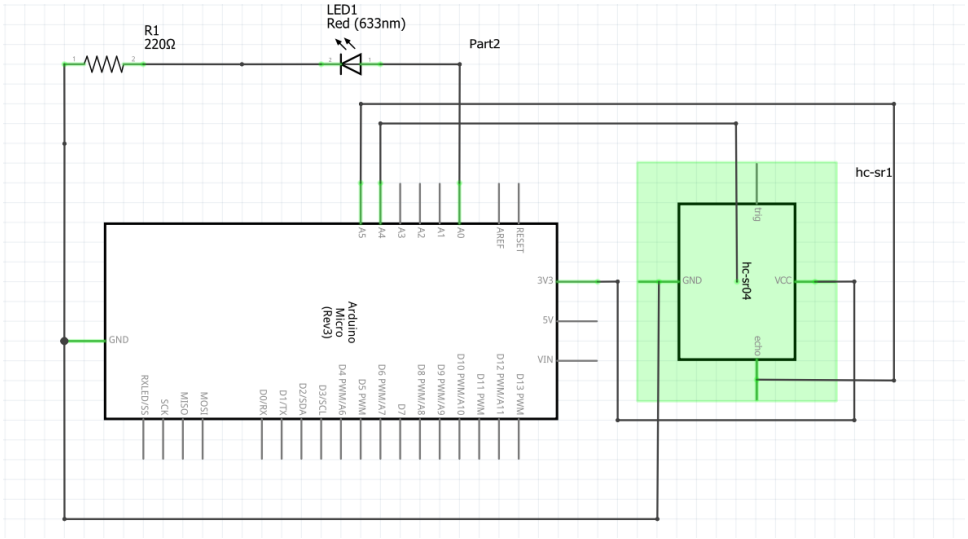
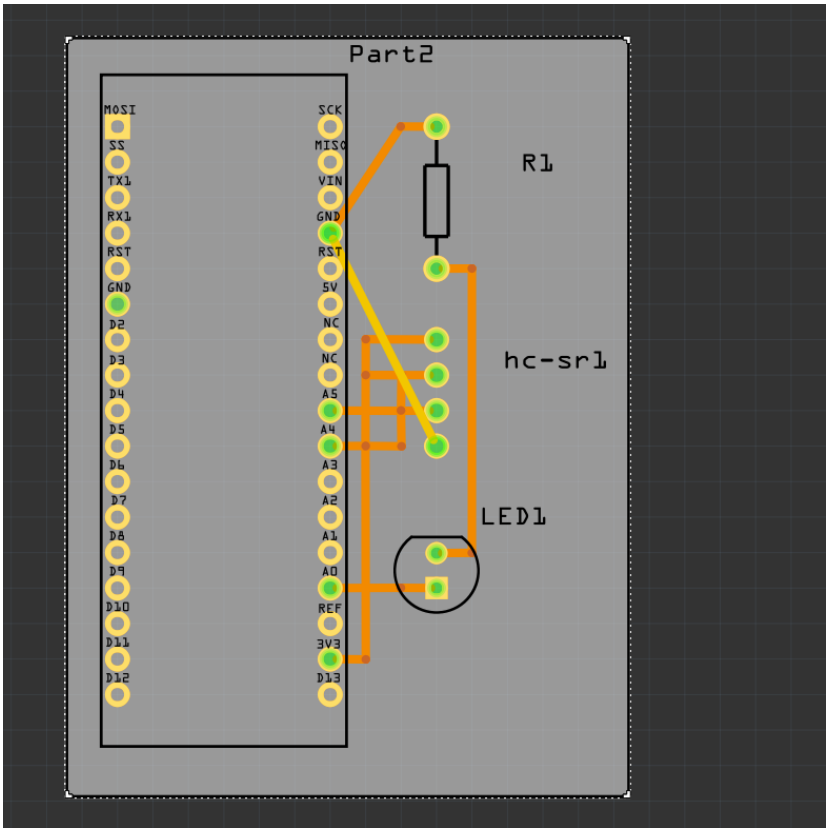
breadboard:



Schematics:



PCB:



Code:

```
// This #include statement was automatically added by the Particle IDE.
#include <MQTT.h>

// This #include statement was automatically added by the Particle IDE.
#include <HC_SR04.h>


SerialLogHandler logHandler;
const int desiredHour = 11;
const int desiredMinute = 20;
double cm = 0.0;
const unsigned long DELAY_DURATION = 12 * 60 * 60 * 1000; // 12 hours in milliseconds
unsigned long detectionEndTime = 0;

int trigPin = D4;
int echoPin = D5;
#define LED_PIN D0

// Define the maximum distance in centimeters for an object to trigger the wave detection
#define DET_DISTANCE 1
HC_SR04 rangefinder = HC_SR04(trigPin, echoPin);

bool msgsend = false;
bool objectDetected = false;
```

```
void messageReceived(char* topic,byte*payload, unsigned int length);
// Initialize the MQTT client and define the MQTT broker settings
MQTT client("broker.emqx.io", 1883, messageReceived, true);
const char* topic = "SIT210/medicine";



// Define a callback function for when a message is received on the subscribed topic
void messageReceived(char* topic,byte*payload, unsigned int length) {
 // Print the received message on the serial monitor

   String topicStr = String(topic);
 String payloadStr = "";
 for (int i = 0; i < length; i++) {
  payloadStr += (char)payload[i];
 }
 // Print message to serial monitor
 Serial.print("Received topic1 message: ");
 Serial.println(payloadStr);

 }



void setup()
{
   Particle.syncTime();
   Serial.begin(9600);
   pinMode(LED_PIN, OUTPUT);
   pinMode(trigPin, OUTPUT);
   pinMode(echoPin, INPUT);
   Spark.variable("cm", &cm, DOUBLE);
   client.connect("particle_device");
   client.subscribe(topic);
}

void loop()
{
   // Get the current time
   Time.zone(+10);  // Set the time zone to Melbourne (+10 UTC offset)
   int currentHour = Time.hour();
   int currentMinute = Time.minute();



  if (currentHour >= desiredHour && currentMinute >= desiredMinute)
   {
      cm = rangefinder.getDistanceCM();
      if(!msgsend){
         Particle.publish("take_medicine", "take_medicine", PRIVATE);
         Serial.println("its time to eat medicine");
         client.publish(topic, "turn_on_buzzer");
         digitalWrite(LED_PIN, HIGH);
         msgsend = true;
         detectionEndTime = millis() + DELAY_DURATION;
      }
```

```
    if ( cm <= DET_DISTANCE) {
       if(!objectDetected){

          client.publish(topic, "detected action");

          Serial.println(cm);
          objectDetected = true;
          digitalWrite(LED_PIN, LOW);

       }

    }


    if (msgsend && millis() >= detectionEndTime) {
       msgsend = false;
       objectDetected = false;
     }
}


       client.loop();
       delay(2000);


 }
```