ADVANCED ACADEMIC CENTER

A CENTER FOR INTER-DISCIPLINARY RESEARCH
2021-22

TITLE

“SMART KEY CHAIN”

SUPERVISED BY

GANDRA PADMIKA

GOKARAJU RANGARAJU
INSTITUTE OF ENGINEERING AND TECHNOLOGY
AUTONOMOUS

Advanced Academic Center

(A Center For Inter-Disciplinary Research)

This is to certify that the project titled

“SMART KEY CHAIN”

is a bonafide work carried out by the following students in partial fulfilment of the requirements for Advanced Academic Center intern, submitted to the chair, AAC during the academic year 2020-21.

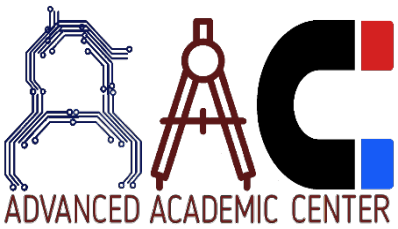
NAME	ROLL NO.	BRANCH
CHILKURI ABHINAV REDDY	21241A12D9	IT
MANNE HARSHITH	21241A05N8	CSE
VEMULA SATWIK	21241A6664	CSM
B.MANOJ TEJA	21241A0306	MECH

This work was not submitted or published earlier for any study

Dr/Ms./Mr.
Project Supervisor

Dr.B.R.K.Reddy
Program Coordinator

Dr.Ramamurthy Suri
Associate Dean,AAC



ACKNOWLEDGEMENTS

We express our deep sense of gratitude to our respected Director, Gokaraju Rangaraju Institute of Engineering and Technology, for the valuable guidance and for permitting us to carry out this project.

With immense pleasure, we extend our appreciation to our respected Principal, for permitting us to carry out this project.

We are thankful to the Associate Dean, Advanced Academic Centre, for providing us an appropriate environment required for the project completion.

We are grateful to our project supervisor who spared valuable time to influence us with their novel insights.

We are indebted to all the above mentioned people without whom we would not have concluded the project.

ABSTRACT:

This is the perfect project for those who frequently place their keys here and there and are unable to find it when needed the most. So here we bring Smart Keychain, a smart companion for your keys. No worries where are keys are just take out your smart phone and give a trigger to your keys, they will automatically says where they are. We made this project using ESP8266 development board.

INTRODUCTION:

Many times we misplace our keys and go searching for them everywhere ,and after a long search, we end up finding them with much distress. So to find our keychain immediately in less time here is a IoT-based Smart Key Chain finder.

In case if you can't find your keys and you remember that you have attached an IoT keychain to your keys, so you take out your phone and open Chrome and open your Keychain Webpage. Then you click on the toggle button(search), and in moments, you hear a beep sound coming from your keychain and with this, you can easily track your keys.

In this project , we build a simple IoT-based Smart Key Chain just using ESP8266-01, Buzzer, and Battery, LED.

PROJECT WORKFLOW

In recent times we misplace our keys and go searching for them everywhere in the house, and after a long search, we end up finding them with much distress.This is the perfect project for those who frequently place their keys here and there and are unable to find it when needed the most. So here we bring Smart Keychain, a smart companion for your keys. Now, the obvious solution here is to place your keys in their right place, but as engineers, what's the fun in doing that. So, we built a simple IoT-based Smart Key Chain just using ESP8266-01, Buzzer,USB cable, connecting wires and LED and also using arudino and connecting them with the help of tools mentioned above .Now in case if you can't find your keys and you remember that you have attached an IoT keychain to your keys, so you take out your phone and open Chrome and open your Keychain Webpage. Then you just reload the web page and in moments, you hear a beep sound coming from your keychain and with this, you can easily track your keys.

HARDWARE USED

ESP8266

The ESP8266 is a low-cost Wi-Fi microchip, with a full TCP/IP stack and microcontroller capability. The ESP8266 module enables microcontrollers to connect to 2.4 GHz Wi-Fi, using IEEE 802.11 bgn. It can be used with ESP-AT firmware to provide Wi-Fi connectivity to external host MCUs, or it can be used as a self-sufficient MCU by running an RTOS-based SDK. The module has a full TCP/IP stack and provides the ability for data processing, reads and controls of GPIOs. Given its low cost, small size and adaptability with embedded devices, the ESP8266 is now used extensively across IoT devices.

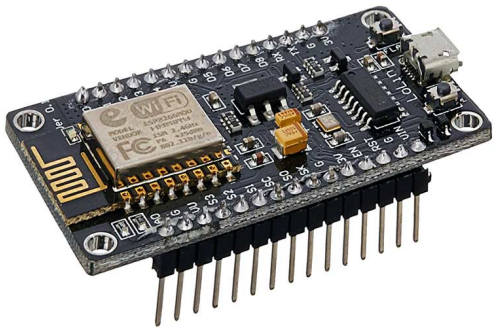


Fig: ESP8266

BUZZER

A buzzer is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke. This buzzer can be used by simply powering it using a DC power supply ranging from 4V to 9V. A simple 9V battery can also be used, but it is recommended to use a regulated +5V or +6V DC supply. The buzzer is normally associated with a switching circuit to turn ON or turn OFF the buzzer at required time and required interval.

Fig: Buzzer

SOFTWARE USED

ARDUINO IDE

For Windows, macOS, and Linux, the Arduino Integrated Development Environment (IDE) is a cross-platform program that uses C and C++ functions. Programs are created and uploaded using it to boards that are compatible with Arduino.

The Arduino development environment is connected to the Arduino board through USB when it is attached to a computer (IDE). The user creates the Arduino code in the IDE, uploads it to the microcontroller, and the code is then executed by the microcontroller while interacting with inputs and outputs like sensors, motors, and lights.

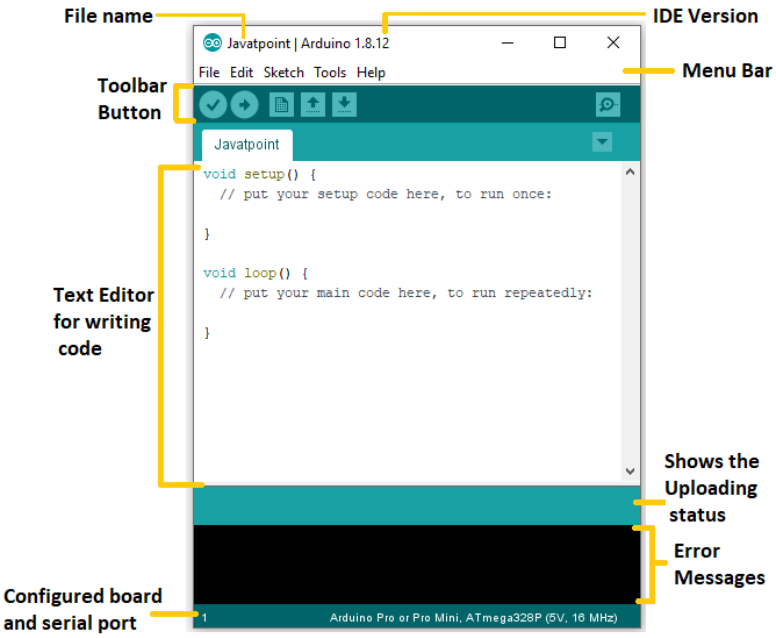


Fig: Arudino IDE

By selecting the magnifying glass icon on the IDE's top right side or from the tools menu, the Arduino sersial monitor may be launched. The main purpose of the serial monitor is to allow computer users to communicate with the Arduino board. The serial monitor is used mainly for interacting with the Arduino board using the computer and is a great tool for real-time monitoring and debugging.

CODE

The Working of esp8266

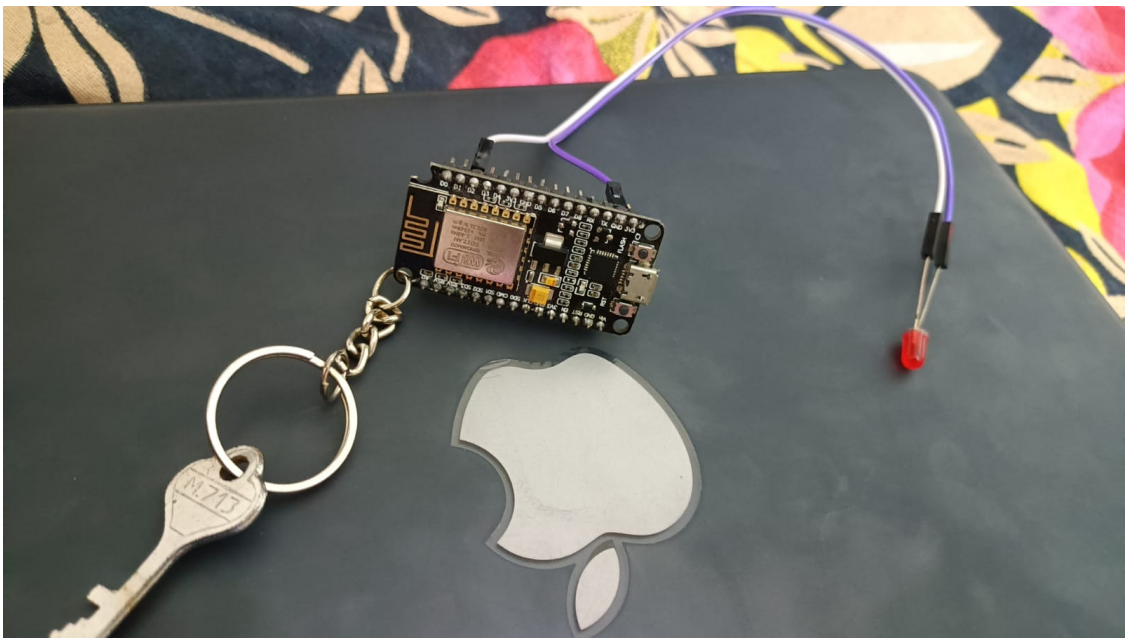
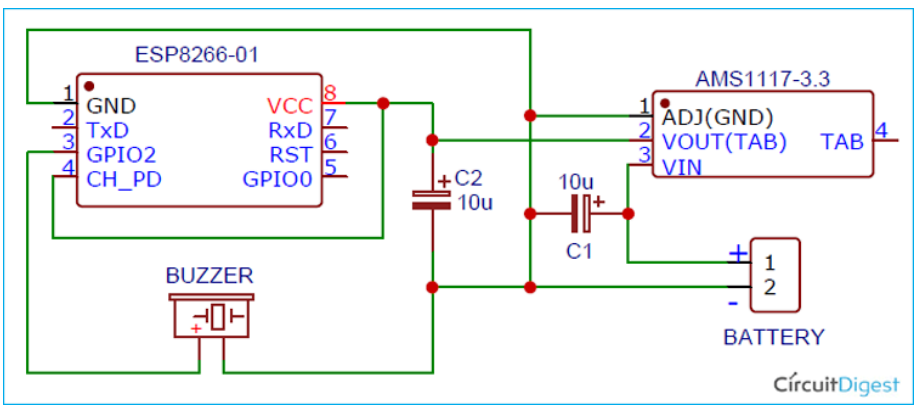


Fig: ESP8266 and LED connection

Connections of esp8266 and buzzer

Buzzer: +ve to pin D2
-ve to ground

WiFi Module- ESP8266:



```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>

//ESP Web Server Library to host a web page
#include <ESP8266WebServer.h>

//---------------------------------------------------------------
//Our HTML webpage contents in program memory
const char MAIN_page[] PROGMEM = R"=====(
<!DOCTYPE html>
<html>
<body>
<center>
<h1>WiFi LED on off demo: 1</h1><br>
<button href="ledOn">FIND DEVICE</button><br>
<hr>
</center>

</body>
</html>
)=====";



#define LED D2

//SSID and Password of your WiFi router
const char* ssid = "Testing";
const char* password = "password";

//Declare a global object variable from the ESP8266WebServer class.
ESP8266WebServer server(80); //Server on port 80

//===============================================================
// This routine is executed when you open its IP in browser
//===============================================================
void handleRoot() {
 Serial.println("You called root page");
 String s = MAIN_page; //Read HTML contents
 server.send(200, "text/html", s); //Send web page
}
int flag = 0;
```

```
void handleLEDon() {
  Serial.println("LED on page");
  digitalWrite(LED,HIGH);
  delay(2000);
  digitalWrite(LED,LOW);
// server.send(200, "text/html", "LED is ON"); //Send ADC value only to client ajax request
  String s = "RELOAD TO FIND THE DEVICE";
  server.send(200, "text/html", s); //Send web page
}



void setup(void){
  Serial.begin(115200);

  WiFi.begin(ssid, password);     //Connect to your WiFi router
  Serial.println("");

  //Onboard LED port Direction output
  pinMode(LED,OUTPUT);
  //Power on LED state off
  digitalWrite(LED,LOW);

  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  //If connection successful show IP address in serial monitor
  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());  //IP address assigned to your ESP

  server.on("/", handleLEDon);      //Which routine to handle at root location. This is display
page
  server.on("/ledOn", handleLEDon); //as Per  <a href="ledOn">, Subroutine to be called
```

```
server.begin();                 //Start server
  Serial.println("HTTP server started");
}
//=============================================================
//                    LOOP
//=============================================================
void loop(void){
  server.handleClient();          //Handle client requests
}
```

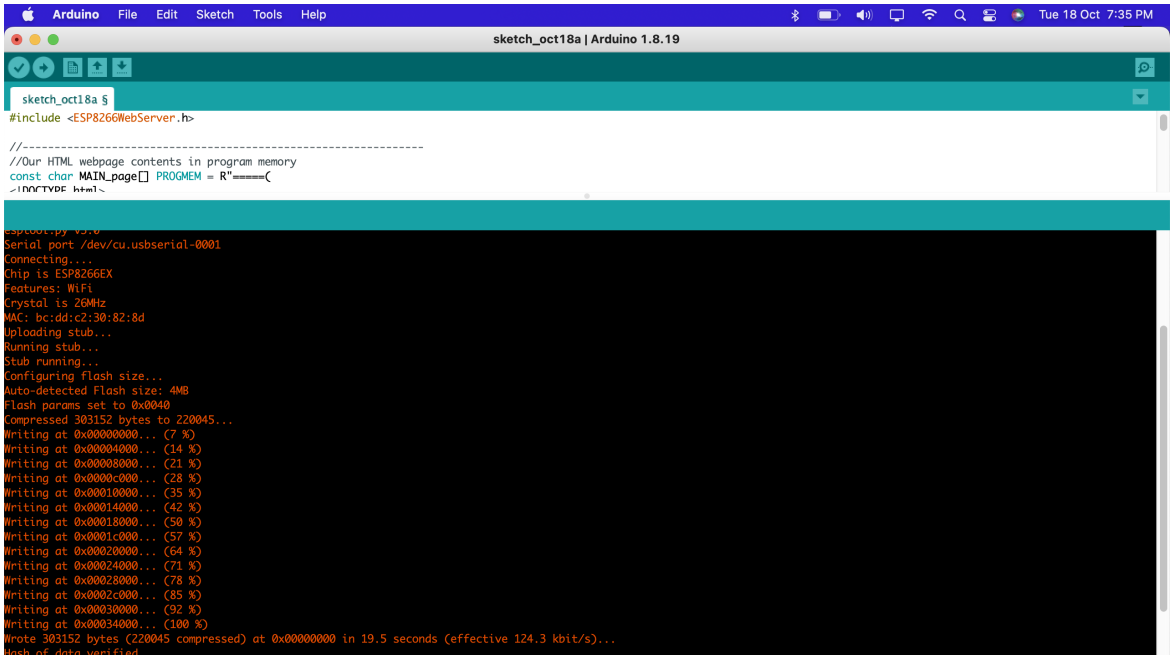


Fig: The output and Uploading the code to esp8266

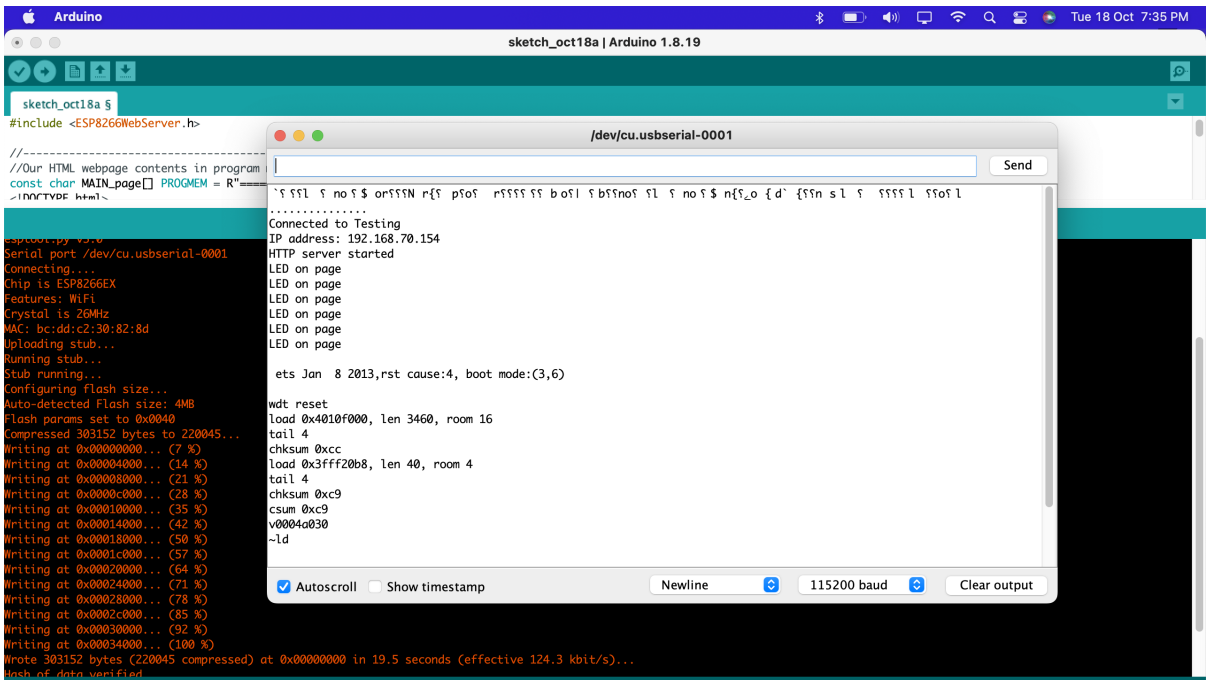


Fig: serial monitor

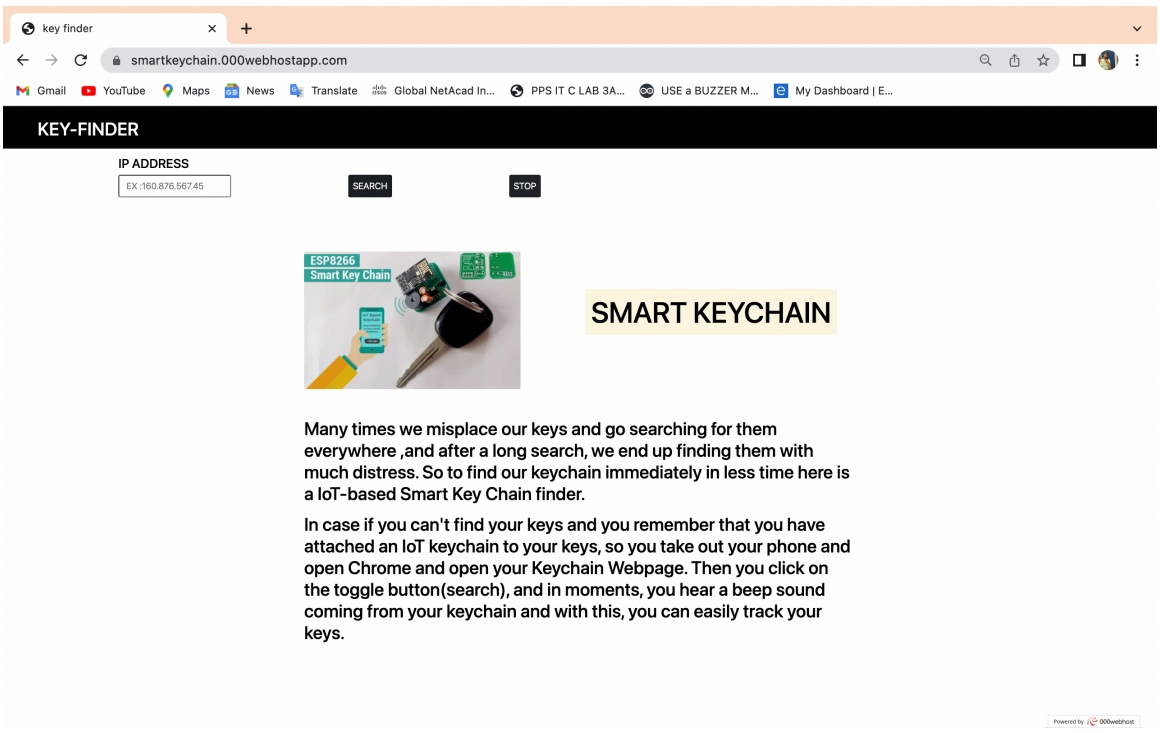


Fig: smart key chain webpage

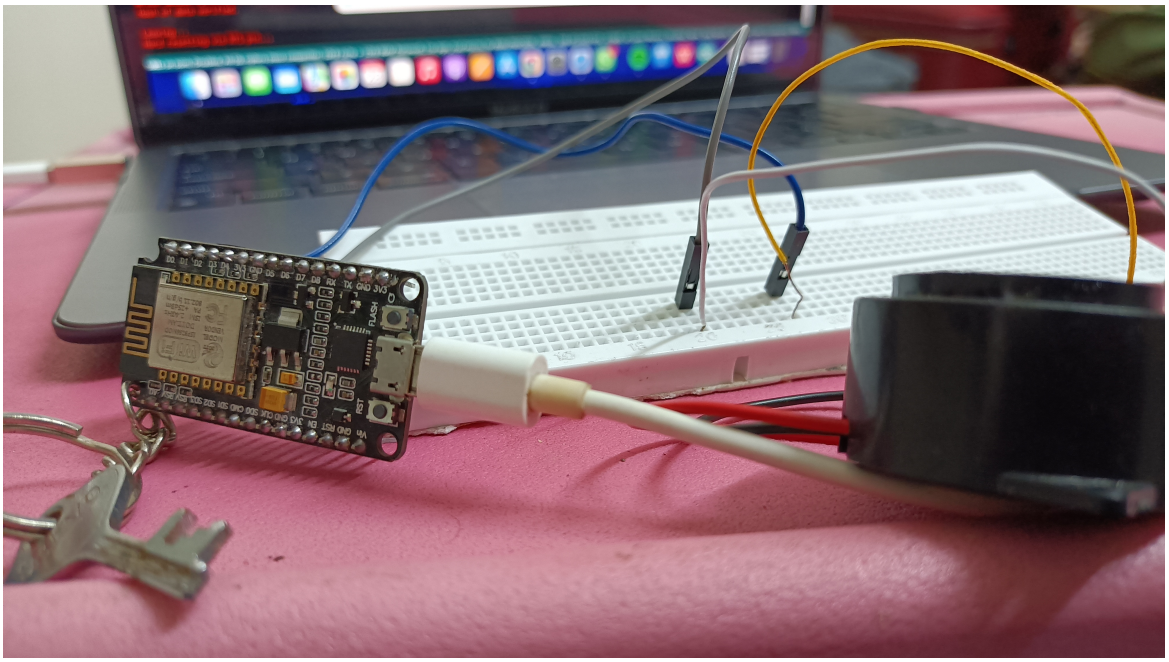


Fig: ESP8266 and buzzer connections

CONCLUSION

Through this project, we have proposed a smart key chain we conclude that In case if you can't find your keys and you remember that you have attached an IoT keychain to your keys, so you take out your phone and open Chrome and open your Keychain Webpage. Then you click on the toggle button(search), and in moments, you hear a beep sound coming from your keychain and with this, you can easily track your keys.

FUTURE DEVELOPMENTS

As the iot based smart key chain is introduced which can be implemented using a webpage application it can accessible by the mobile app also. An Additional feature of displaying the location tracker of the user can be added to this device. The smart keychain can be modified in small size .

REFERENCES

1. https://circuitdigest.com/microcontroller-projects/diy-iot-based-key-chain-finder-
2. https://www.instructables.com/DIY-Smart-Keychain/
3. https://www.google.com/search?q=iot+smart+key+chain&sxsrf=ALiCzsZkbk71kex8hMmk0Y_iQqPdi2iA7A:1667052625929&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjBm9z8zoX7AhX77jgGHfPAAJsQ_AUoAnoECAIQBA&biw=1280&bih=689&dpr=2#imgrc=G0NQZvly98tEnM
4. https://circuitdigest.com/fullimage?i=circuitdiagram_mic/Smart-key-Finder-Circuit-Diagram.png
5.