Home security system

IOE Mini Project

Submitted in complete fulfillment of the requirements of the degree for the

IOE (Mini Project) Lab

Submitted by

Ms. Anjali Punsi Roll No. **57**
Ms. Diya theryani Roll No. **66**
Mr. Ronak karia Roll No. **31**

Under the guidance of
Mrs. Charusheela Nehete

Department of Information Technology

Vivekanand Education Society's Institute of Technology

University of Mumbai

2023-2024

VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY (VESIT) Chembur, Mumbai 400074

CERTIFICATE

This is to certify that **Ms. Anjali punsi (57)**, **Ms. Diya Theryani (66)**, and **Mr. Ronak karia(27)** have satisfactorily carried out the project work, under the head - Internet of Everything Lab at Semester VII of BE in Information Technology as prescribed by the MumbaiUniversity.

Prof. Guide Name	**External Examiner**
Charusheela nehete	
H.O.D	**Dr.(Mrs.) J.M.NairPrincipal**
Dr.(Mrs.) Shalu Chopra	
Date: 16/10/2023	**Place: VESIT, Chembur**

Declaration

I declare that this written submission represents my ideas in my own words and where other's ideas or words have been included, I have adequately cited and ref- erenced the original source. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

- - - - - - - - - - -
(Signature)

Anjali punsi - 57
- - - - - - - - - - -
(Signature)

Diya theryani - 66

- - - - - - - - - - -
(Signature)

Ronak karia - 31

Contents

List of Figures

List of Tables

Abstract

The Internet of Things (IoT) describes the network of physical objects—“things”—thatare embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet. IoT creates an ecosystem among devices which makes it accessible remotely and things in IoT represents the devices like sensors, microcontrollers, and mobile phones that’s connected to a wireless network In this case the Wi-Fi module, Node MCU (esp82),act as a gateway to connect the home appliances. Additionally, this Internet based Home security system has been integrated with a Ultrasonic sensor which makesthe system work when there is any human detected by the sensor. The system hasbeen designed accordingly.

Keywords - ***IoT, Wi-Fi Model, ESP32, Home Appliances, UltrasonicSensor***

Chapter 1 Introduction

1.1 Introduction

Nowadays, we have remote controls for our television sets and other electronic systems, which have made our lives real easy.Have you ever wondered about home security which would give the facility of controlling tube lights, fans and other electrical appli- ances at home? Off-course, Yes! But, are the available options cost-effective? If the answer is No, we have found a solution to it. We have come up with a new systemcalled home security.

Home security is the automatic control of electronic devices in your home. These devices are connected to the Internet, which allows them to be controlled remotely. With home security, devices can trigger one another so you don’t have to con- trol them manually via an app or voice assistant. Home security makes life more convenient and can even save you money on heating, cooling and electricity bills.

1.2 Literature Survey

1) Literature Survey for IoT-based Smart Home security: A Comparative Analysis Comparative Analysis Published in: 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)
2) Santoso, Freddy K., and Nicholas CH Vun. ”Securing IoT for smart home systems.” Securing IoT for smart home system Published in 2015 international symposium on consumer electronics (ISCE). IEEE, 2015

1.3 Objectives

The main objective of this project is to build a smart home device which can be usedto control the home appliances via the internet. And push the data to aws cloud The home security device that you build can be integrated with almost all the home appliances. A smart home is a home-like environment that possesses ambient intelli- gence and automatic control, which allows it to respond to the behavior of residents and provide them with various facilities.

1.4 Hardware and Software requirements

Hardware Requirements:
1) ESP32 board
2) Ultrasonic Sensor
3) LED

Software Requirements:
1) AWS IoT Core
2) PC with i3 processor or more
3) Arduino software

1.5 Wireless Technology Used

- MQTT for data collection
- WiFi - for connectivity with AWS

1.6 Cost Estimation

Components	**Price (in Rs.)**
ESP32	550
Led Sensor	19
Ultrasonic Distance Sensor	50
Total estimated cost	619

Table 1.1: Estimated costs for the system

Chapter 2

IoT System Design

2.1 Purpose and Requirements

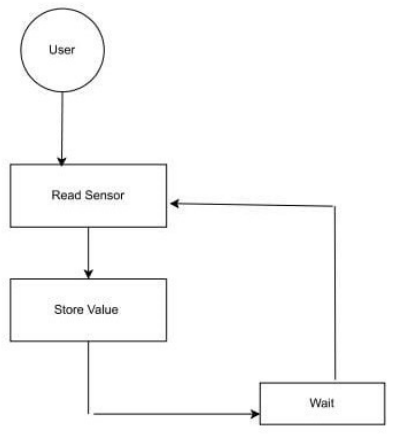
The main purpose of this project is to build a smart home device which can be usedto control the home appliances via the internet. The home security device thatwe built can be integrated with almost all the home appliances. A smart home is a home-like environment that possesses ambient intelligence and automatic control, which allows it to respond to the behavior of residents and provide them with various facilities.

For this purpose, sensors like the ultrasonic sensor to sense the presence of the human in room, the led to on the light whenever the person is present in the room. This is prepared using the esp32 board. The network helps with obtaining data through the Internet of Things. The whole system consists of nodes which are situated at different parts of the board. The data or the value (distance from sensor to object) is collected by the board and uploaded to the cloud i.e. AWS

2.2 System Design

2.2.1 Process Model Specification

The process specification shows that the sensors are read after fixed intervals and the distance from object to ultrasonic sensor are stored.



Process Model Specification for home Automation

Figure 2.1: Process Model Specification

2.2.2 Domain Model Specification

In this domain model the physical entity is the environment which is being monitored. Hence, there is a virtual entity for the environment. Sensors connected to the single board mini computer. resources are software components which can be either on device or network-resources.

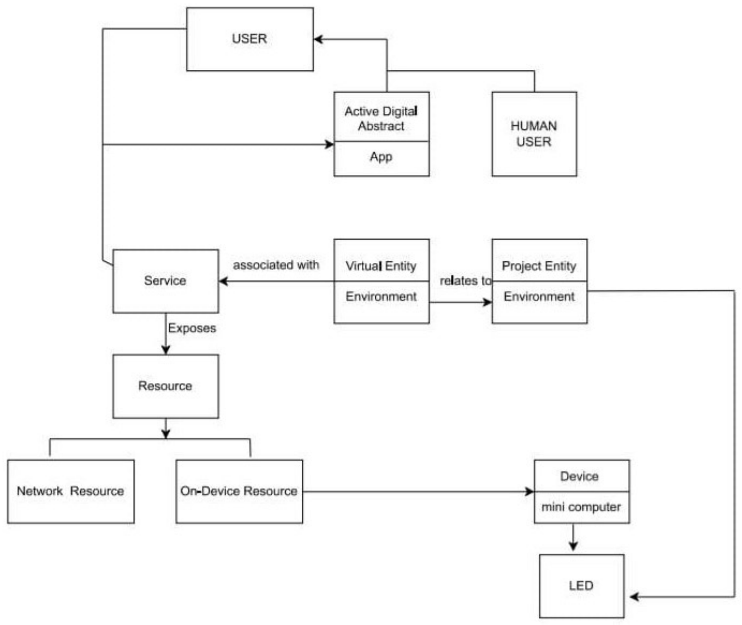


Figure 2.2: Domain Model

2.2.3 Information Model Specification

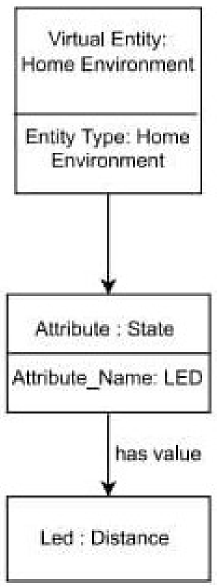

Virtual Entity: Home Environment
Entity Type: Home Environment
Attribute : State
Attribute_Name: LED
has value
Led : Distance

Figure 2.3: Information Model Specification

2.2.4 Service Specification

Services include only the controller service. The controller service runs as a native service on the device and monitors the led distance every 10 second. The controller service calls the AWS MQTT service to store these measurements in the cloud.

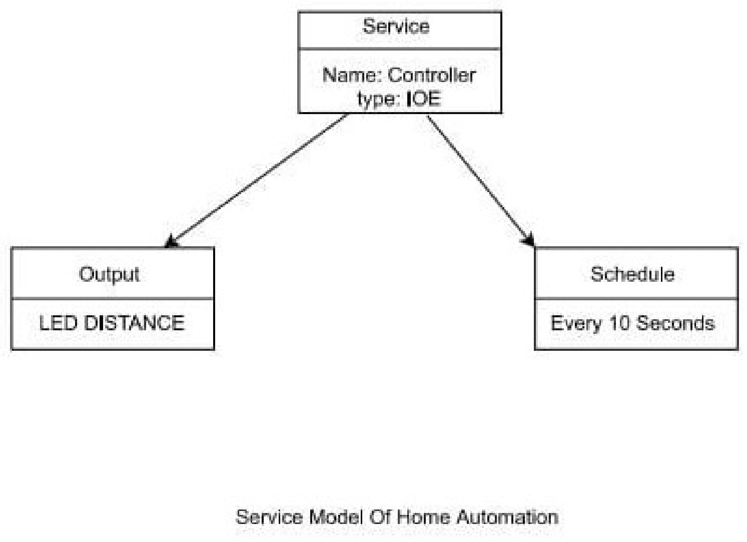


Figure 2.4: Service Specification

2.2.5 IoT Deployment Level Specification

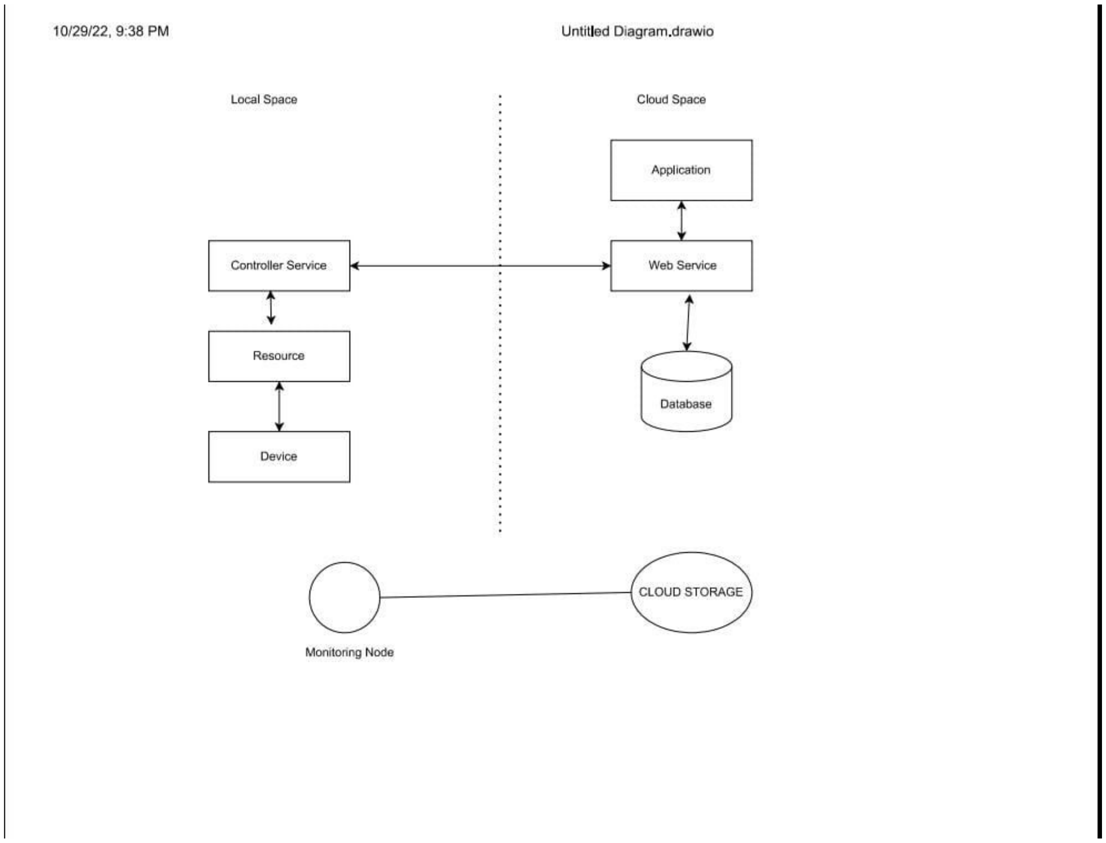


Figure 2.5: IoT Deployment Level Specification

2.2.6 Functional View Specification

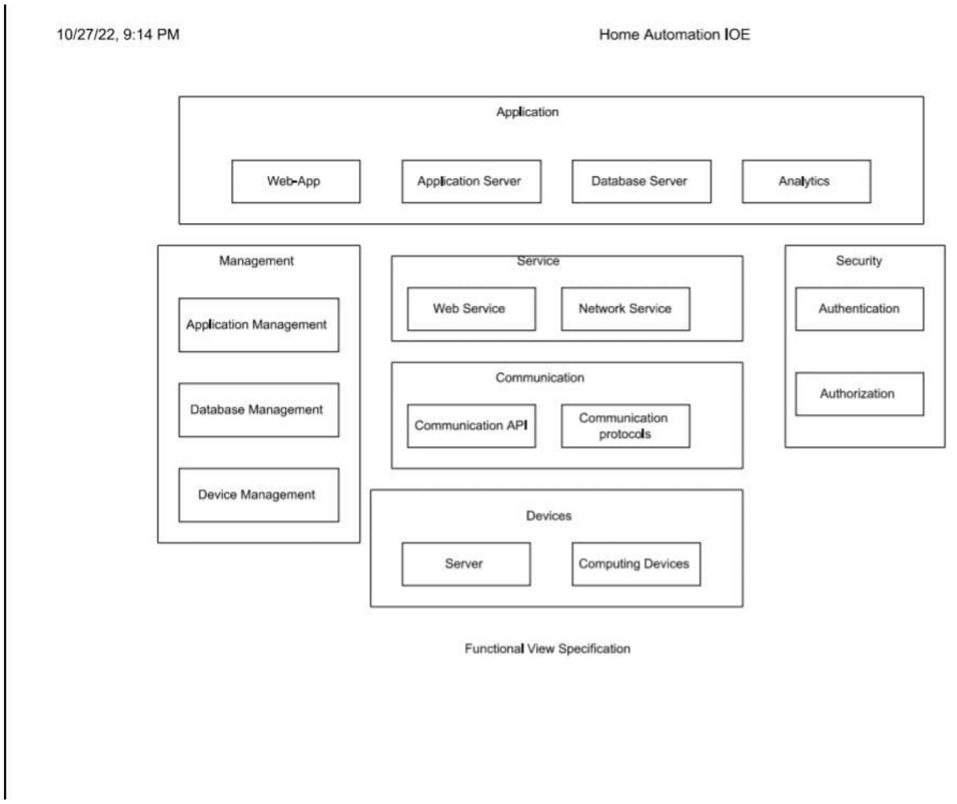


Figure 2.6: Functional View Specification

2.2.7 Operational View Specification

Native ervice	**Controller ervice**
Web App	Flask App
Web App	Web Server
Database Server	AWS Cloud torage
Observer	Cloud App, Veb App
Authentication	Web App
Authorization	Web App
Communication rotocol	Application: TTP
Computing evice	ESP32
Sensor	Light Sensor

Table 2.1: Mapping of functional groups with operational entities

2.2.8 Device and Component Integration

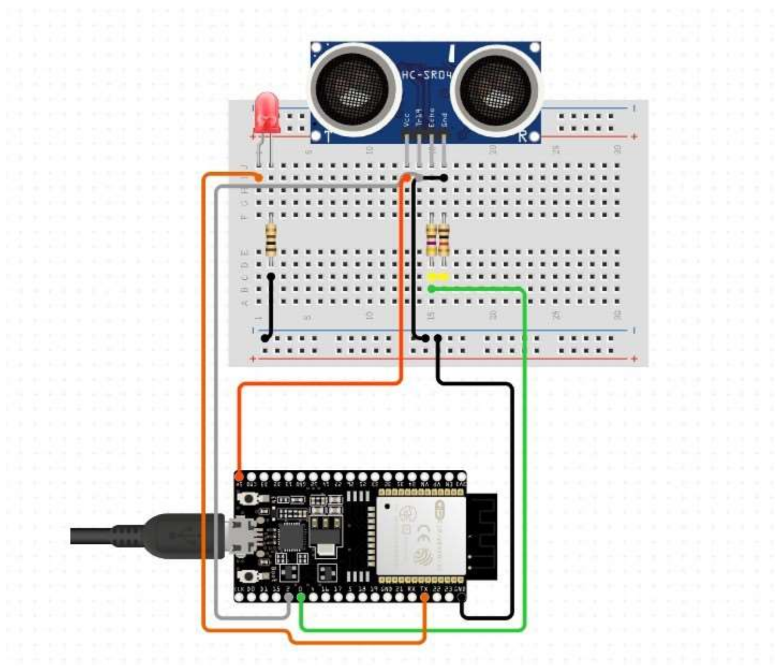


Figure 2.7: Device and Component Integration

2.3 Block Diagram

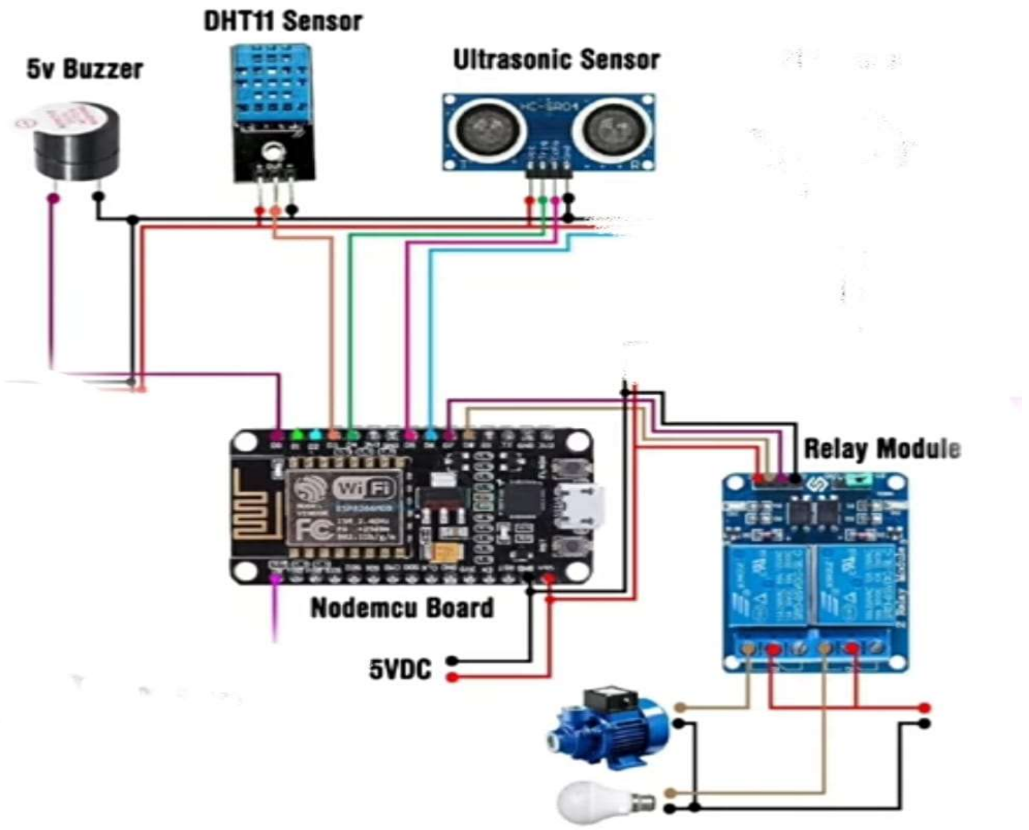


Figure 2.8: Block Diagram

Chapter 3

Implementation and ResultAnalysis

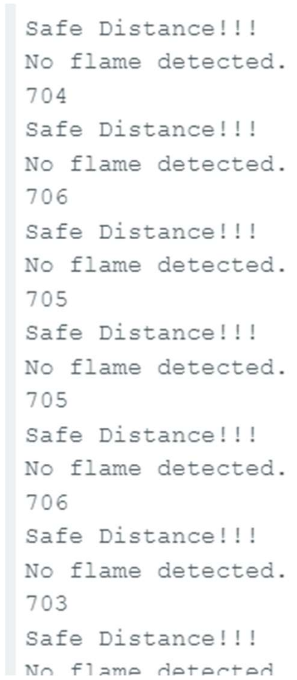
3.1 Interfacing with Real-time Data

3.1.1 Physical Interfacing with Real-time Data

Connect the sensors to the board properly. We connect the board to the computer using the Micro USB cable to upload the code from Arduino IDE. Install the required Arduino libraries. Write the following code that collects data from the respective sensors and prints the same on the serial monitor.

```
#include  "secret.h"
#include <WiFiClientSecure.h>#include <PubSubClient.h> #include
<ArduinoJson.h> #include  "WiFi.h"

#define lamp 25
#define trigPin 33
#define echoPin 32


void setup()
{
Serial.begin(115200); connectAWS();
pinMode (lamp, OUTPUT); pinMode(trigPin, OUTPUT);pinMode(echoPin, INPUT);
digitalWrite(lamp, LOW);

}
void loop()
{
digitalWrite(trigPin, LOW);delayMicroseconds(2);
digitalWrite(trigPin, HIGH); delayMicroseconds(10); digitalWrite(trigPin,  LOW);
duration = pulseIn(echoPin, HIGH);distance = (duration/2) / 29.1;
if (distance < 10)
{
digitalWrite(lamp,HIGH);
}
else
{
digitalWrite(lamp,LOW);
}
Serial.print(distance); Serial.println(" cm"); delay(500);
}
```

OUTPUT:



```
Safe Distance!!!
No flame detected.
704
Safe Distance!!!
No flame detected.
706
Safe Distance!!!
No flame detected.
705
Safe Distance!!!
No flame detected.
705
Safe Distance!!!
No flame detected.
706
Safe Distance!!!
No flame detected.
703
Safe Distance!!!
No flame detected
```

Figure 3.1: Interfacing with Real-time Data

3.1.2 Cloud Interfacing with Real-time Data

AWS IoT provides cloud services and device support that you can use to implement IoT solutions. AWS IoT Core provides the services that connect your IoT devices tothe AWS Cloud so that other cloud services and applications can interact with your internet-connected devices.

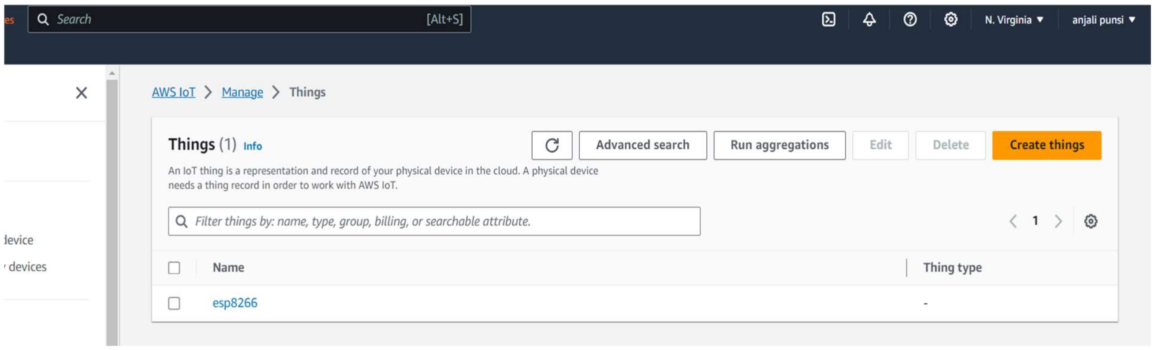
AWS IoT lets you select the most appropriate and up-to-date technologies for your solution. To help you manage and support your IoT devices in the field, AWS IoT Core supports these protocols:

- MQTT (Message Queuing and Telemetry Transport)

- MQTT over WSS (Websockets Secure)
- HTTPS (Hypertext Transfer Protocol – Secure)

We are using AWS MQTT client for publishing real-time data to AWS from the ESP32 board. MQTT is a lightweight and widely adopted messaging protocol that is designed for constrained devices.

Create an account on AWS using the email ID and password. Go to the IoT Core service from the AWS console. Create a thing associated with our ESP32 board. Spec- ifying thing properties, configure device certificate and attach policies to certificate. Download and store the certificates and keys in the same folder as the .ino code file.



Things (1) Info

An IoT thing is a representation and record of your physical device in the cloud. A physical device needs a thing record in order to work with AWS IoT.

Filter things by: name, type, group, billing, or searchable attribute.

Name	Thing type
esp8266	-

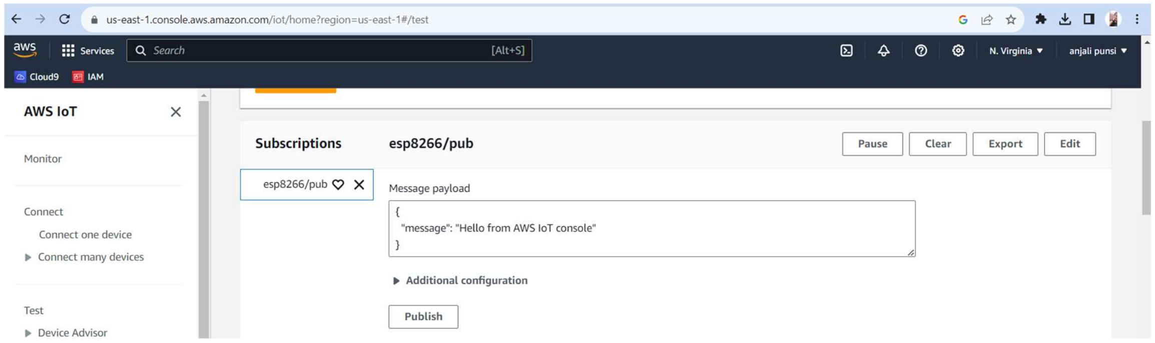
Figure 3.2: AWS Thing

```
#include "secret.h"
#include <WiFiClientSecure.h>#include <PubSubClient.h> #include
<ArduinoJson.h> #include "WiFi.h"

#define AWS_IOT_PUBLISH_TOPIC       "ESP_Home_security/pub" #define
AWS_IOT_SUBSCRIBE_TOPIC "ESP_Homesecurity /sub"

#define lamp 25
#define trigPin 33
#define echoPin 32

unsigned long lastMillis = 0;long duration, distance;
WiFiClientSecure net = WiFiClientSecure();PubSubClient client(net);


void connectAWS()
```

```
{
WiFi.mode(WIFI_STA); WiFi.begin(WIFI_SSID,  WIFI_PASSWORD);

Serial.println("Connecting to Wi-Fi"); while (WiFi.status() != WL_CONNECTED)
{
delay(500);
Serial.print(".");
}

// Configure WiFiClientSecure to use the AWS IoT device credentials
net.setCACert(AWS_CERT_CA);
net.setCertificate(AWS_CERT_CRT); net.setPrivateKey(AWS_CERT_PRIVATE);

// Connect to the MQTT broker on the AWS endpoint we defined earlier
client.setServer(AWS_IOT_ENDPOINT, 8883);

// Create a message handler client.setCallback(messageHandler);

Serial.println("Connecting to AWS IOT");while  (!client.connect(THINGNAME))
{
Serial.print("."); delay(100);
}

if (!client.connected())
{
Serial.println("AWS IoT Timeout!");return;
}

// Subscribe to a topic client.subscribe(AWS_IOT_SUBSCRIBE_TOPIC);

Serial.println("AWS IoT  Connected!");
}


void messageHandler(char* topic, byte* payload, unsigned int length)
{
Serial.print("incoming: ");Serial.println(topic);
```

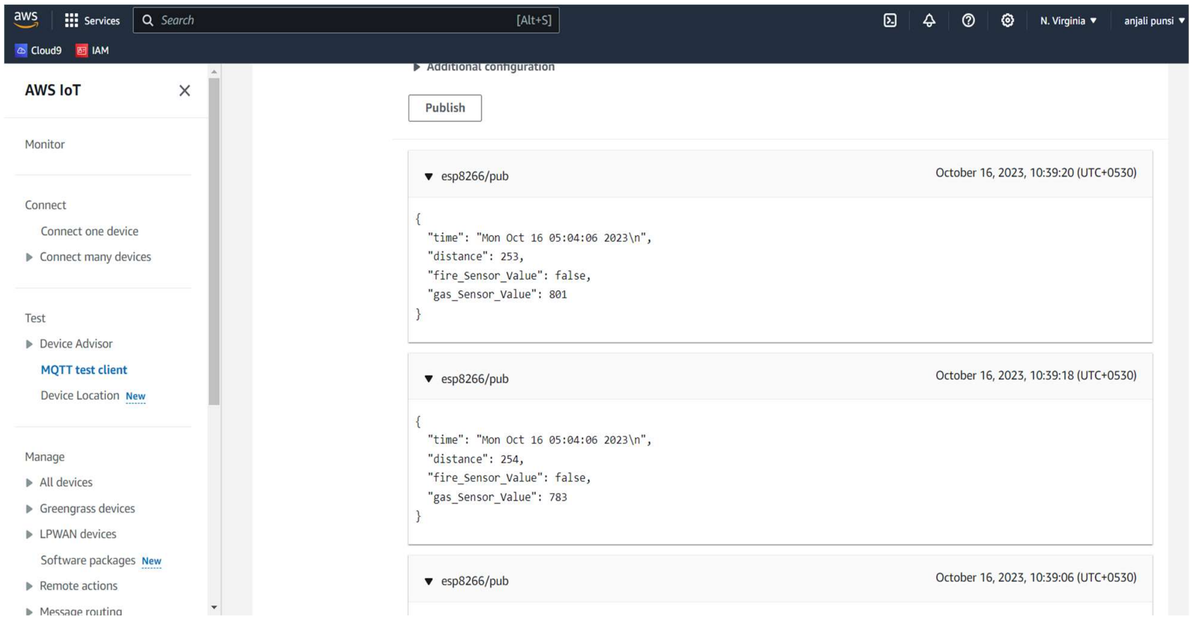
```
StaticJsonDocument<200> doc; deserializeJson(doc, payload);
const char* message = doc["message"];Serial.println();

for (int i = 0; i < length; i++)
{
Serial.print((char)payload[i]); // Pring payload content
}
char led = (char)payload[62]; // Extracting the controlling commandfrom the
Payload to Controlling LED from AWS
Serial.print("Command: ");Serial.println(led);

if (led == 49) // 49 is the ASCI value of 1
{
digitalWrite(lamp, HIGH); Serial.println("Lamp_State changed to HIGH");
}
else if (led == 48) // 48 is the ASCI value of 0
{
digitalWrite(lamp, LOW); Serial.println("Lamp_State changed to LOW");
}
Serial.println();
}


void publishMessage()
{
StaticJsonDocument<200> doc;doc["distance"] = distance; char jsonBuffer[512];
serializeJson(doc, jsonBuffer); // print to client

client.publish(AWS_IOT_PUBLISH_TOPIC,   jsonBuffer);
}


void setup()
{
Serial.begin(115200); connectAWS();
pinMode (lamp, OUTPUT); pinMode(trigPin, OUTPUT);pinMode(echoPin, INPUT);
digitalWrite(lamp, LOW);
```

```
}

void loop()
{
digitalWrite(trigPin, LOW); delayMicroseconds(2); digitalWrite(trigPin, HIGH);
delayMicroseconds(10); digitalWrite(trigPin,  LOW); duration = pulseIn(echoPin,
HIGH);distance = (duration/2) / 29.1;
if (distance < 10)
{
digitalWrite(lamp,HIGH);
}
else
{
digitalWrite(lamp,LOW);
}
Serial.print(distance); Serial.println(" cm"); delay(500);

if (!client.connected())
{
connectAWS();
}
else
{
client.loop();
if (millis() - lastMillis > 5000)
{
lastMillis = millis();publishMessage();
}
}
}
```



```
{
  "message": "Hello from AWS IoT console"
}
```

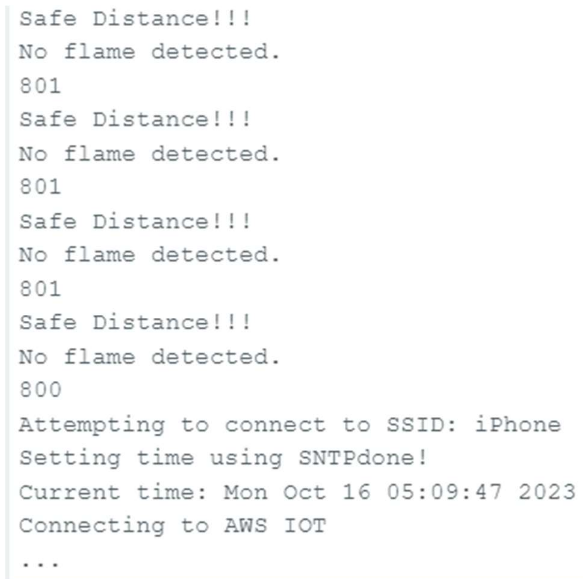
Figure 3.3: Subscribing to ESP32 on MQTT test client



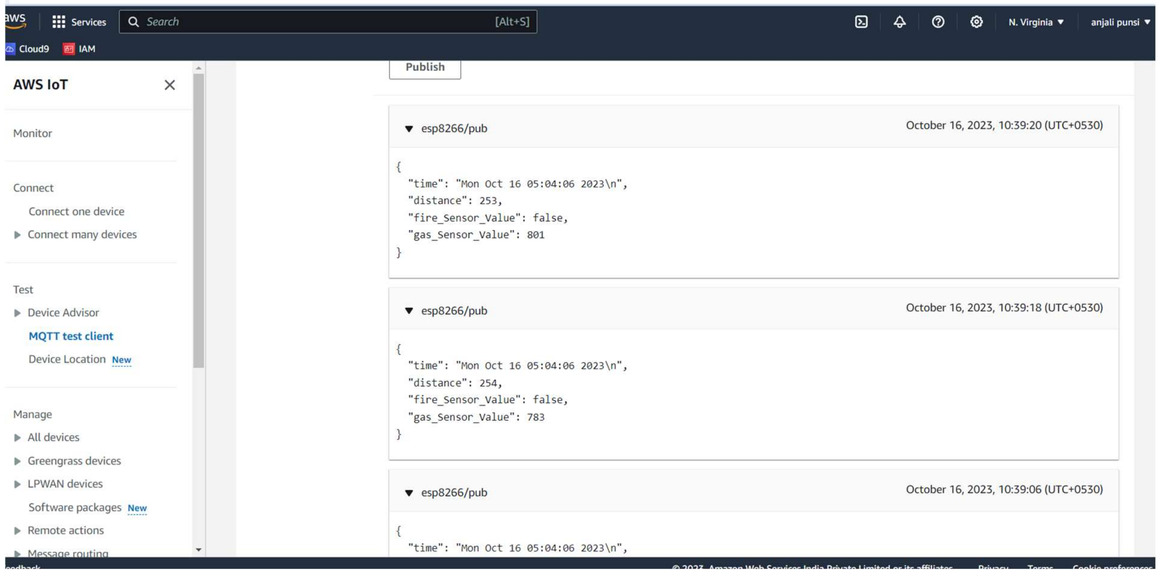
```
esp8266/pub                                   October 16, 2023, 10:39:20 (UTC+0530)
{
  "time": "Mon Oct 16 05:04:06 2023\n",
  "distance": 253,
  "fire_Sensor_Value": false,
  "gas_Sensor_Value": 801
}

esp8266/pub                                   October 16, 2023, 10:39:18 (UTC+0530)
{
  "time": "Mon Oct 16 05:04:06 2023\n",
  "distance": 254,
  "fire_Sensor_Value": false,
  "gas_Sensor_Value": 783
}

esp8266/pub                                   October 16, 2023, 10:39:06 (UTC+0530)
```

Figure 3.4: Live MQTT data published from ESP32 board

3.2 Outcome of Analytics



```
Safe Distance!!!
No flame detected.
801
Safe Distance!!!
No flame detected.
801
Safe Distance!!!
No flame detected.
801
Safe Distance!!!
No flame detected.
800
Attempting to connect to SSID: iPhone
Setting time using SNTPdone!
Current time: Mon Oct 16 05:09:47 2023
Connecting to AWS IOT
...
```



esp8266/pub — October 16, 2023, 10:39:20 (UTC+0530)

```
{
  "time": "Mon Oct 16 05:04:06 2023\n",
  "distance": 253,
  "fire_Sensor_Value": false,
  "gas_Sensor_Value": 801
}
```

esp8266/pub — October 16, 2023, 10:39:18 (UTC+0530)

```
{
  "time": "Mon Oct 16 05:04:06 2023\n",
  "distance": 254,
  "fire_Sensor_Value": false,
  "gas_Sensor_Value": 783
}
```

esp8266/pub — October 16, 2023, 10:39:06 (UTC+0530)

```
{
  "time": "Mon Oct 16 05:04:06 2023\n",
```

Figure 3.6: ESP Home security

3.2 Application Development

We developed the application using flask Flask is used for developing web applications using python, implemented on Werkzeug and Jinja2. Following are some libraries re- quired
from tkinter import *import time as t import json
import AWSIoTPythonSDK.MQTTLib as AWSIoTPyMQTTfrom awscrt import mqtt

We created an web page for showing the distance fetched from cloud storage andcontrol the led by using butt

3.4 Implementation Code and Snapshots

```
from tkinter import *import time as t import json
import AWSIoTPythonSDK.MQTTLib as AWSIoTPyMQTTfrom awscrt import mqtt

# Define ENDPOINT,
CLIENT_ID, PATH_TO_CERTIFICATE,PATH_TO_PRIVATE_KEY,
PATH_TO_AMAZON_ROOT_CA_1, MESSAGE, TOPIC, and RANGE
ENDPOINT = "a3rh2jvfe6wvsn-ats.iot.us-west-2.amazonaws.com"CLIENT_ID =
"iotconsole-1667048527569-1"
PATH_TO_CERTIFICATE = "certificates
/bcca6128416b15f8a176a92671d34230015e1ebabfa63b3cbd3a3df3c670bbe8-
certificate.pem. crt"
PATH_TO_PRIVATE_KEY = "certificates
/bcca6128416b15f8a176a92671d34230015e1ebabfa63b3cbd3a3df3c670bbe8-private
.pem.key"
PATH_TO_AMAZON_ROOT_CA_1 = "certificates/AmazonRootCA1.pem"MESSAGE = "Hello
World"
TOPIC = "ESP_Home_Automation/pub"
RANGE = 20

myAWSIoTMQTTClient = AWSIoTPyMQTT.AWSIoTMQTTClient(CLIENT_ID)
myAWSIoTMQTTClient.configureEndpoint(ENDPOINT, 8883)
myAWSIoTMQTTClient.configureCredentials( PATH_TO_AMAZON_ROOT_CA_1,
PATH_TO_PRIVATE_KEY, PATH_TO_CERTIFICATE)

myAWSIoTMQTTClient.connect() distance = 0;
action = None auto_on = False
dataq = {"distance": 0}

def refresh(client, userdata, message):global   distance,dataq print("navin")
payload = message.payload.decode("utf-8")payload  =  json.loads(payload)
print(payload)
distance = payload['distance']dataq = {"distance": distance}
```

```
    myAWSIoTMQTTClient.subscribe("ESP_Home_Automation/pub",1 ,refresh)
    print("hello")




    from flask import Flask,render_template,url_for,request,redirect, make_response
    import random
    import json
    from time import time from random import random
    from flask import Flask, render_template, make_responseapp = Flask(_name_)


    @app.route('/', methods=["GET", "POST"])def main():

if request.method == 'POST':print(request.args);

                              return render_template('index.html')


    @app.route('/post/data', methods=["GET", "POST"])def status():
    if request.method == 'POST':
        stat = request.data.decode("utf-8")if(stat == 'status=ON'):
    print("ONN") myAWSIoTMQTTClient.publish("ESP_Home_Automation/sub",json.dumps({
   "message": "Hello from AWS IoT consle","led_Control":"1"
    }),1)
elif(stat =='status=OFF'):
    myAWSIoTMQTTClient.publish("ESP_Home_Automation/sub",json.dumps({
   "message": "Hello from AWS IoT consle","led_Control":"1"
    }),1)

                              return render_template('index.html')

    @app.route('/data', methods=["GET", "POST"])def data():
    # Data Format
    # [TIME, Temperature, Humidity]global dataq
    # Temperature = random() * 100
# Humidity = random() * 55# airdata = random() * 55
# data = {"time":time() * 1000,"temperature": Temperature, "humidity":
Humidity, "airdata": airdata}
response = make_response(json.dumps(dataq))response.content_type =
'application/json' return  response

  if _name_ == "_main_":app.run(debug=True)
myAWSIoTMQTTClient.disconnect()
```

Chapter 4

Conclusion and Future Scope

The working prototype of our Home security system has been tested and imple-mented. Since it comprises a Ultrasonic sensor (HC Sro4), the appliances work onlyif there is any physical motion with the defined range . The ultrasonic Sensor wouldallow us to elicit data such as distance to and from an object/human and with thatdata the precision of human motion detection. The data is pushed to aws cloud i.e dis- tance it will push the data that the object is at which distance from our sensor.All the appliances connected to the board can be monitored remotely which works flawlessly . This prototype can be used to solve real world problems, automating the street lights.For instance and substantial amount of electricity can be saved by switching off the appliances/devices when there is no physical motion.

As a future enhancement, we are also working on configuring More sensors attached to system for betterment. And adding the securities feature in our home securityfor the users. We will use more sensors with high precision value.

References

[1] Sensors and their applications ‘https://www.electronicshub.org/ different-types-sensors/’

[2] Gill, Khusvinder, et al. *‘A zigbee-based Home security system.’* IEEE Transac- tions on consumer Electronics 55.2 (2009): 422-430

[3] Home security system ‘https://www.researchgate.net/publication/ 224264238_Requirements_for_Smart_Home_Applications_and_Realization_ with_WS4D-PipesBox’

[4] ‘https://www.irjet.net/archives/V2/i3/Irjet-v2i3317.pdf’