SINGAPORE POLYTECHNIC

SCHOOL OF ELECTRICAL AND ELECTRONIC ENGINEERING
AND
MECHANICAL AND AERONAUTICAL ENGINEERING

In partial fulfilment of the requirements for the module
ME2203 : ENGINEERING EXPLORATION AND DESIGN

By

Woo Kah Hoong Sean (P1822869)
Jeren Tong En Cheng (P1805350)
Ashwin Dinesh(P1802360)

To lecturers:
Mr Roy Ang
Mr Teo Shin Jen

Laser Sensor on the cloud

Acknowledgement

The team would like to express our gratitude to Mr Teo Shin Jen, lecturer, School of Electrical and Electronic Engineering, for introducing us to Cloud Computing. The team would like to thank Mr Roy Ang For his insight into mechanical engineering and the EA team.

TABLE OF CONTENTS

Summary

The report highlights the innovativeness of Cloud Computing and its ecosystem in Amazon Web Services (AWS) And what can be achieved with the internet of things

AWS is a subsidiary of Amazon, which offers cloud-based services from web hosting to machine learning. Established in 2006, AWS has since expanded to be the single largest cloud platform on the market today. Its clients range from Airbnb to Under Armour, even local firms like DBS.

Cloud computing enables anyone from an individual developer to multinational corporations to take advantage of a large pool of available hardware and software in any part of the world to do anything from developing new software or handle business operations. Furthermore, existing resources on-site can be integrated with the cloud, for a variety of reasons ranging from compliance to cost.

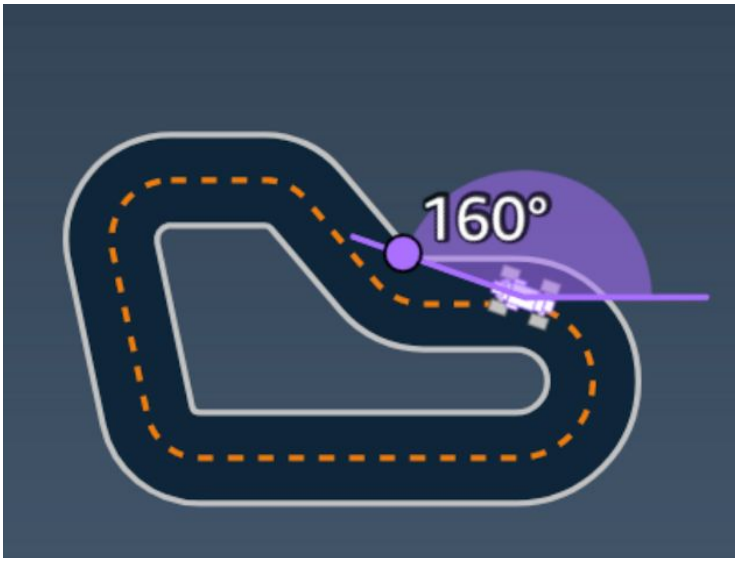
Our prototype Shows the innovativeness of IOT via a laser sensor and we have written a detailed report on it.
This report covers the problem that our product will solve, the design process into making our product, the other products that you can make with it, the finalised design of the product, how to use our product and improvements we can make for the product

1. Background

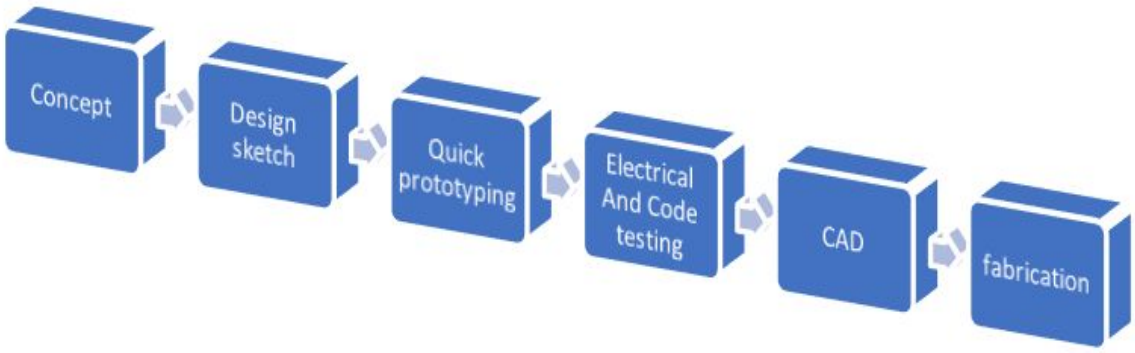
AWS started operating commercially in 2006 as a subsidiary of Amazon.com that provides on-demand cloud computing platforms to customers on a pay-as-you-go basis. Fees are based on a number of factors, like the services required and the hardware of the host machines required. As part of the subscription agreement, Amazon provides security for subscribers' system and handles the maintenance and upgrades of the subscribers' systems. AWS operates from many global regions including Singapore.

Aws,often holds submits in their key global location every year. During which, they hold the Aws Deepracer league .The deep racer league is a 3D racing simulator, and global racing league where participants the use machine learning and Get hands-on with a fully autonomous 1/18th scale race car driven by reinforcement learning. The aim of the competition to see who clears the track in the shortest amount of time. the figure below shows one of the many circuits used.

A lot of the time, lap timing can be very fast, mostly below 15 secs .The lap timing for now is taken down by a person with a stopwatch. WIth a race that is very reliant on small accurate timings, human error can become a huge problem where a delayed timing from the timekeeper can cost you your spot in the team. Thus, our team wanted to design a product to take down an accurate time in seconds and milliseconds and send it to the cloud for judging to make the competition more fair and more accurate



2. DESIGN PROCESS

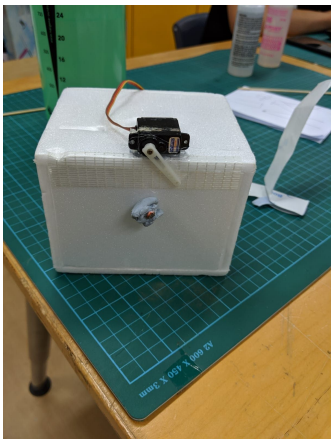


We started out with a concept in our teams mind. We translated our concept to sketches then we did some quick prototyping with scrap and recycled materials such as cardboard paper packing foam. After that we went through electrical and code testing with the prototypes .Then we made the CAD the followed by fabrication.

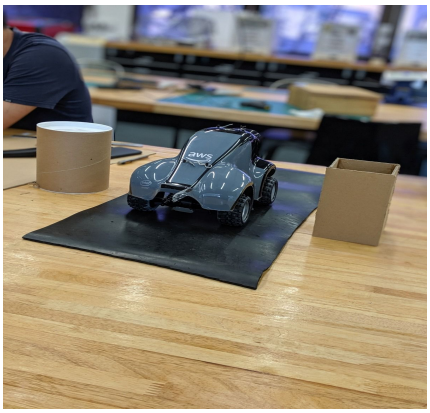
Here are our ideas:

3. Quick Prototyping

We quickly tried to make a prototype of our design with a laser pointer, arduino, and but however the height of our laser was too far off the ground that it was over the deep racer car, the servo motor vibrated too much so the laser was always moving that the serial monitor would have sudden change in values when the light intensity changed only once and will stop

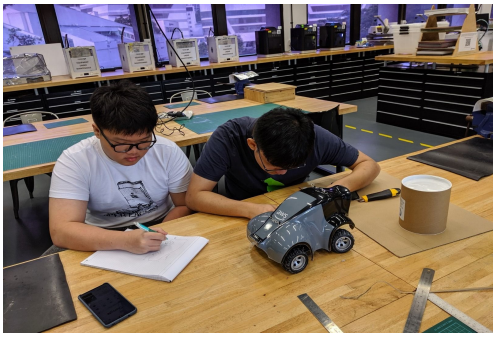


Which is why we got the deep racer car from our lecturer and measured the dimensions of the car to level our laser to the right level, we removed the servo motor. We also realised that our circuit was connected wrongly and that the LDR was not working so we fixed those problems as well



5. Examples/Applications

Though our product is mainly created for the AWS Deepracer, we believe people can either improve or make entirely new projects based on our product. Here are some ideas we thought of:



1. An alarm system

   We made it so that our product is modular, so the code can be modified so that when something passes the laser, it sends a signal to the police, acting as a house alarm system.

2. Accurate speed analysis

   Our product gets rid of human error when timing objects, so the code can be modified to calculate speed using distance/time. This information can them be sent to the could so that researchers can use the data immediately.

6. FINAL DESIGN

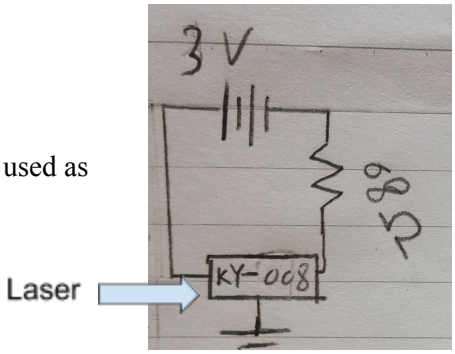
6.1. Electrical design

We thought of the design of the circuit and decided on a simple circuit that would be easy to assemble and operate. We also decided to make it modular to open opportunities for future add-ons for different applications or to improve the product. Below are the final schematic designs and the implementation that we came up with:

TRANSMITTER CIRCUIT:LASER

In part for our modular design, any power source or resistor can be used as long as the circuit has 30mA-60mA of current

Even the laser can be replaced with something more powerful or even from your own laser pointer the circuit will work

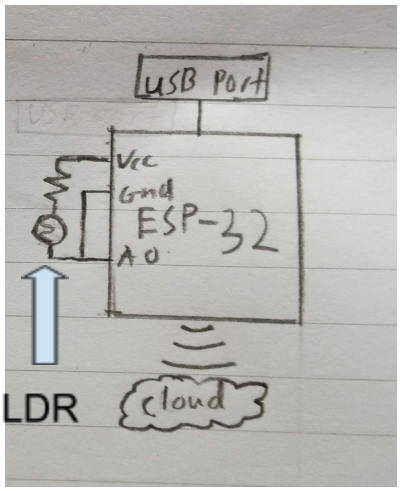


RECEIVER CIRCUIT:STOPWATCH

ESP-32 is used as a microcontroller and also a wifi module which will send timings to the AWS server

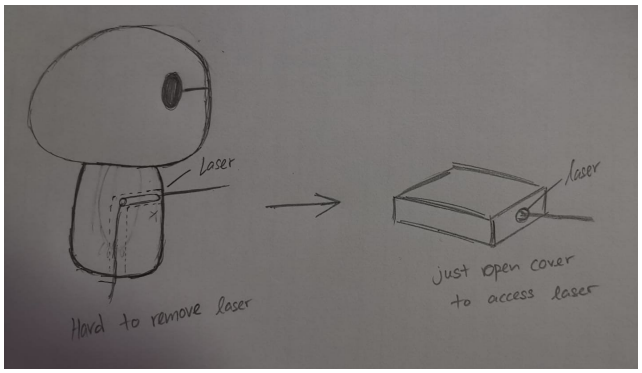
ESP-32 can take timings with the LDR, where its resistance changes with light intensity

Allows the ESP-32 to take timings and send it to cloud, which will be explained with more detail at coding design(Page 16)

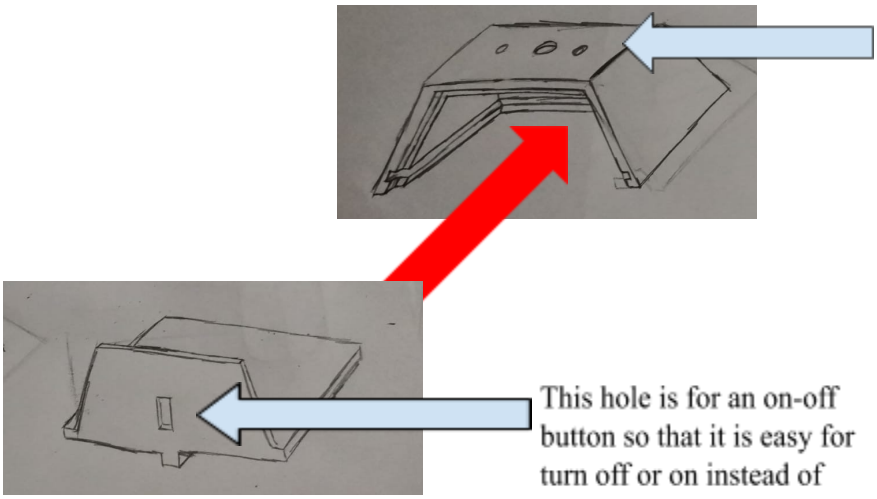


6.2. Design

We wanted something to make sure the cars do not harm the circuits when they hit the product. So we decided to create a casing that would protect the circuits while acting as a stand for the laser and LDR. As seen from the concept designs, we wanted the laser to go through the model, however, since we decided that we want our product to be modular, we moved the position of the laser and the LDR to a seperate case so that it would be more easily accessible.



For the base, we decided that we want one that is able to carry the model, as well as protect the circuit. So we designed the base to be wider at the bottom for stability. The bottom of the base is made to slide out so that circuits can be accessed much easier.

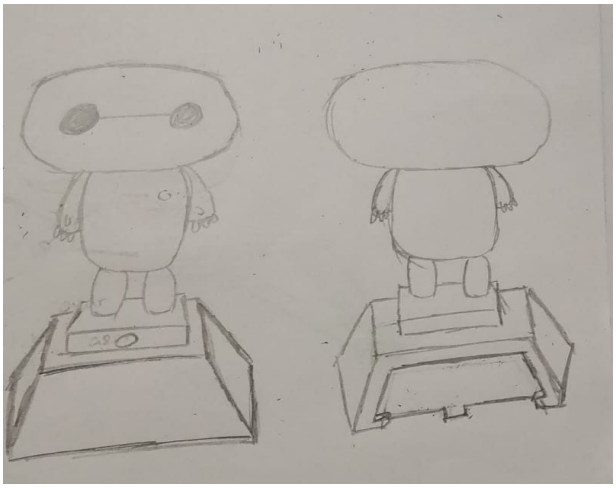


Magnets

Once the circuit is in place, the circuit holder can be pushed into the base with the help of slots on both sides, so that it can slide in easily

This hole is for an on-off button so that it is easy for turn off or on instead of taking off the circuit holder to off.

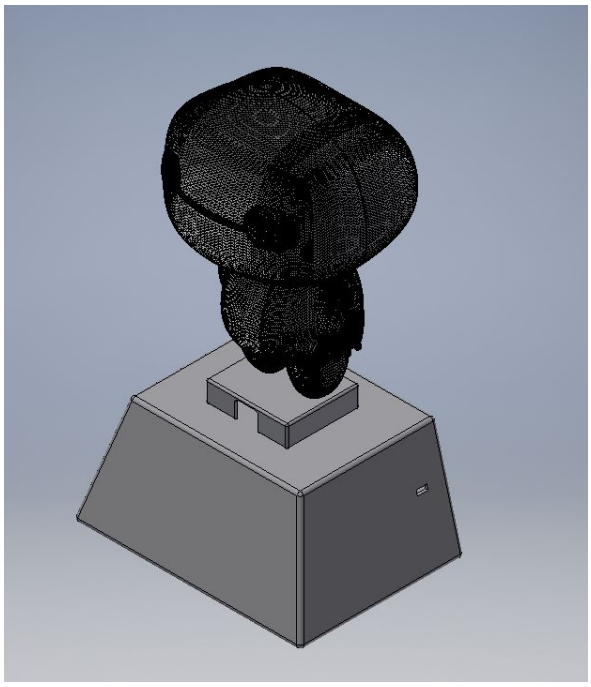
For the model, we chose to have baymax as the model as it was neutral to the theme. We used 3D scanning to scan a solid model to get a hollow model. However, we found a better model online and decided to use that instead.



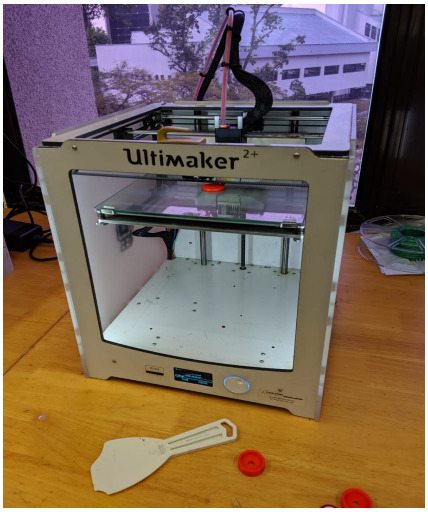
The model is attached onto the cover of the laser and the LDR with glue. Initially, the laser and LDR would be implemented in the model, however, due to complications (Inventor unable to handle mesh and extruding hole), the laser and LDR are now put in a case under the model. Since the cars have image recognition, the model is repurposed as a image that will prompt the car to avoid the product. The model would be split into 3 parts so that the printing time would be be decreased and the amount of support would also significantly decrease.

7. Fabrication

CAD ASSEMBLY



We mainly used 3d printing for our project



Once we 3D printed the parts we attached the parts with glue

8. Code

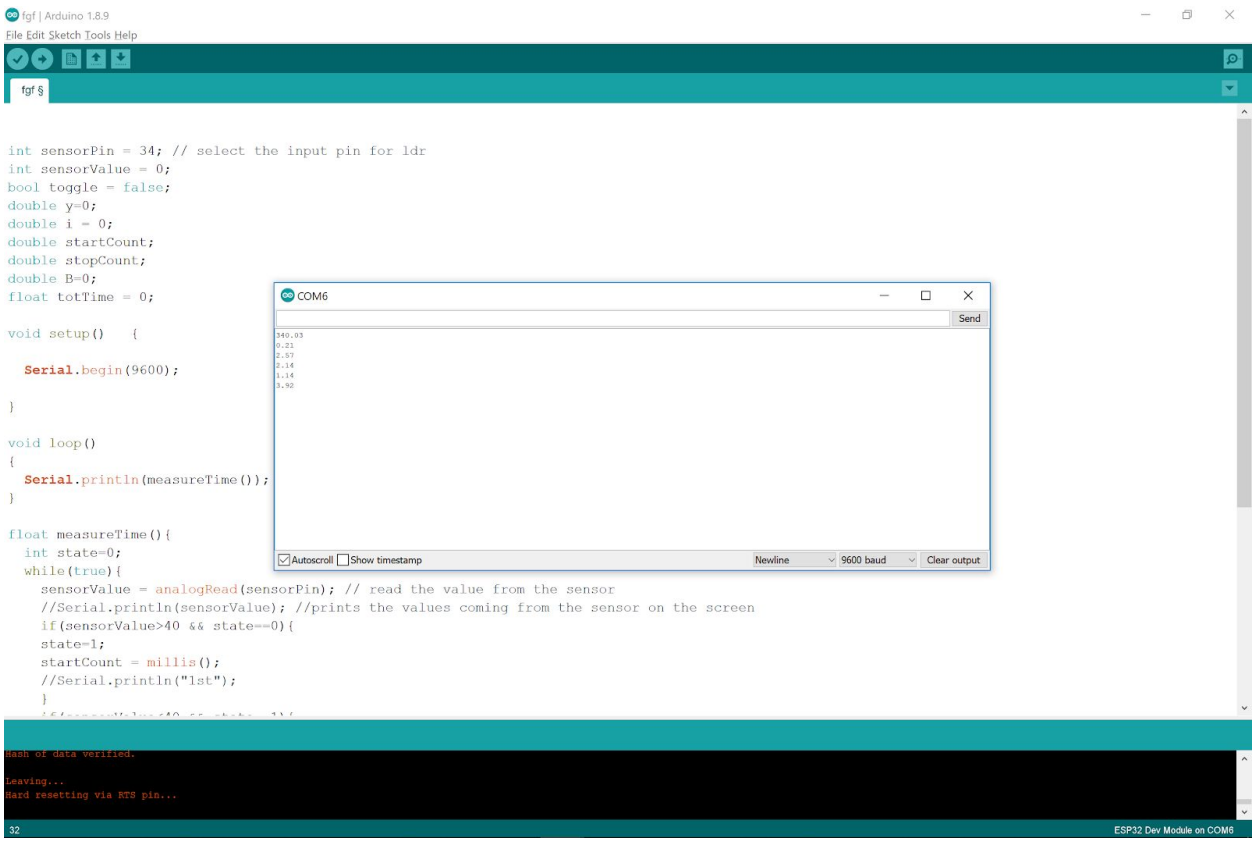
The main part of the code(// is comments)

```
int sensorPin = 34; // select the input pin for ldr

int sensorValue = 0;intiallise

bool toggle = false;

double y=0;

double i = 0;

double startCount;

double stopCount;
```

```
double B=0;
float totTime = 0;
void setup()   {
Serial.begin(9600);
 }
void loop()//loop calculates time diff between the trigger
{
  Serial.println(measureTime());
}
float measureTime()//this function for the time diff calculation of the trigger
{
  int state=0;
  while(true){
    sensorValue = analogRead(sensorPin); // read the value from the sensor
    //Serial.println(sensorValue); //prints the values coming from the sensor
on the screen
    if(sensorValue>40 && state==0){
    state=1;
    startCount = millis();//start
    //Serial.println("1st");
    }
    if(sensorValue<40 && state==1){
    state=2;
    startCount = millis();
```

```
    //Serial.println("2nd);
    }
    else if(sensorValue>40 && state==2){
     state =3;
     //Serial.println("3rd");
    }
    else if(sensorValue<40 && state==3){
      stopCount = millis();//stop
      i = (stopCount - startCount) / 1000;
      return i;
    }
    //delay(100);
   }
}
```

The main part of the code basically calculates the time difference between the trigger points of the laser.we used state as during the debugging of the code we found out that the trigger instantly triggers twice making the time difference zero and to prevent the timer from triggering with inavalid inputs .we used the millis() from the arduino it is the built in clock and timing from when the program starts the below figure shows the output of the esp32 program



As u can see the esp 32 outputs the precise time during output to the milliseconds this shows the code is working

The second part of the code is to send the data to the cloud using mqtt(MQTT is an ISO standard publish-subscribe-based messaging protocol) below is the full code

```
#include <WiFi.h>

#include <WiFiClientSecure.h>

#include <PubSubClient.h>

/*Define All the variables*/

#define LED_PIN 2

int sensorPin = 34; // select the input pin for ldr
```

```
int sensorValue = 0;

bool toggle = false;

double y=0;

double i = 0;

double startCount;

double stopCount;

double B=0;

float totTime = 0;

char charVal[10];


/* Change the following configuration options

This where the wifi and mqtt internet variables such as the certificate and pem keys */
```

```
const char* ssid = "din-2.4ghz";

const char* password = "999999999";

const char* aws_iot_hostname = "aqwnpr6dd5wb1-ats.iot.ap-southeast-1.amazonaws.com";//host adresss

const char* aws_iot_sub_topic = "topic/hello";

const char* aws_iot_pub_topic = "another/topic/echo";//topic to aws

char* aws_iot_pub_message ;


const char* client_id = "MyIoT";


const char* ca_certificate = "-----BEGIN
CERTIFICATE-----\nMIIDQTCCAimgAwIBAgITBmyfz5m/jAo54vB4ikPmljZbyjANBgkqhkiG9w0BAQsF\nADA5MQswCQYDVQQGEwJV
UzEPMA0GA1UEChMGQW1hem9uMRkwFwYDVQQDExBBbWF6\nb24gUm9vdCBDQSAxMB4XDTE1MDUyNjAwMDAwMFoXDTM4
MDExNzAwMDAwMFowOTEL\nMAkGA1UEBhMCVVMxDzANBgNVBAoTBkFtYXpvbjEZMBcGA1UEAxMQQW1hem9uIFJv\nb3QgQ0
EgMTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALJ4gHHKeNXj\nca9HgFB0fW7Y14h29Jlo91ghYPl0hAEvrAIthtOgQ3
pOsqTQNroBvo3bSMgHFzZM\n9O6II8c+6zf1tRn4SWiw3te5djgdYZ6k/oI2peVKVuRF4fn9tBb6dNqcmzU5L/qw\nIFAGbHrQgLKm+a/sRxm
PUDgH3KKHOVj4utWp+UhnMJbulHheb4mjUcAwhmahRWa6\nVOujw5H5SNz/0egwLX0tdHA114gk957EWW67c4cX8jJGKLhD+rcdqsq08
p8kDi1L\n93FcXmn/6pUCyziKrlA4b9v7LWIbxcceVOF34GfID5yHI9Y/QCB/IIDEgEw+OyQm\njgSubJrIqg0CAwEAAaNCMEAwDwYDVR
0TAQH/BAUwAwEB/zAOBgNVHQ8BAf8EBAMC\nAYYwHQYDVR0OBBYEFIQYzIU07LwMlJQuCFmcx7IQTgoIMA0GCSqGSIb3DQE
BCwUA\nA4IBAQCY8jdaQZChGsV2USggNiMOruYou6r4lK5IpDB/G/wkjUu0yKGX9rbxenDI\nU5PMCCjjmCXPI6T53iHTfIUJrU6adTrCC2
qJeHZERxhlbI1Bjjt/msv0tadQ1wUs\nN+gDS63pYaACbvXy8MWy7Vu33PqUXHeeE6V/Uq2V8viTO96LXFvKWlJbYK8U90vv\no/ufQJVtM
VT8QtPHRh8jrdkPSHCa2XV4cdFyQzR1bldZwgJcJmApzyMZFo6IQ6XU\n5MsI+yMRQ+hDKXJioaldXgjUkK642M4UwtBV8ob2xJNDd2Zh
wLnoQdeXeGADbkpy\nrqXRfboQnoZsG4q5WTP468SQvvG5\n-----END CERTIFICATE----- "; //CA certificate
```

```
const char* iot_certificate = "-----BEGIN
CERTIFICATE-----\nMIIDWTCCAkGgAwIBAgIUXuJzTFJSCCEmo9RprHeMAzrP/XMwDQYJKoZIhvcNAQEL\nBQAwTTFLMEkGA1UE
CwxCQW1hem9uIFdlYiBTZXJ2aWNlcyBPPUFtYXpvbi5jb20g\nSW5jLiBMPVNlYXR0bGUgU1Q9V2FzaGluZ3RvbiBDPVVTMB4XDTE5
MDgxMTA0NDYy\nOVoXDTQ5MTIzMTIzNTk1OVowHjEcMBoGA1UEAwwTQVdTIElvVCBDZXJ0aWZpY2F0\nZTCCASIwDQYJKoZI
hvcNAQEBBQADggEPADCCAQoCggEBALYQZgti7twWAZw8r4yU\nfDVeBou2t2KnDRqFSpo13Rj78PJln0F7HZpWpI3eljWjrpclQygV5kQ
H+odzUhKE\nB8VbGfwNJ8Io9/MoRXK0xlGRlwECs1moGI18MIPZVsGs3RuU4ERMf4oVAWD8BuM2\nGmBBEjSgW5yS+pCy3tdnatvEw/
qZ3g4tQa4sWh/hFGWJxOkrRuyVbJnPR4eXN+qI\np3i9a0kkxGCTpGfHc5gHgrWnQEyBj1pnqB3AvzYIpTTUF1XzHt2TET6Wf4EMnPop\n7
9iEjZLJDXnyCdwPfZ7UFotZYTB9oLwO3ghgvzXTutx2giDCI5T2/OkStD/nRhwe\ng/ECAwEAAaNgMF4wHwYDVR0jBBgwFoAUQqISItS8f
9c0YCSAacZPWqmdw3gwHQYD\nVR0OBBYEFJQN5Q0AkM3nC2A+DkDWsSUfpF3vMAwGA1UdEwEB/wQCMAAwDgYDVR0P\nAQH
/BAQDAgeAMA0GCSqGSIb3DQEBCwUAA4IBAQBOrp2Dg5+yAUuJGMJd0QEawFib\nmtuEEFEIMpyk19imI1LTtW38LSKR9fC2aB6zhns
H2o6SBBCz3cQYF21uTQLfOFa7\n04/yJvEKDh4O2UMgqrw74kXMZ+op4C5YrOG4dRS30syEnfM8bYEhyAem2qkxkeqR\ndFE3f0Kh5+75b
YomflV4nqqWWtIHloT5yCorSXx45NpFwJ7sIwOX5BcP+FjlQkzm\nZFGtwbNLDuCyHk0boU3x/ZGX9UC7/67LzAeRzlcaWc9LrydEjFOUA
0DF9SlZpakd\nxcBdKqrVZX6oLZ2IP2fV3o1bWacZluW1+P+bghkoeqWBOyKmn6W8fd5Vra0T\n-----END CERTIFICATE-----\n"; //Own
certificate

const char* iot_privatekey = "-----BEGIN RSA PRIVATE
KEY-----\nMIIEpAIBAAKCAQEAthBmC2Lu3BYBnDyvjJR8NV4Gi7a3YqcNGoVKmjXdGPvw8mWf\nQXsdmlakjd6WNaOulyVDKBXmR
Af6h3NSEoQHxVsZ/A0nwij38yhFcrTGUZGXAQKz\nWagYjXwwg9lWwazdG5TgREx/ihUBYPwG4zYaYEESNKBbnJL6kLLe12dq28TD+p
ne\nDi1BrixaH+EUZYnE6StG7JVsmc9Hh5c36oineL1rSSTEYJOkZ8dzmAeCtadATIGP\nWmeoHcC/NgilNNQXVfMe3ZMRPpZ/gQyc+inv2I
SNkskNeflJ3A99ntQWi1lhMH2g\nvA7eCGC/NdO63HaCIMIjlPb86RK0P+dGHB6D8QIDAQABAoIBACuXE4eIYV4KWspY\ncUN8TZzNb
U3IKnIRQv7liO4RFWWk9JRkplOcMYNBWVaqQ1aGYQfJISIY1C/XpTEG\n+FB8kgu7eyYw1CMVggAS1UXYaF8hN4FDb0N5Dp/Zaxg2U
iI+zBWNeICBvEuIKfNZ\nhCCkVH5VNhGpjyWWvy0j1lzGYvxI4SqevyuOF1N03TRv5IsR+2SMPYn54eHVWYDi\nXofqz9uXilm3mDyXvF
h08NJ6Adu6wNdW53sasq8P+Xl9wDEbr9qvfcOxxAdPM0K3\nShQIbgI/Fuy42bAyDDVdPm9hDiwOwbvutVG+KMhjE0m+ufLSC1qWQRKe
UtX9k+0R\n5qzIedECgYEA7oSPsmqa1UtCORXPu7DvYIMbhLsoD6JtJ2f3ajT3dIStzUD90E/1\nHTFmHupla3xOpYfPwjixhfYeCFknNg7Xaqj
siXI68yhk27nJ5Zw3n3AIEpDf+7gC\ngZiMxdLi30Yu7kLnu3btvq2MzMFhTPW2fj3pC0vRMOqJpGBORJ/YA80CgYEAw2iQ\nnDVHGTxb4i
PHW6BhUIB/e5xIvPPfLNREVZv2latav8zEIjec3XNpBuW+V82sQk5y\na40P0iZJenTu+xn7KGdF6VdRI6Hx+BA+zdPdGpLa3Ef0awAVijLQ
GoqshTjIqOMa\noYnsGy9/U3eg0W5HkwprIwG/EdWDNMzavlJHJLUCgYEApl+NJHS7/9/5F3yhZFit\nAPMy9MjfztgLr5sTt9+uZHZQDYYy
BWZgobZdd18k027oJQqw3tdQ7RKaLFSTx19e\n36VpKQ6nFAmvIJ1duW3YVrx8xQaNVjYqeV+BglPxsLBx9/xvTBJPD9xbBFA3oMkc\nPL
7OmfSOvMI8tI4Ayll9AlkCgYEAwDJ86efNQpkHlumyEAEzNkbtHNue/5qujB1M\nCRgZ9SRRYg05bJ5ef8IgL1SRXzNAlwcpq3dXdE+MPhH
9bKUlT5wkO1pSyEVlBaSU\n4UXdV2HSOo/ba9LtXLyVy0VSYsZzNwoJC5Ho1ORmy+YCG3vokkvRDOGO2S+0mZ5U\nGJY8u3ECgYBuf
3kACTejiycg7AgbDSder3SgvIPU5a34xPw4evB0L+XO2ViDlzMo\nVfaLeAcomCru16dbjzK0unaffQ6sRPFk6lmwtcXjoEtdnO4Tv4ogDXVmv
1BSvT9T\nY8e5c7gr4AQ0uuWFTRhVBqmooa61gm4gFfZkqkscD50x/YIkLiLqLA==\n-----END RSA PRIVATE KEY-----\n"; //Own private
key



#define SSID_HAS_PASSWORD //comment this line if your SSID does not have a password



/* Global Variables */

WiFiClientSecure client;

PubSubClient mqtt(client);
```

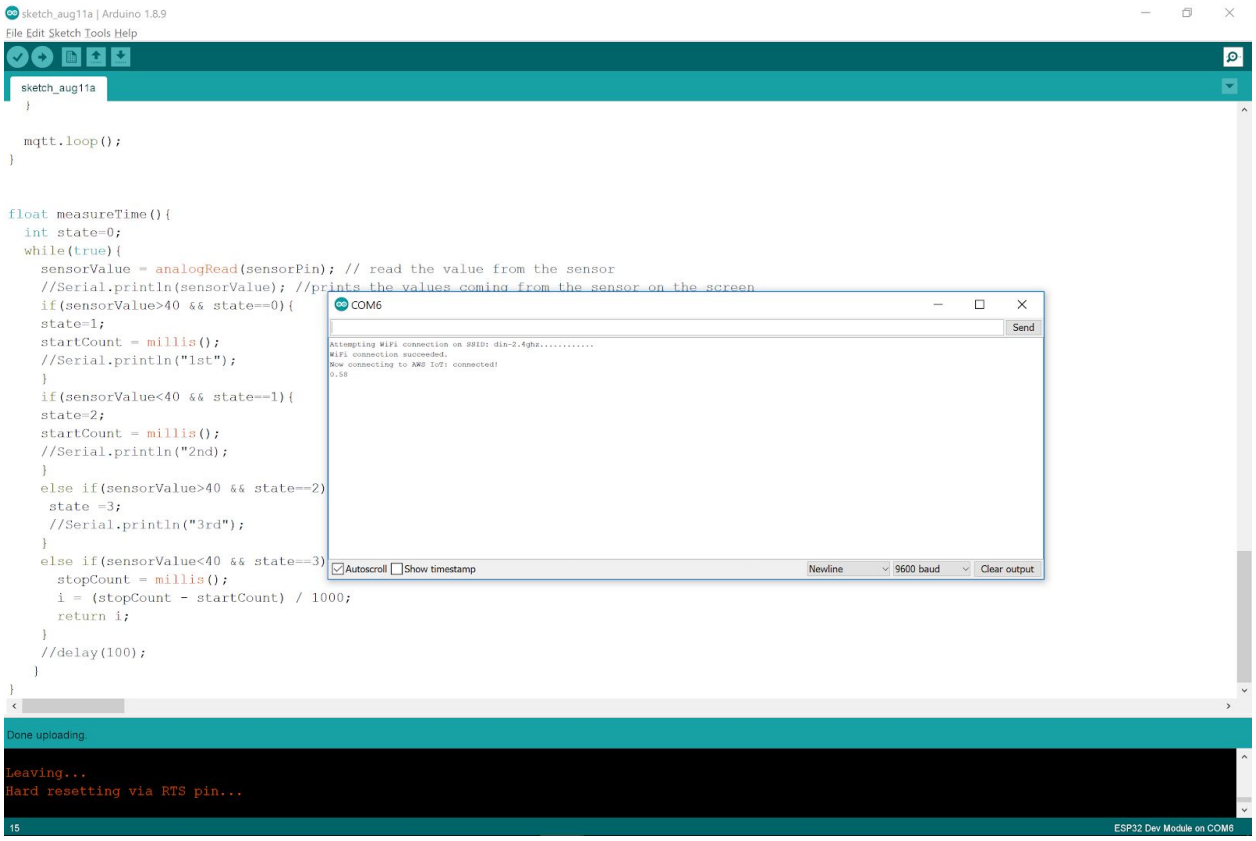
```
/* Functions */
void sub_callback(const char* topic, byte* payload, unsigned int length) {
  Serial.print("Topic: ");
  Serial.println(topic);

  Serial.print("Message: ");
  for (int i = 0; i < length; i++)
    Serial.print((char) payload[i]);
  Serial.println();

  if ((char) payload[0] == '1')
    digitalWrite(LED_PIN, HIGH);
  else if ((char) payload[0] == '0')
    digitalWrite(LED_PIN, LOW);

  mqtt.publish(aws_iot_pub_topic, aws_iot_pub_message);
}

void setup() {
  //Initializations
  Serial.begin(9600);
  Serial.print("Attempting WiFi connection on SSID: ");
  Serial.print(ssid);
  pinMode(LED_PIN, OUTPUT);
  digitalWrite(LED_PIN, LOW);
```

```
  // WiFi
  #ifdef SSID_HAS_PASSWORD
  WiFi.begin(ssid, password);
  #else
  WiFi.begin(ssid);
  #endif

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print('.');
  }
  Serial.print("\nWiFi connection succeeded.\n");

  client.setCACert(ca_certificate);
  client.setCertificate(iot_certificate);
  client.setPrivateKey(iot_privatekey);

  // AWS IoT MQTT uses port 8883
  mqtt.setServer(aws_iot_hostname, 8883);
  mqtt.setCallback(sub_callback);
}
```

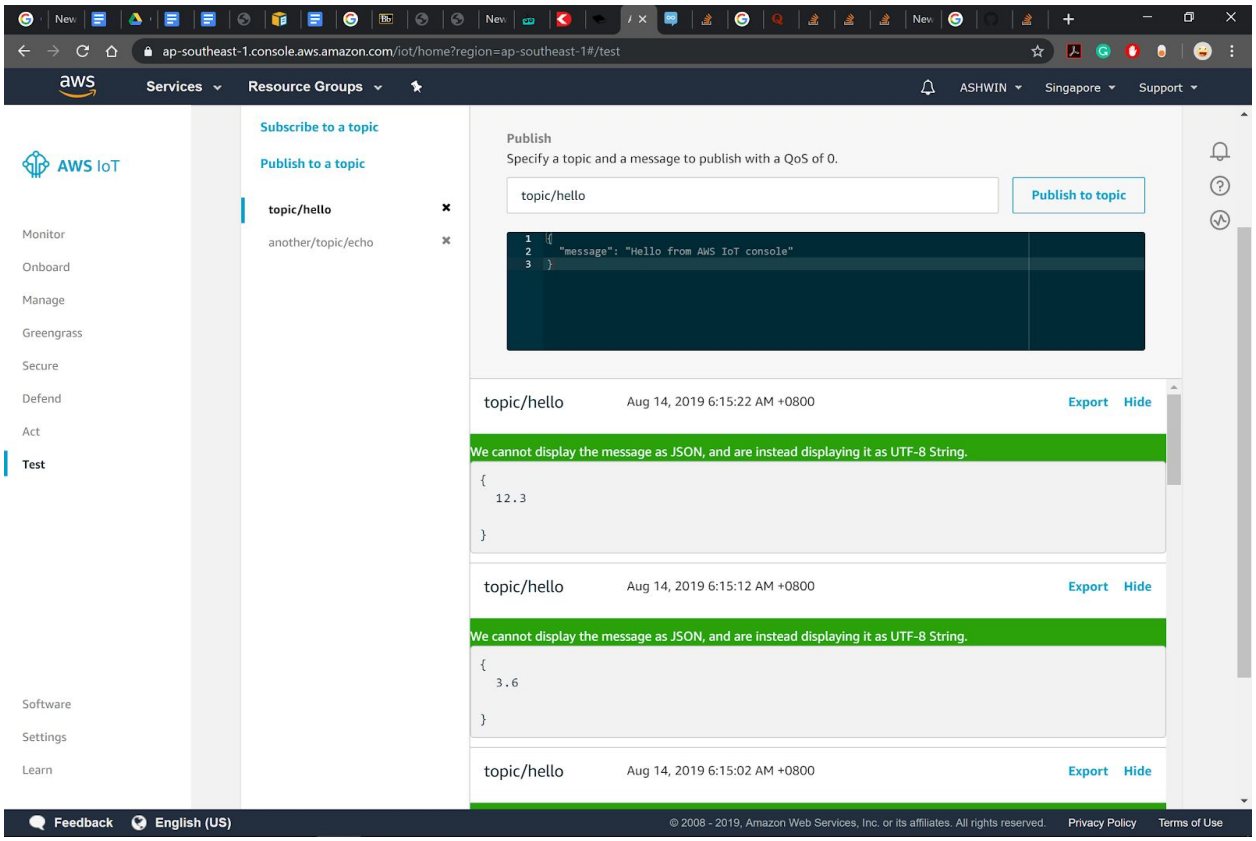
```
void loop() {
  // reconnect on disconnect
  while (!mqtt.connected()) {
    Serial.print("Now connecting to AWS IoT: ");
    if (mqtt.connect(client_id)) {
      Serial.println("connected!");



      Serial.println(measureTime());
dtostrf(measureTime(), 4, 3, charVal);//Convert float to string to  be published
    mqtt.publish(aws_iot_pub_topic, charVal);




    } else {
      Serial.print("failed with status code ");
      Serial.print(mqtt.state());
      Serial.println(" trying again in 5 seconds...");
      delay(5000);
    }
  }

  mqtt.loop();
}
```

```
float measureTime(){
 int state=0;
 while(true){
  sensorValue = analogRead(sensorPin); // read the value from the sensor
  //Serial.println(sensorValue); //prints the values coming from the sensor on the screen
  if(sensorValue>40 && state==0){
  state=1;
  startCount = millis();
  //Serial.println("1st");
  }
  if(sensorValue<40 && state==1){
  state=2;
  startCount = millis();
  //Serial.println("2nd);
  }
  else if(sensorValue>40 && state==2){
   state =3;
   //Serial.println("3rd");
  }
  else if(sensorValue<40 && state==3){
    stopCount = millis();
    i = (stopCount - startCount) / 1000;
    return i;
  }
  //delay(100);
 }
}
```

The code shown above first connects to your wifi once it have connected to wife it will connect to aws iot core and one it has connected to iot core it starts the main part of the program which is the measuretime() and publishes the mqtt to the iot cloud in topic .The figure below shows the iot receiving the message.



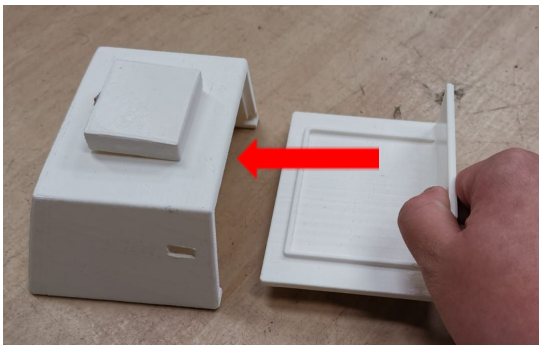
As You can see from ide console after the iot is connected main program starts and runs then publishes to cloud



This topic receiving the message via the cloud . The code works well as it updates to the mqtt and is able to be exported .

9. How To Use

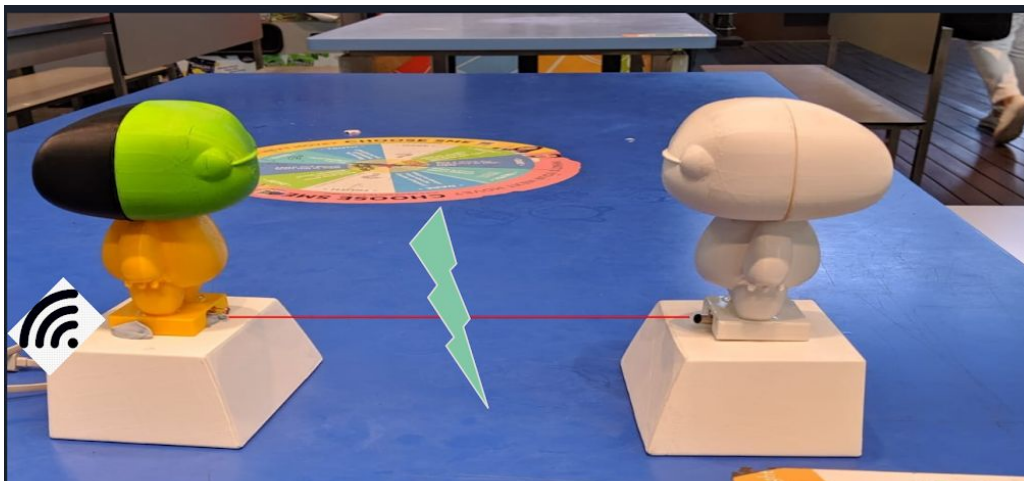
1) Place circuit on the circuit holder slide the circuit holder into the slot in the base



2) Cover the laser and LDR with the laser box which are attached to the base with magnets

3)repeat step2 and 3 with the esp side

4)Place the models as such and switch on the microcontroller and laser module and aim the laser to the ldr.



5)To start time the laser has to interrupted ,the next the laser is interrupted the the timer stops and publishes the timing to the cloud via mqtt.

10. Conclusion

Cloud computing has certainly brought about new possibilities in this digital day and age. From opening up new opportunities for entrepreneurs to multinational corporations accelerating their processes. While cloud computing has become prominent only more recently, the benefits of cloud computing, such as reducing the cost of accessing traditional server capabilities like mass data storage or processing, and misconceptions have already emerged.

Our prototype in general works as intended . We learnt what can be achieved through Iot and the limitation along the way we have changed and improved our Design and code to make it more reliable.we

11. Recommendations and improvements

- The laser should have a mechanism to aim
- The base should have rubber for friction to prevent the product to move if a car hits
- The design could be improved to become modular for other purposes
- Code can be further improved
- The laser should have a mechanism to aim
- The base should have rubber for friction to prevent the whole structure from moving if hit by car
- Parts should be joined by magnets and separated into smaller parts for it to be more modular

12. 2D drawings