**SEMM3253-22**

**Project Theme: Home IOT automation**

**Project: IOT security system**

**Lecturer: Dr Shaharil Mad Saad**

**Muammar Royyan Ibrahim**

**A18KM3011**

**School of Mechanical Engineering**

**Universiti Teknologi Malaysia**

**Video**

youtu.be/M3HDAHMt9Vs

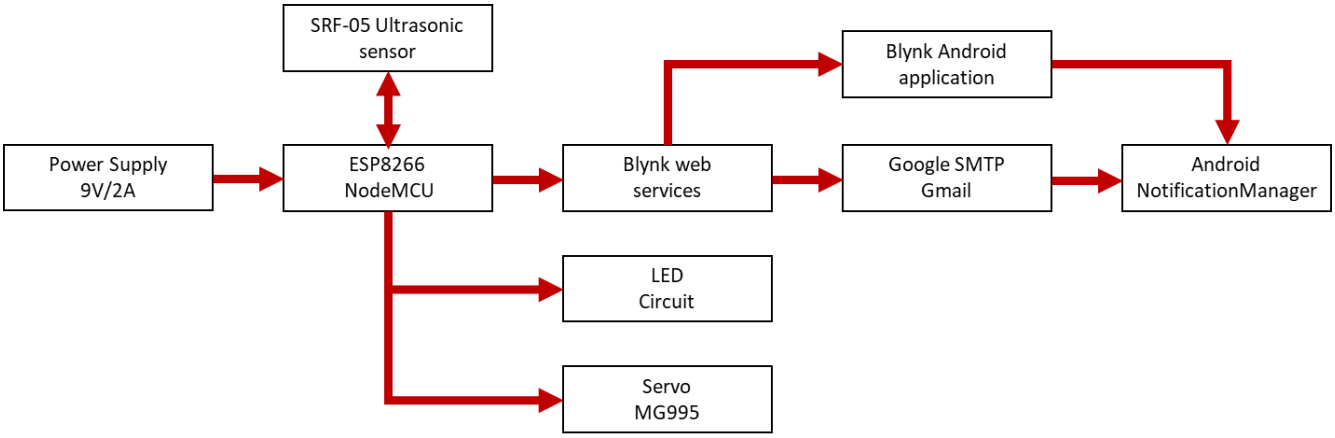
## Introduction

IoT, is a system of interrelated devices connected to the internet to transfer and receive data from one another, it involves the control of consumer tools and appliances using the internet. Unit recently the idea of IoT is not feasible but now with the growth in popularity of wireless networks, the newest home appliances are capable of connecting to wireless data networks. Smart homes are the best example of the implementation of this idea.

IoT home automation is the ability to control domestic appliances by electronically controlled, internet-connected systems. It may include setting complex heating and lighting systems in advance and setting alarms and home security controls, all connected by a central hub and remote-controlled by a mobile app. In essence the 'things' around the house can share information to produce outputs. For example, when your alarm rings in the morning, it sends data to the electronic window shades to open and water heater to turn on. Furthermore, the data from sensors can be sent to cloud IoT services which allows for more sophisticated functions such as predictive programs and email alerts. The automated "intelligent" house can provide a more secure, pleasant, and cost effective living environment.

Home security is a fast growing industry with a current market cap of $1.18 Billion and a projected growth rate of 19% a year. This technology is greatly supported by IoT. Most homes use a CCTV device for security, with current IoT service simply being remote viewing of the camera. There is a lack of home security devices which provide deal-time alerts to the user, as CCTVs require advanced AI yet to be in market to sort between threats and acquaintance of home owners. One stop-gap solution is an IoT based vault opening detection device which alerts the home owner through smartphone if a storage medium such as a cubbard or safe has been opened.

### Block Diagram




### Block Diagram Explanation

1. The home automation uses ESP8266 as the main microcontroller, it takes input data from ultrasonic sensor, and outputs data to Blynk web services, and I/O pins
2. PSU is a power bank which supplies 9V/2A current, esp8266 uses 3.3V.
3. ESP8266 connects to local WiFi and authenticates to blynk service using authentication key
4. When ultrasonic sensor doesn't detect object within vlose proximity, event is triggered and esp8266 notify blynk and notifications are sent to user phone

5. LED and Servo activates when event triggered

**Bill of Material**

| Component | Quantity |
|---|---|
| ES8266 | 1 |
| Breadboard | 2 |
| LED | 1 |
| 1K Resistor | 1 |
| USB to MicroUSB cable | 1 |
| MG995 Servo | 1 |
| SRF-05 Ultrasonic sensor | 1 |
| 10k mAh Powebank | 1 |
| Jumper Wire | 10 |

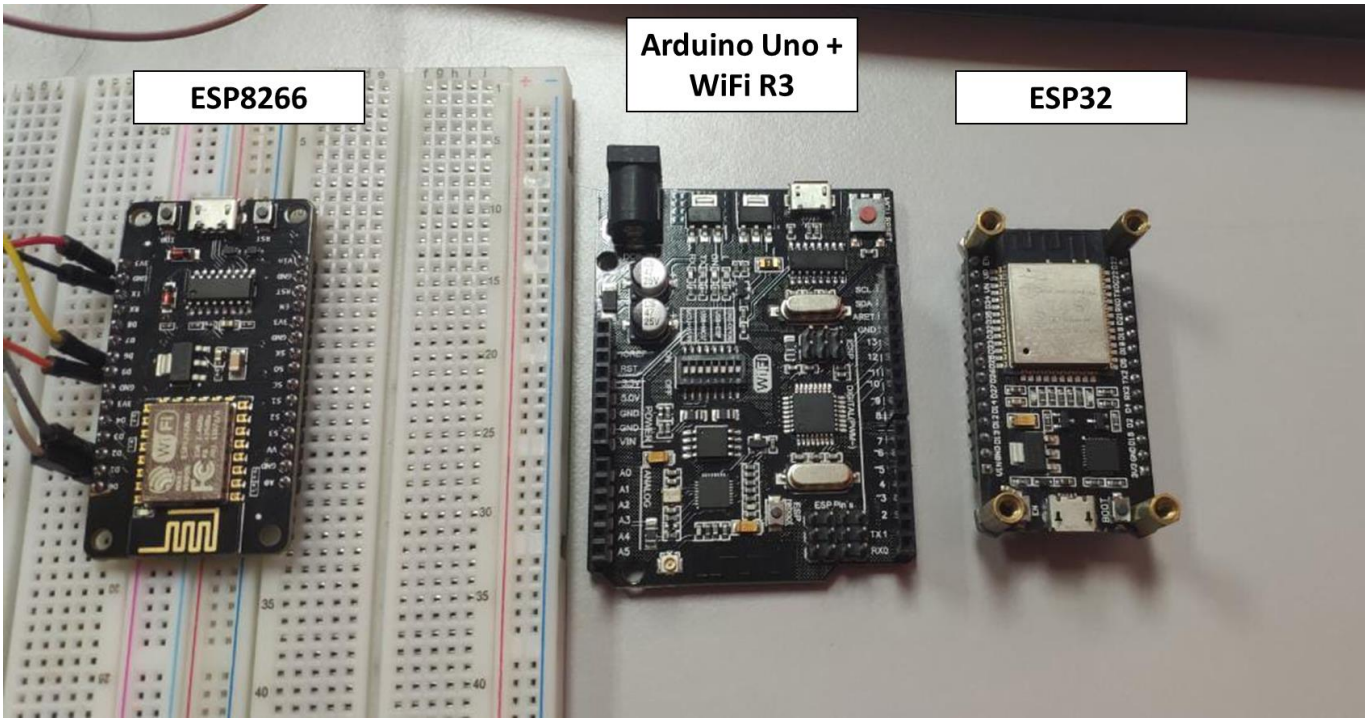
**Required Installations**

1. Arduino IDE
2. BLYNK app
3. HTML5 Supported Browser
4. ESP8266 windows comm port drivers

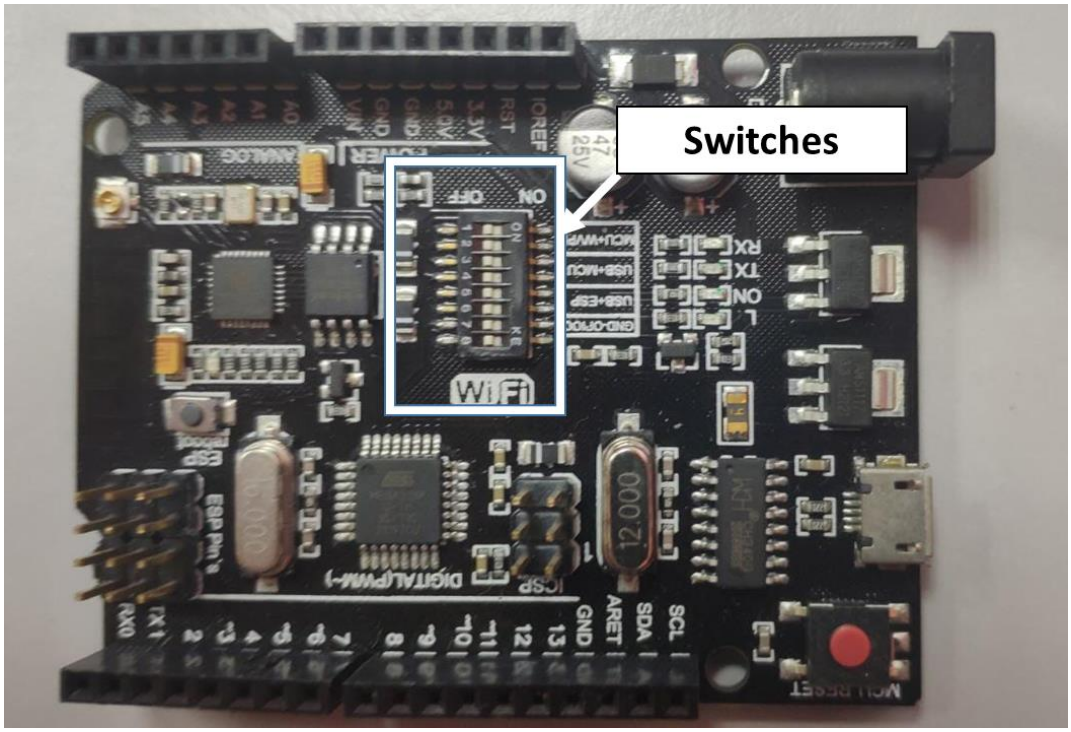
## Mechanical Design

**ESP8266 & Microcontroller considerations**

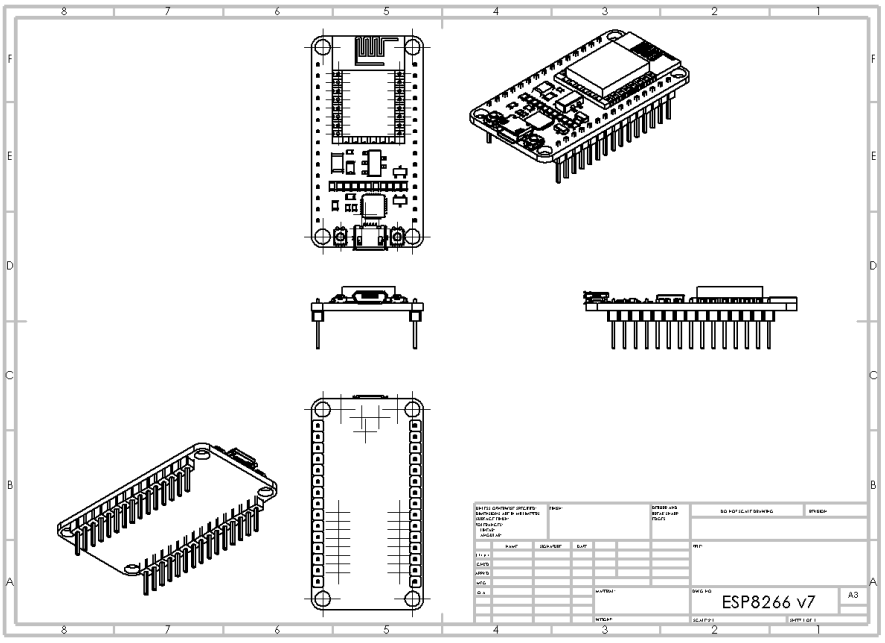
Several considerations were made in choosing the microcontrollers; esp32, esp8266 and Arduino Uno Wifi. Initially the first prototype was fabricated using Arduino Uno + WiFi R3 microcontroller, this is an Arduino uno with built-in esp8266. It works using the 7 switches (fig.2) on the board. Activating Switch 1 and 2 allowed serial communication between Arduino and Esp in a slave-master format, activating switch 3,4 allowed for Arduino programming and 5,6,7 to program the esp8266. The first program was set up for the Arduino to control LEDs, Servo and Sensor, whilst esp communicated with blynk services. However, there were issues with the serial communication between boards, and thus the use of this microcontroller was scrapped (fig.1). A singe esp8266 nodeMCU was then chosen, this microcontroller can replace Arduino completely for control as it technically has a stronger microcontroller. Esp32 was not considered as Bluetooth is not a project requirement. In terms of functionality, Uno's ATmega is the weakest microcontroller, ESP8266 comes second, and ESP32 is the strongest. The Esp32 can connect to both WiFi and Bluetooth whilst maintaining the smallest profile. The Esp8266 can only connect to WiFi, and the Uno needs serial connection with wifi modules to perform such. All 3 boards can be powered using microUSB but the Arduino can use 8mm Plug from PSU.




(fig.1 the 3 microcontrollers side by side, source: my camera)




(fig.2 Arduino Uno + WiFi Switches)

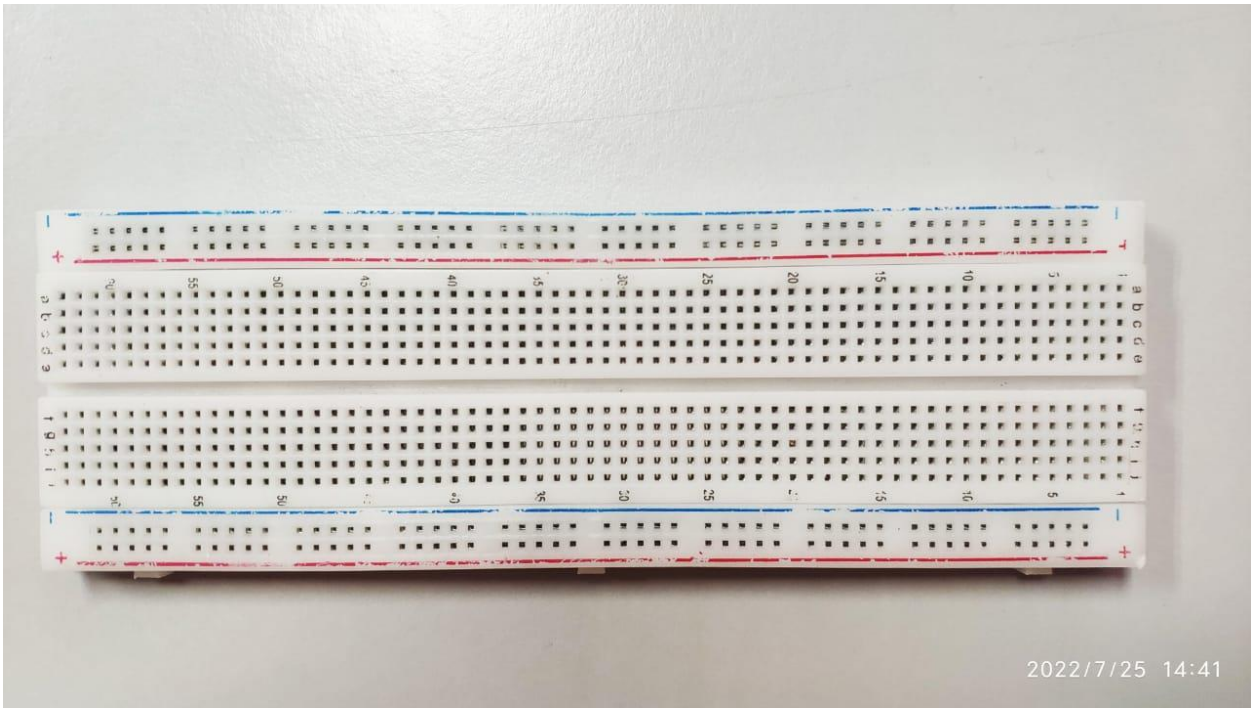



**Breadboard**

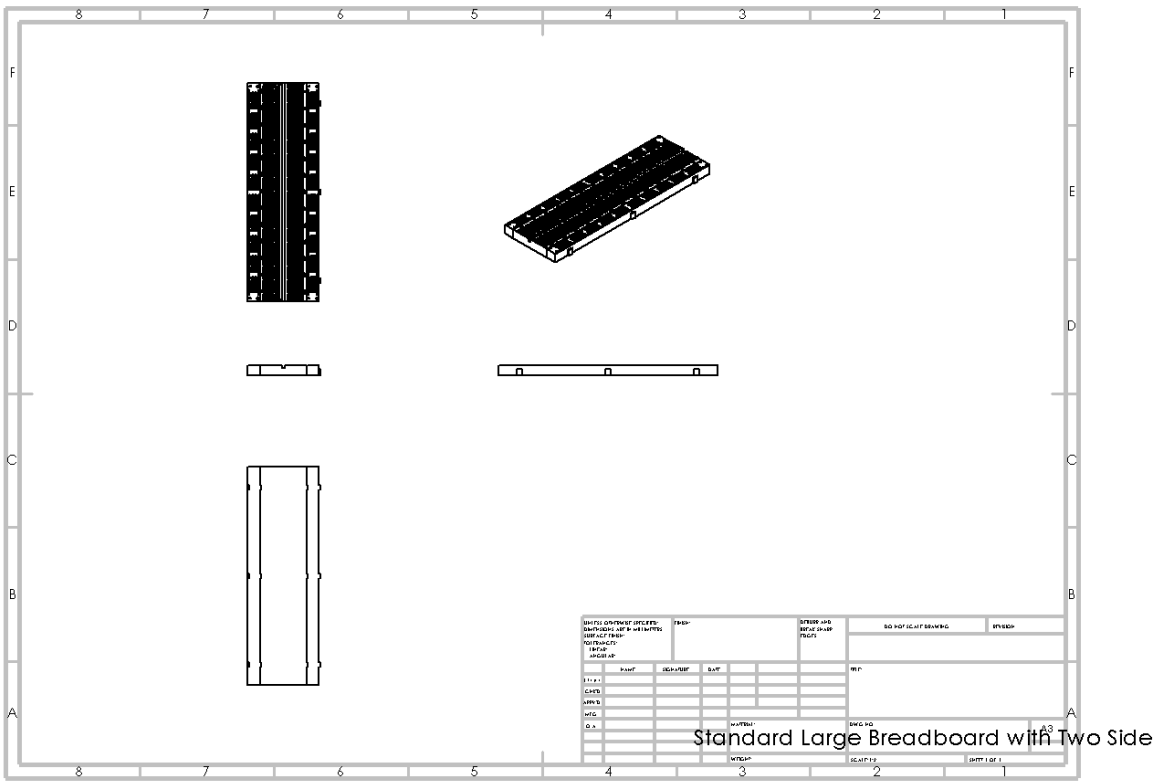
Breadboards are temporary work boards for electronic circuits, they are designed to allow the rapid creation of circuits without the need for soldering or making permanent connections. The general shape of a breadboard is shown in Fig. 3. Compatible with most breadboards, jumper wires are used to connect circuits. Breadboards allow components to be easily inserted and removed allowing for fast prototyping. If an engineer designs a simple module or circuit that they want to test, a breadboard provides a cheap and quick solution (as compared to designing a PCB.

Some breadboards have power rails on either side of the main rows and breadboards are able to house large parts, including DIP 40 ICs. Most breadboards will have clips on the front, back, and sides that allow them to be connected to other breadboards to allow the construction of more complex circuits.

A bare board already has some connections. The horizontal rows are connected throughout the row and may make a complete row with the addition of a simple jumper at the center point. These rows are noted with red and blue or black markings.




(fig.3 breadboard used)



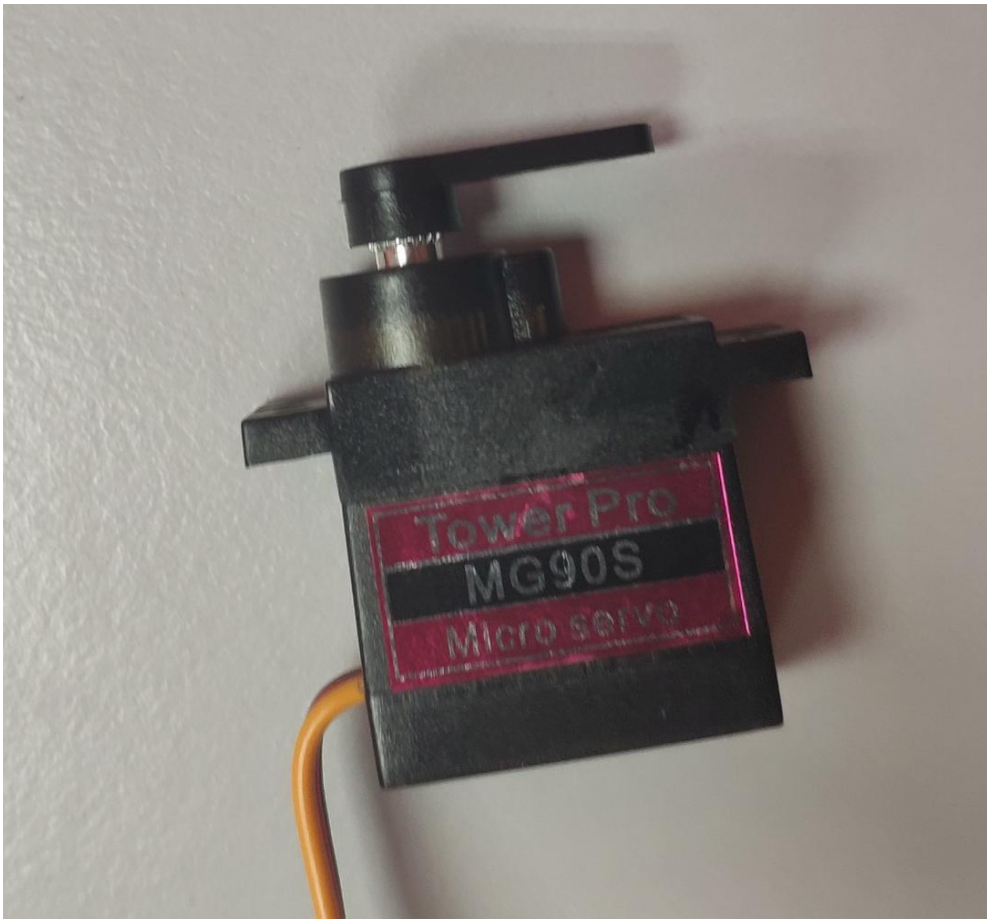

(fig.4 Breadboard design)

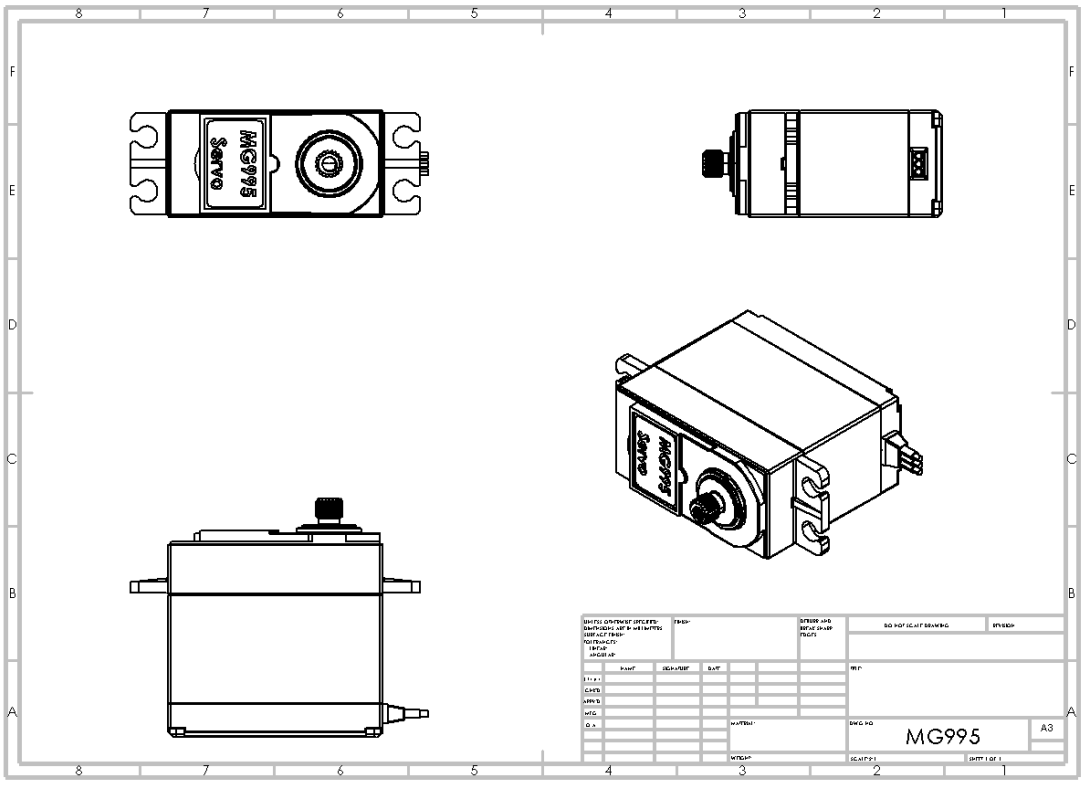
**MG995**

Micro Servo Motor MG995 is a tiny and lightweight server motor with high output power. Servo can rotate approximately 180 degrees. Unlike a DC motor, a servo motor is a type of motor that can rotate with great precision. Normally this type of motor consists of a control circuit that provides feedback on the current position of the motor shaft, this feedback allows the servo motors to rotate with great precision. Servo allows you to rotate an object at some specific angles. A servo motor usually comes with a gear arrangement that allows us to get a very high torque servo motor in small form factor. Due to these features, they are being used in many applications RC cars and Robotics.

MG90s is an alternative to the commonly used SG90s servo used in Arduino projects. This servo has full metal gearing making it more durable than the plastic gearing of SG90s. Normally this servo is used by RC car hobbyists for its durability.

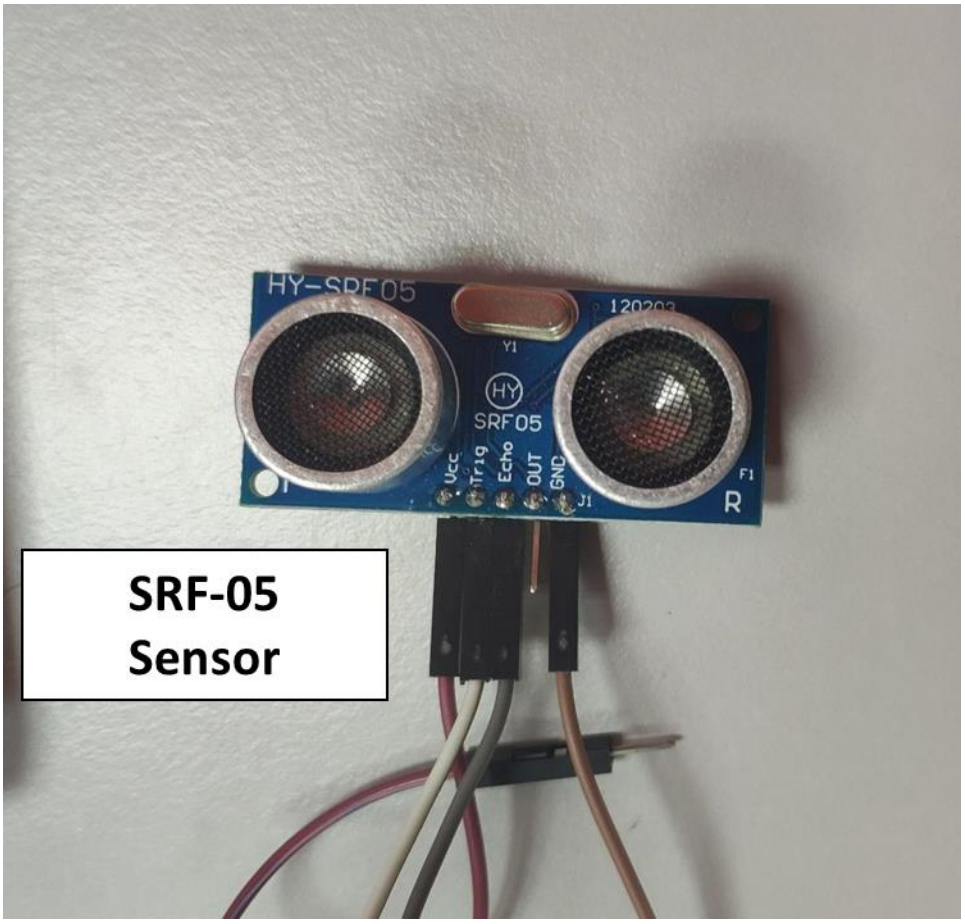
Servo motors are rated in kg/cm most hobby servo motors are rated at 3kg/cm or 6kg/cm or 12kg/cm. This kg/cm is how much weight your servo motor can lift at a particular distance. For example: A 6kg/cm Servo motor should be able to lift 6kg if the load is suspended 1cm away from the motors shaft, the greater the distance the lesser the weight carrying capacity.  The position of a servo motor is decided by electrical pulse and its circuitry is placed beside the motor.



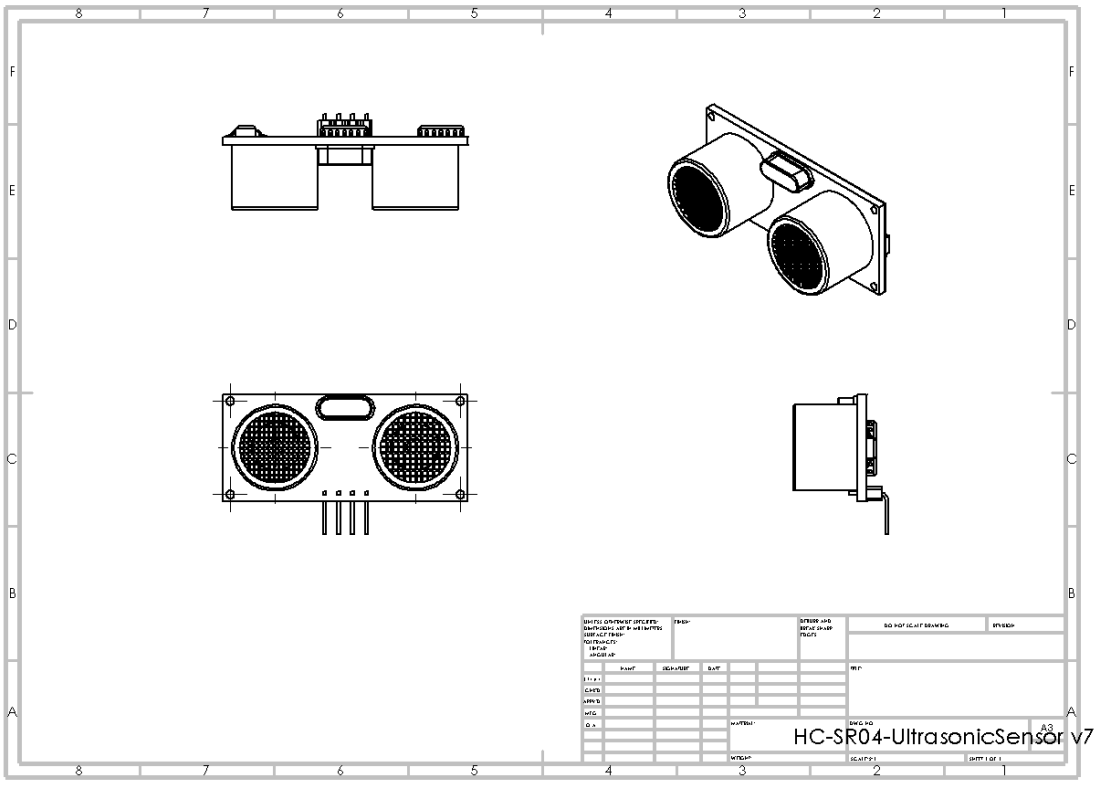



**SRF-05**

The SRF-05 is an ultrasonic sensor which is often used to replace the widely used HC-SR04 uses SONAR to determine the distance of an object just like the bats do. It offers non-contact range detection with high accuracy and stable readings in an easy-to-use package with range of 2 cm to 400 cm. The operation is not affected by sunlight or black material, although acoustically, soft materials like cloth can be difficult to detect. It comes complete with ultrasonic transmitter and receiver module.

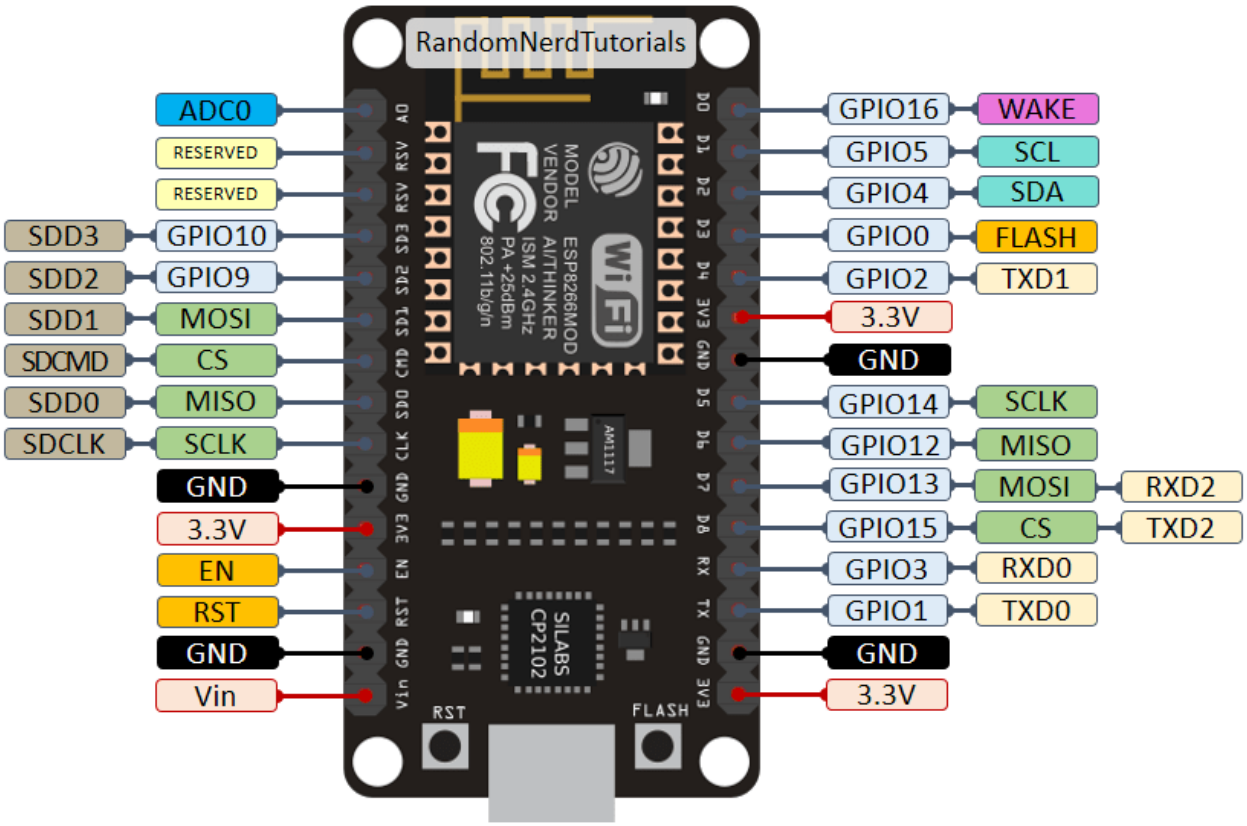



**SRF-05 Sensor**



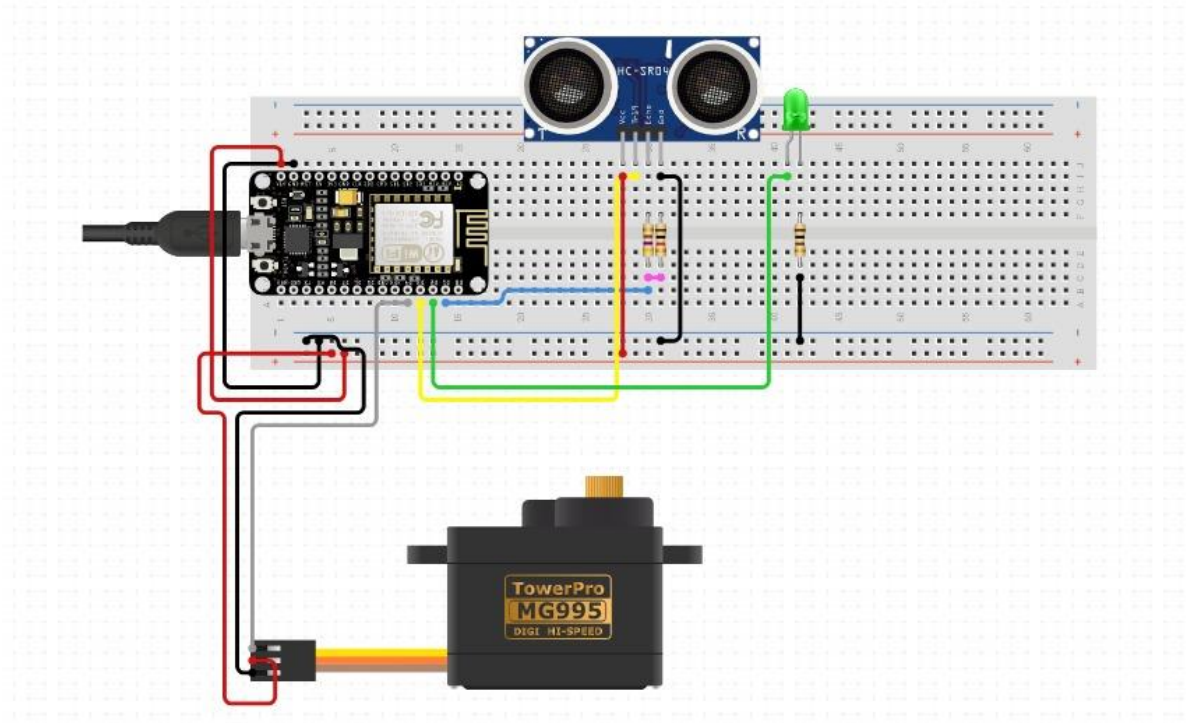

## Electrical Design

ESP8266 NodeMCU (fig.5) stands for Node Microcontroller Unit. It uses an open-source Lua-based firmware that is compatible with code used in Arduino, with addition of respective libraries. This is a microcontroller with wiFi chip on-board and thus suitable for IoT prototyping applications. The module that runs this firmware is ESP-12E and that module is based on 32-bit ESP8266 MCU. It supports up to 2.4 GHz WiFi and can handle WPA/WP2 encryption. The ESP-12E comes with a programmer and a 3.3v SMPS unit. So, you do not need any external programmer to program this board and you can easily run this board directly on 5V from USB connected to computer. This microcontroller is 3.3V, therefore applying 5V to its pins will cause damage. The high voltage input must be through its microUSB port. Only 11 of 16 pins are usable, pins 0-5 and 12-16. The ‘rst’ button must be clicked before uploading a sketch into it.

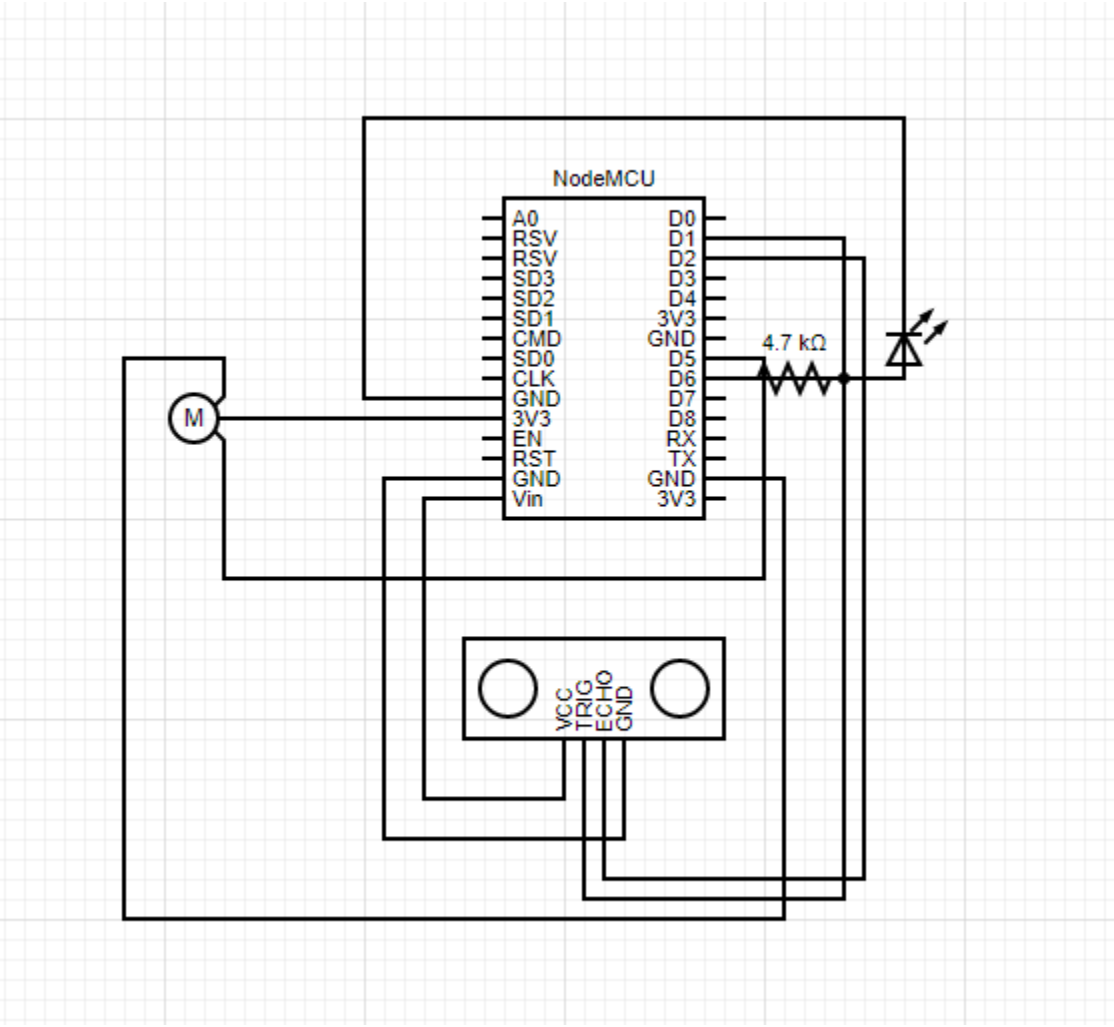



(fig.5 Esp8266 pin layout)

**Circuit Diagram**

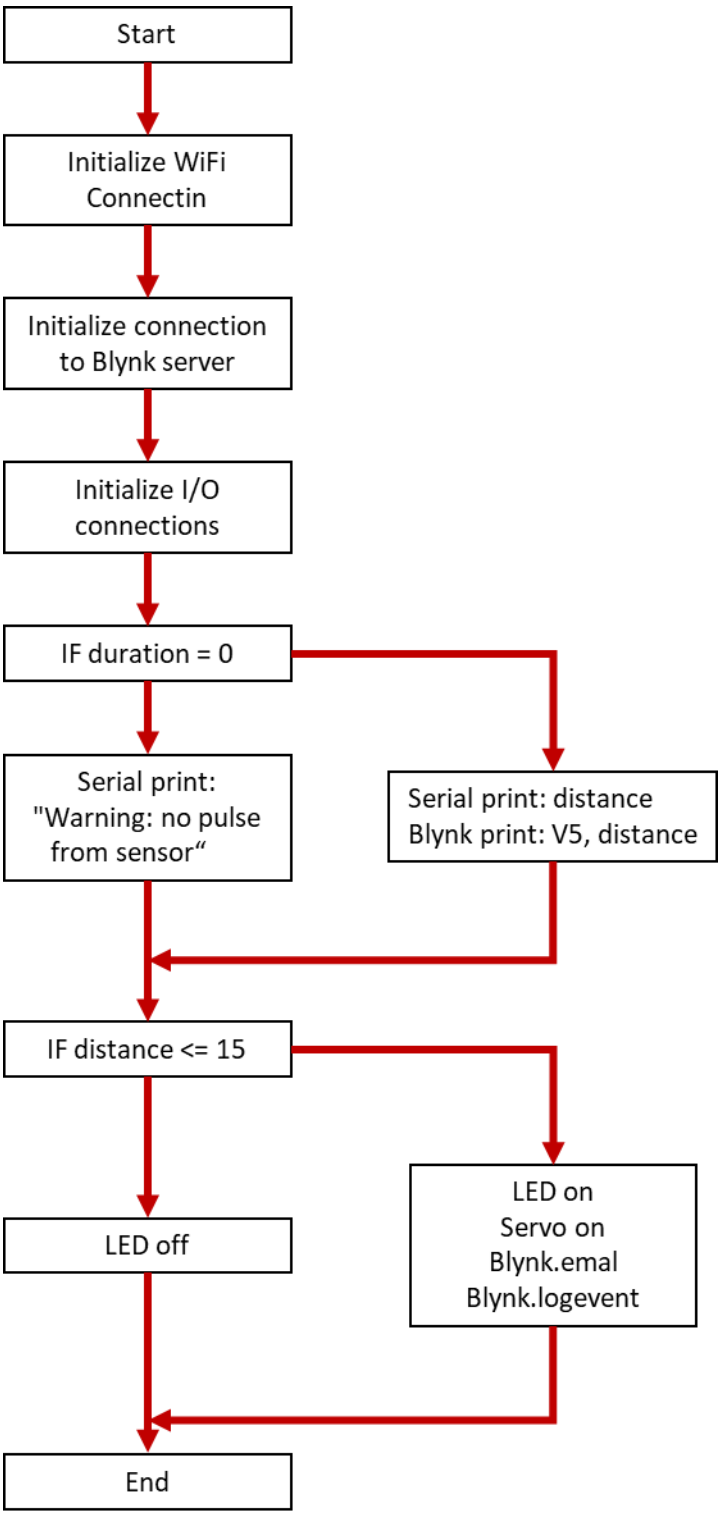


**Schematic Diagram**



## Control Design

**Flow chart**

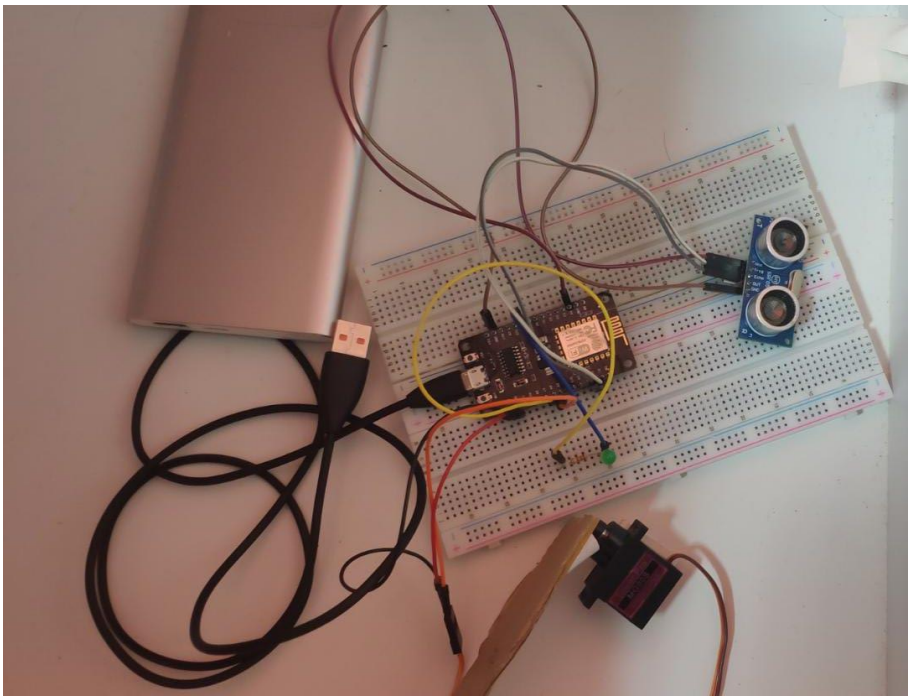


**Flow chart explanation**

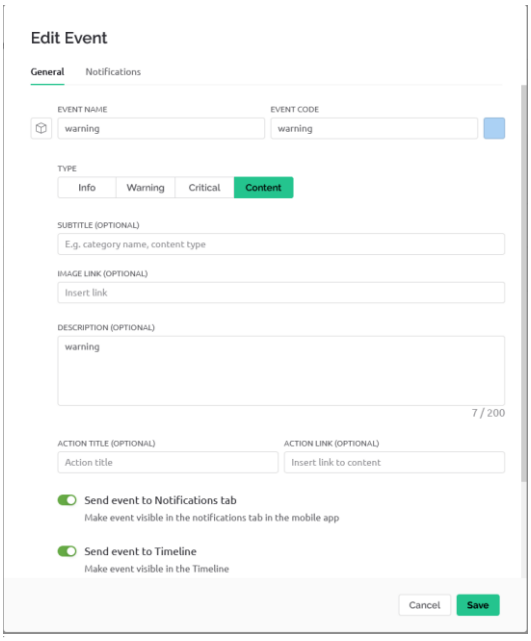
When the circuit is powered, the esp8266 will immediately connect to the local wifi network based on the SSID preset in code. The microcontroller then connects to blynk server using a special authentication key as well as activates I/O pins to receive inputs from ultrasonic sensor and out actuation to LED and Servo motor. Loop is made to detect information from ultrasonic sensor, if 'duration' = 0, microcontroller will print a "no pulse" warning to the serial monitor, else if there is input from sensor, the microcontroller outputs that data to serial monitor and blynk virtual pin. Another loop is created right after to detect the event. If distance recorded by sensor is less than 15cm, LED is turned off and loop ends, else if distance is more than 15cm (ie: the drawer/safe was opened) LED will be turned on, servo will rotate, and an event will be sent to blynk cloud service to send user an email and push notification.

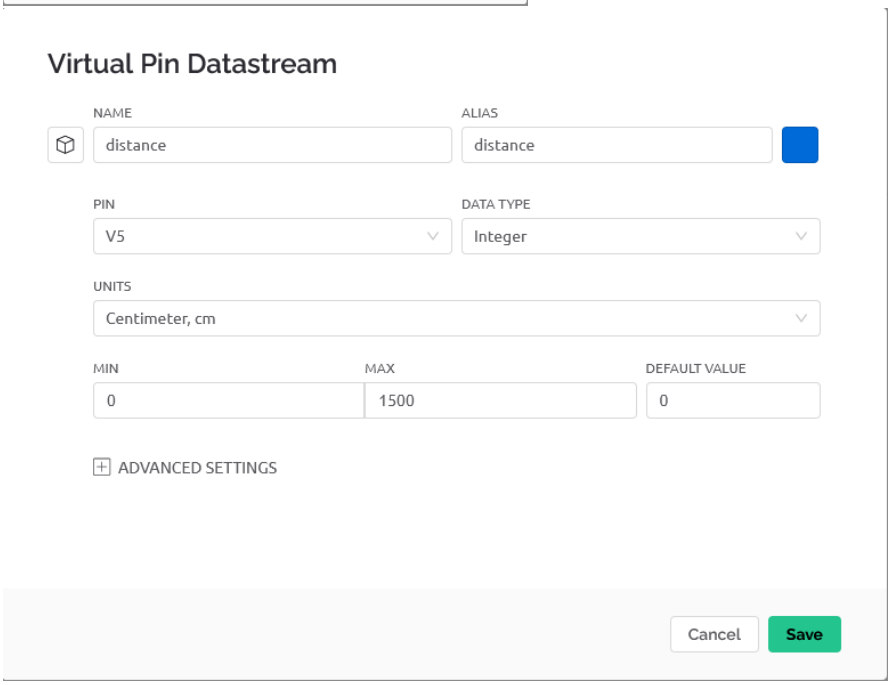
**Methodology**

1. Setup the circuit based on schematic diagram.

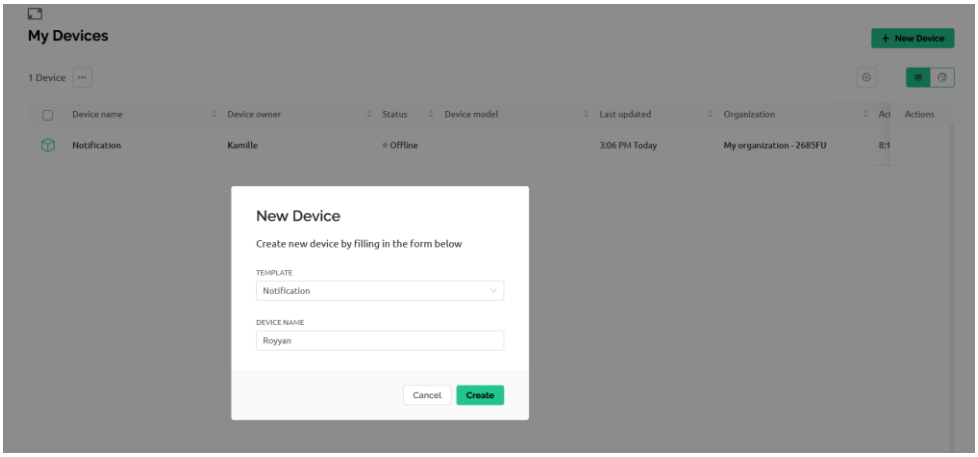


2. Open Arduino IDE>Tools>Manage Libraries; and install ESP8266 Library, Blynk Library.
3. Open File>Preferences and add esp8266 dependencies.
4. Open Tools>Boards>Generic ESP8266 module.
5. On blynk.io create a new template and set up DataStream and events tab.




6. On blynk.io create a new device based on previous template.



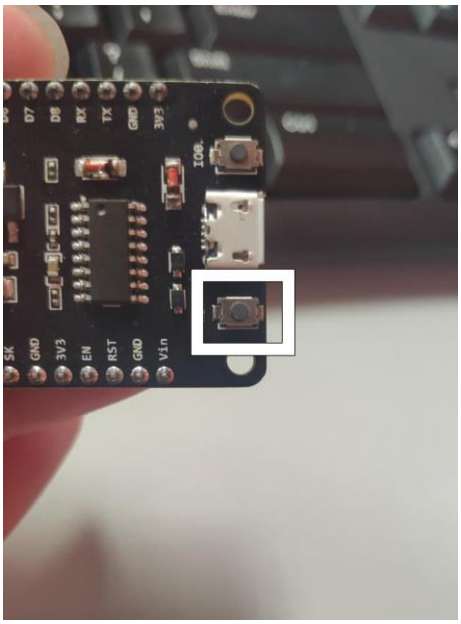

7. Paste blynk.io authentication details in the IDE.

FIRMWARE CONFIGURATION

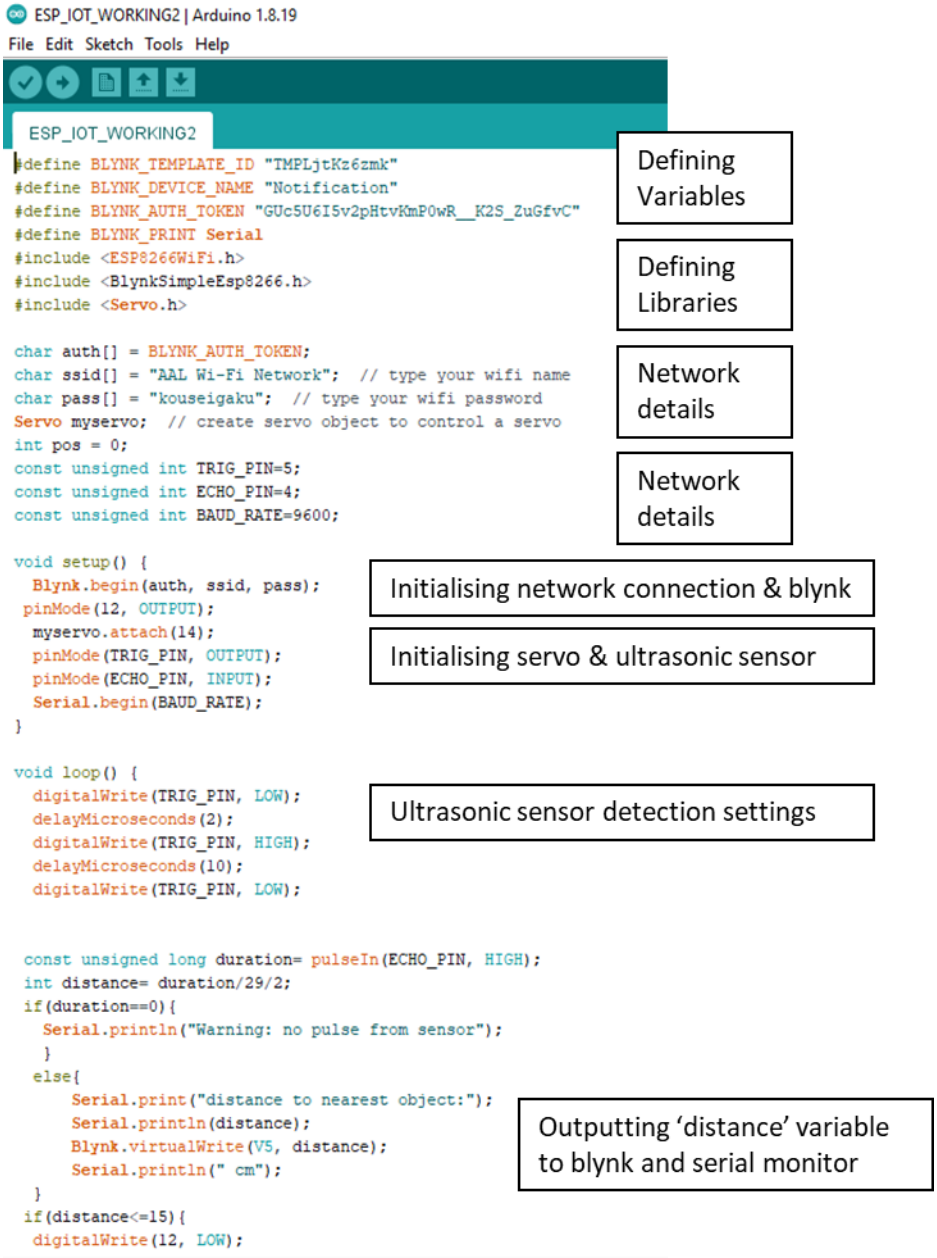
```
#define BLYNK_TEMPLATE_ID "TMPLjtKz6zmk"
#define BLYNK_DEVICE_NAME "Notification"
```

Template ID and Device Name should be included at the top of your main firmware

8. Click the ‘RST’ button on microcontroller and click upload on the IDE.



**Code**


ESP_IOT_WORKING2 | Arduino 1.8.19
File Edit Sketch Tools Help
ESP_IOT_WORKING2
#define BLYNK_TEMPLATE_ID "TMPLjtKz6zmk"
#define BLYNK_DEVICE_NAME "Notification"
#define BLYNK_AUTH_TOKEN "GUc5U6I5v2pHtvKmP0wR__K2S_ZuGfvC"
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <Servo.h>
char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "AAL Wi-Fi Network"; // type your wifi name
char pass[] = "kouseigaku"; // type your wifi password
Servo myservo; // create servo object to control a servo
int pos = 0;
const unsigned int TRIG_PIN=5;
const unsigned int ECHO_PIN=4;
const unsigned int BAUD_RATE=9600;
void setup() {
Blynk.begin(auth, ssid, pass);
pinMode(12, OUTPUT);
myservo.attach(14);
pinMode(TRIG_PIN, OUTPUT);
pinMode(ECHO_PIN, INPUT);
Serial.begin(BAUD_RATE);
}
void loop() {
digitalWrite(TRIG_PIN, LOW);
delayMicroseconds(2);
digitalWrite(TRIG_PIN, HIGH);
delayMicroseconds(10);
digitalWrite(TRIG_PIN, LOW);
const unsigned long duration= pulseIn(ECHO_PIN, HIGH);
int distance= duration/29/2;
if(duration==0){
Serial.println("Warning: no pulse from sensor");
}
else{
Serial.print("distance to nearest object:");
Serial.println(distance);
Blynk.virtualWrite(V5, distance);
Serial.println(" cm");
}
if(distance<=15){
digitalWrite(12, LOW);
Defining Variables
Defining Libraries
Network details
Network details
Initialising network connection & blynk
Initialising servo & ultrasonic sensor
Ultrasonic sensor detection settings
Outputting 'distance' variable to blynk and serial monitor

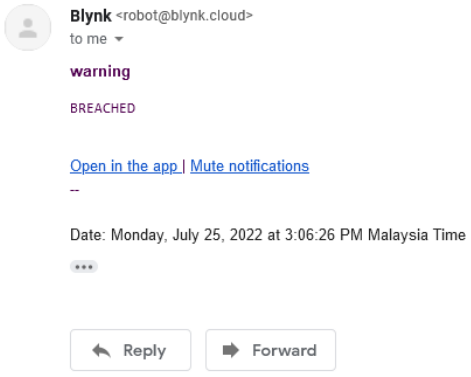
```
 }
 else {
  digitalWrite(12, HIGH);
  Blynk.email("xplatformgamer@gmail.com",
  "BREACHED", "Your safe has been opened");
  Blynk.logEvent("warning","BREACHED");
 for(pos=0;pos<=180;pos++){
myservo.write(pos);
delay(15);
}
  delay(5000);

 for(pos=180;pos>=0;pos--){
myservo.write(pos);
delay(15);
}

  }
 delay(100);
 }
```

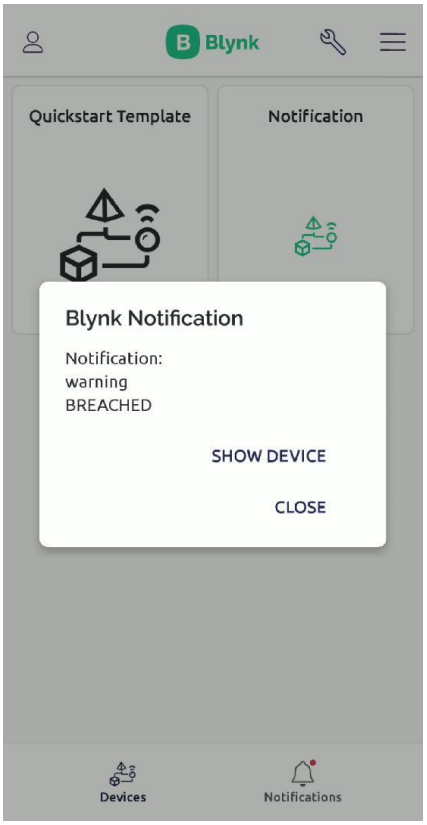
Blynk email presets

Referencing 'warning' event in blynk console

Servo movement controls

## Performance Analysis and Discussion




The performance of the completed system is satisfactory, it was tested by placing inside a cubbard and closing the lid before the microcontroller initializes. When the cubbard is opened, a push notification on android phone is immadiately received through blynk app, and an email followed suit within a span of 30 seconds. I believe this system serves its limited purpose well and only requires improvements in form factor. A gsm module could also be added in-case wiFi insinde the home gets shut down. The purpose of this system is for the user to be able to call authorities if it is triggered while the home owners are outside. The system is set detect event when distance is more than 15cm, so it is not sensitive to false

triggering. Sensitivity can also be adjusted by manipulating the event in blynk console. Design considerations were made for added security such as using a powerbank connected to power outlet. In case the power gets unplug, the system will function for some time.

As AI based CCTVs are still expensive and experimental, this project provides the most cost effective method of IOT based home security, the system can be placed in cubbards with valuables in them or even on doors with come code modifications. Allowing for security on entry points. The weakness is if a robber enters the hoe throgh unconventional means such as windows. A majority of home robberies happen by breaking the door and cabinet locks while the owner is away, so said system should be good enough for general security.

## Conclusion

In this project, we proposed a simple solution for automated home security system based on ESP8266 microcontroller and blynk.io. The end product fabricated is cost effective, small, and easy to work with.

Moreover, the proposed system uses blynk cloud services API, allowing access to email without difficult workarounds. This is because major mail services use high levels of spam protection, and mail from simple SMTP services will be sent to spam, forcing developers of these systems to get premium SMTP services. The advantages of home automation is highlighted; smart processing, low effort, and low maintenance once set up. However, it comes at cost of data privacy as external processing servers such as blynk is needed to perform advanced functions. In conclusion the objective of fabricating home security system based on IoT is successful, and systems such as proposed in this project can be repurposed for other IoT services such as water level warning systems.

## Video URL

https://youtu.be/M3HDAHMt9Vs