FYP REVIEW ONE REPORT

STOCK PREDICTION USING STACN

A PROJECT REPORT

Submitted by

Nehanth Kasi Geetha 2019115061

Sparsh Kumar Sharoff 2019115101

Vinayak S 2019115121

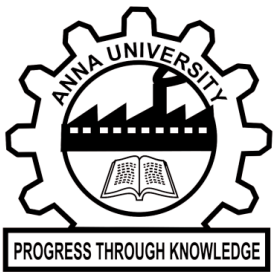
submitted to the Faculty of

INFORMATION SCIENCE AND TECHNOLOGY

PANEL - 1

As a part of

IT5712 - FINAL YEAR PROJECT - 1



DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600 025
OCTOBER 2022

AIM

To develop a module which detects water leakage from a tap, informs the leakage via telegram and updates the details in a cloud database.

ABSTRACT

India has 16% of the world's population and only 4% of the world's water resources, which are depleting rapidly. The demand for water is expected to grow from 40 billion cubic metres (bcm) currently to around 220 bcm in 2025. In such a dire situation, it is essential that each and every drop of water be saved and utilised to the fullest. The proposed system implemented by our module tried to enforce this reason by stopping tap water leaks in our department. Our module does this by sending a message via telegram to notify the management whenever there is any tap leakage. Along with sending a message, it also accompanies a buzzer alarm which alerts any bystander about the leak and prompts them to close the tap. The leakage details are at the same time saved in a cloud database in order to facilitate future analysis of the timings and durations of the same.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Water crisis is a reality in today's world and we need to ensure that water is saved to the fullest. We, being students from IT backgrounds, have tried to help in solving the problem by creating an innovative system which saves water that is leaking from taps. Tap water leakage is a major problem in every municipality and solving this problem can save millions of cubic cm of water. This will be implemented in the Information Science and Technology department's water systems on a trial basis which can then be scaled to a higher level later. We will be using the ESP32 microprocessor which facilitates Wifi in order to create a system which enables the storage of the data in cloud platforms so that future analysis of the data is possible. An ultrasonic sensor is used to detect leakage and its stoppage is done via buzzer and the telegram app notification.

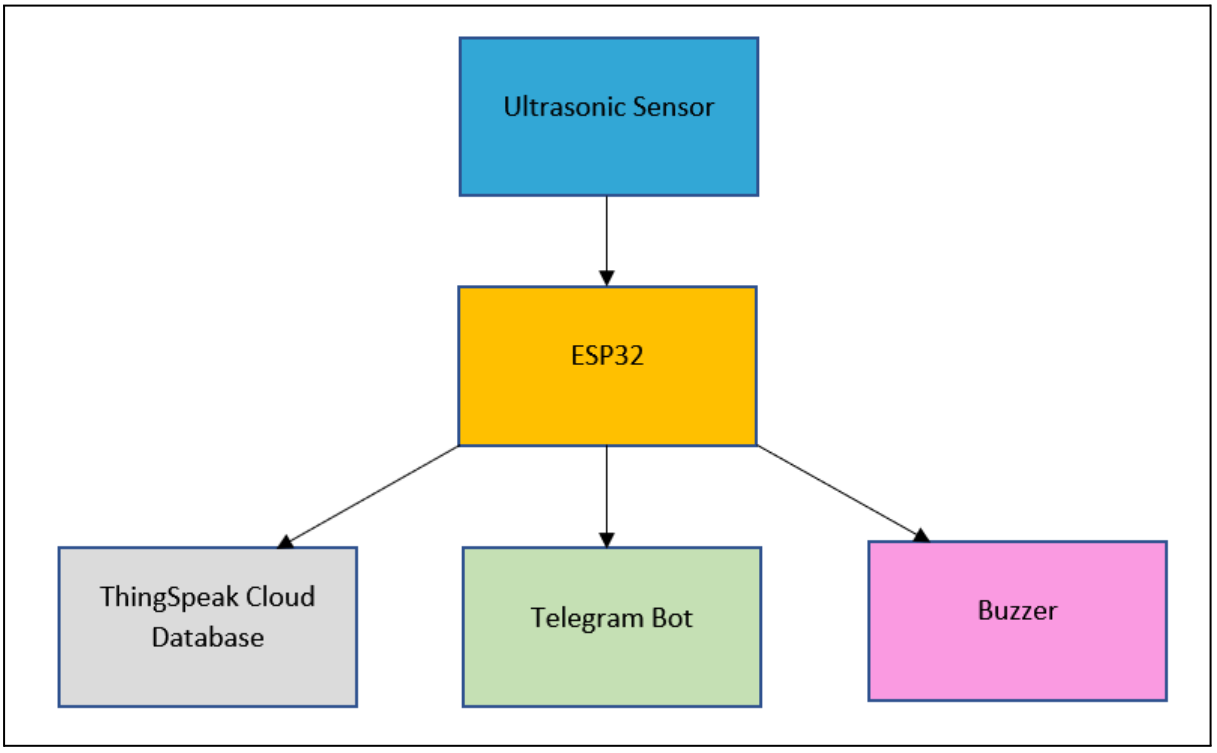
Unlike the usual systems that are already present, our project does not just detect leakages but also updates the information to the cloud platform - ThingSpeak in order to perform future analysis of the data. It uses an ultrasonic sensor in an innovative way in order to detect leakages while at the same time using telegram bot API calls in order to send notifications.

CHAPTER 2
SYSTEM ARCHITECTURE

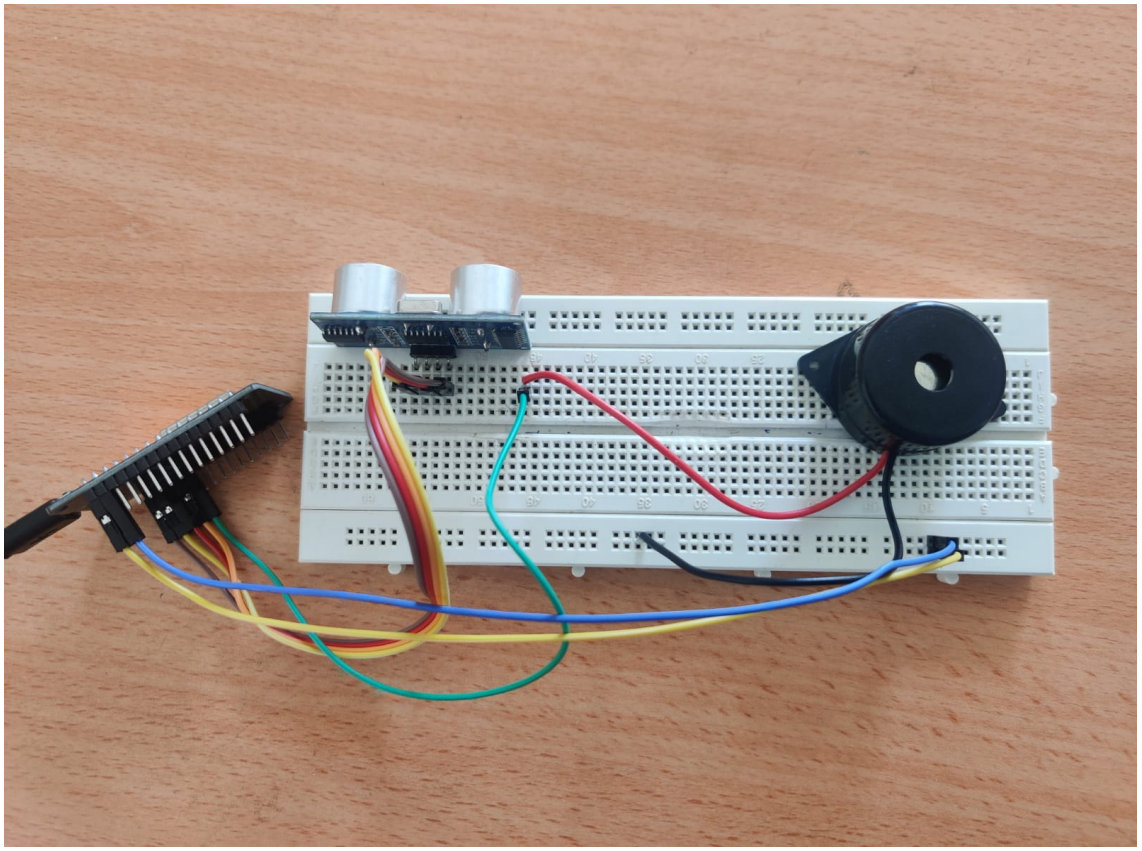
2.1 PROPOSED SYSTEM

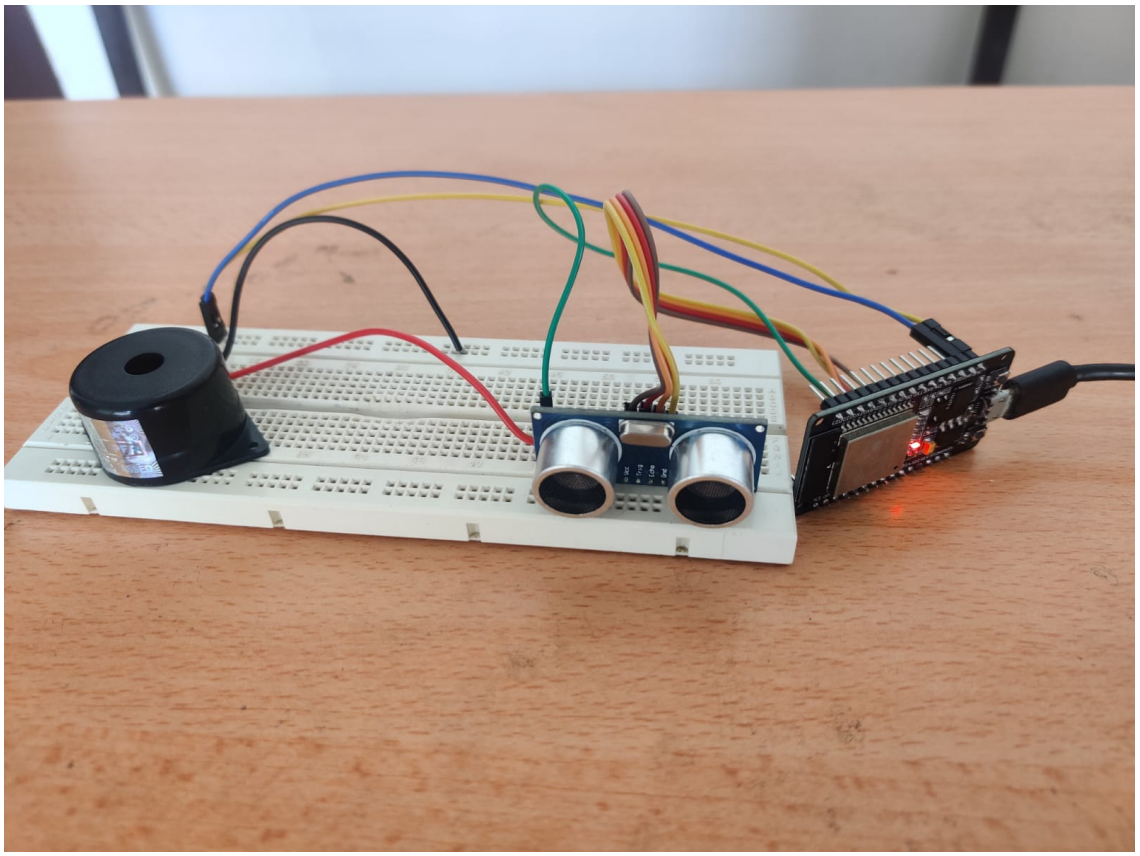
The proposed system consists of three main portions: The first portion includes all the hardware components which pick up all of the data which is detected. The hardware includes the ultrasonic sensor, esp32, buzzer and connecting wires. The second portion includes the details about how the data which is picked up is delivered to the cloud database via Wifi present in the ESP32. This data can then be used for future analysis. The third module is the telegram messenger which notifies of any leak when detected to the management.

2.2 BLOCK DIAGRAM

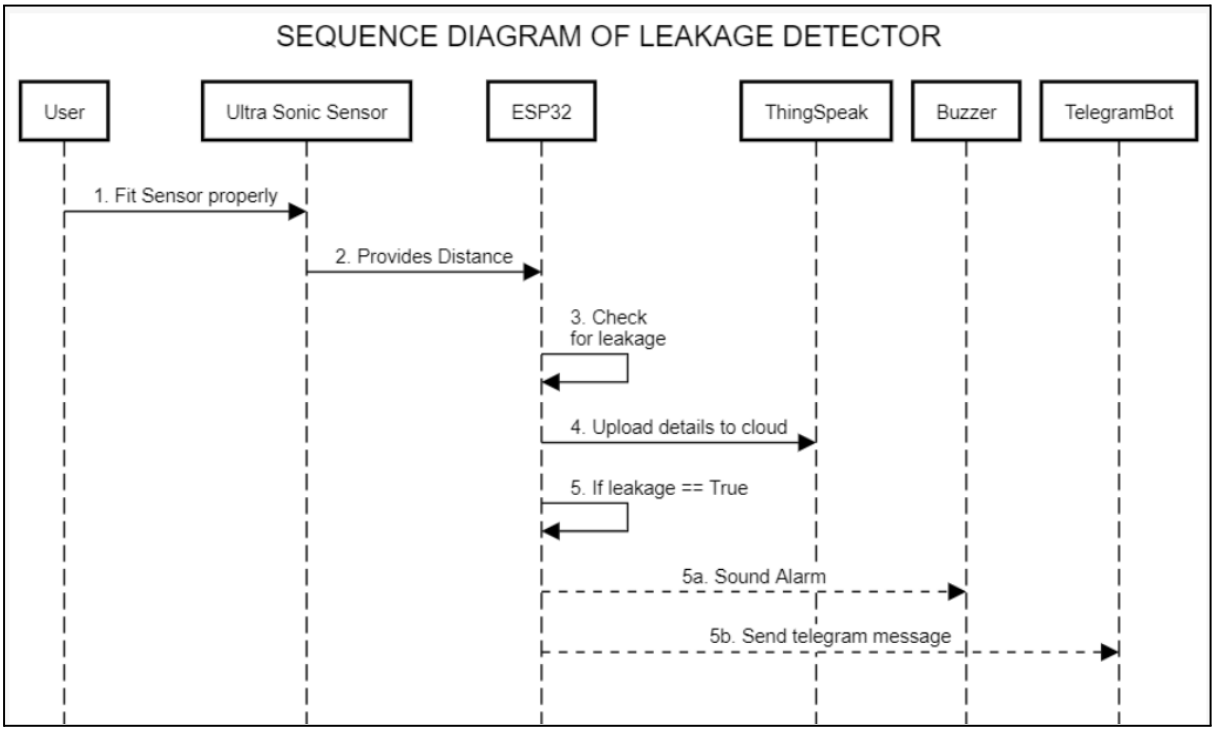


2.3 WORKING MODEL





2.4 SEQUENCE DIAGRAM

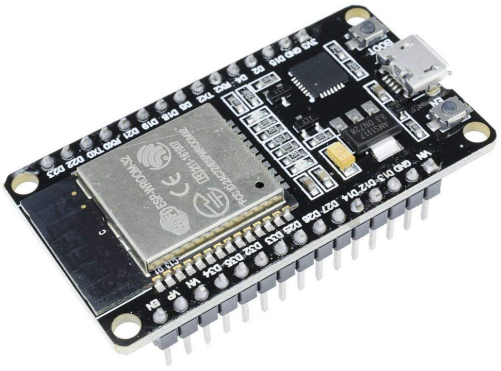


2.5 TOOLS AND COMPONENTS USED

1. Breadboard
2. ESP32 - WROOM-32
3. Ultrasonic Sensor
4. M-M Jumper Wires
5. M-F Jumper Wires
6. Piezo Buzzer
7. 9V DC Adapter
8. 3.3V/5V MB102 Breadboard Power Supply Module
9. PC with Arduino IDE

CHAPTER 3
DETAILS OF COMPONENTS

3.1 ESP32 - WROOM-32



ESP32-WROOM-32 is a powerful, generic Wi-Fi + Bluetooth + Bluetooth LE MCU module that targets various applications, ranging from low-power sensor networks to the most demanding tasks. The chip embedded is designed to be scalable and adaptive. There are two CPU cores that can be individually controlled, and the CPU clock frequency is adjustable from 80 MHz to 240 MHz.

3.2) Ultrasonic Distance Sensor Module - HC-SR04

The HC-SR04 ultrasonic sensor uses sonar to determine the distance to an object. This sensor reads from 2cm to 400cm (0.8inch to 157 inch) with an accuracy of 0.3cm (0.1 inches). In addition, this particular module comes with ultrasonic transmitter and receiver modules.

3.3 Piezo Buzzer

A piezo buzzer is a type of electronic device that's used to produce a tone, alarm or sound. It works by applying an alternating voltage to the piezoelectric ceramic material. The introduction of such an input signal causes the piezoceramic to vibrate rapidly, resulting in the generation of sound waves.

CHAPTER 4
IMPLEMENTATION

4.1 WORKING PRINCIPLE

The ultrasonic sensor is used to find the distance between the source and the closest object. It sends the readings to the ESP32. Here the readings are processed and if the value is less than 10 cm, then the tap is considered open. ESP32 uses wifi connectivity to get the actual time. If the tap is open for more than 30 seconds, it is considered to be a leak. The buzzer is alarmed, information about the leak is sent via a telegram bot and the entire leakage instance is uploaded to the ThingSpeak cloud database.

4.2 PROGRAM CODE

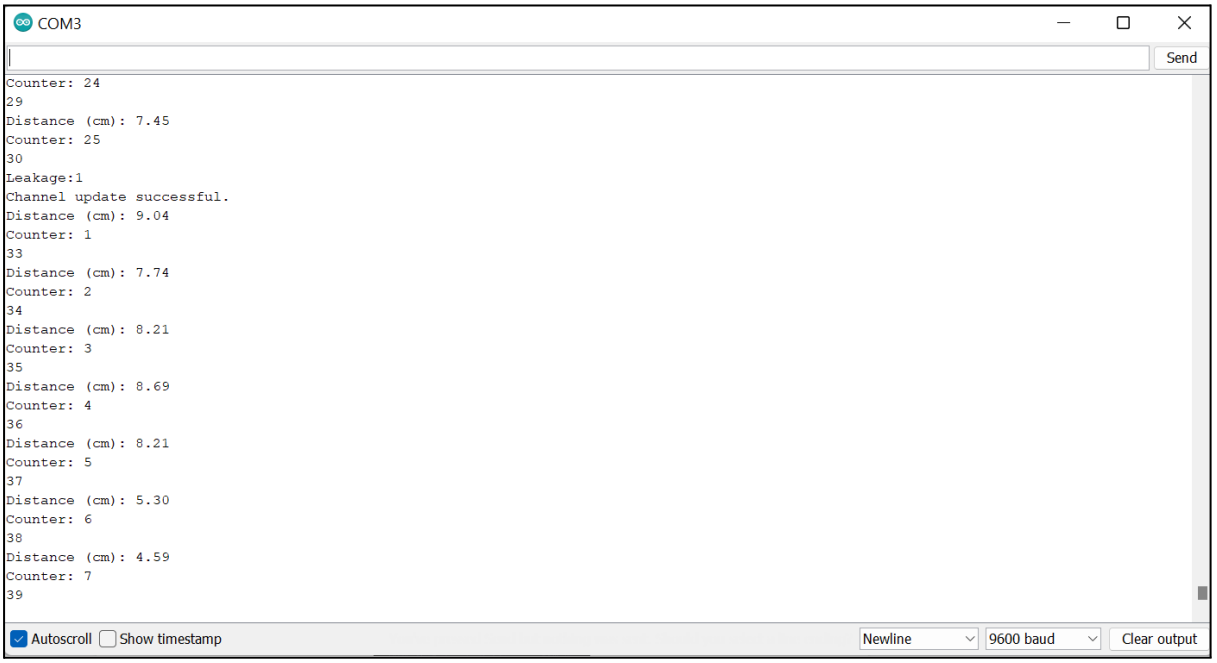
```
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include "time.h"
#include "ThingSpeak.h"
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

#define BOTtoken "5443593942:AAFfr1fQQA16FnhOwPv_H4MFEukpj-jgU7A"

#define CHAT_ID "718836497"

WiFiClient client1;
WiFiClientSecure client;
UniversalTelegramBot bot(BOTtoken, client);
// Initialize Telegram BOT

long unsigned int CHANNEL_ID = 1760669;
const char* CHANNEL_API_KEY = "T3DDI9F615EQAUKF";

const char* ssid       = "VinayakSur";
const char* password   = "12345678";

const char* ntpServer = "pool.ntp.org";
const long  gmtOffset_sec = 16200;
const int   daylightOffset_sec = 3600;

const int trigPin = 13;
const int echoPin = 12;
const int buzpin = 14;
```

```
//define sound speed in cm/uS
#define SOUND_SPEED 0.034
int counter1 = 0;
int counter = 0;
int leakage;
int buzztime = 0;

void setup() {
  Serial.begin(9600); // Starts the serial communication
  Serial.printf("Connecting to %s ", ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
      delay(500);
      Serial.print(".");
  }
  Serial.println(" CONNECTED");
  client.setCACert(TELEGRAM_CERTIFICATE_ROOT);
  ThingSpeak.begin(client1);

  //init and get the time
  configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);

  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  pinMode(buzpin, OUTPUT); // Sets the buzPin as an Output
}

void loop() {
  struct tm timeinfo;

  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Reads the echoPin, returns the sound wave travel time in microseconds
  long duration = pulseIn(echoPin, HIGH);

  // Calculate the distance
  float distanceCm = duration * SOUND_SPEED/2;

  // Prints the distance in the Serial Monitor
  Serial.print("Distance (cm): ");
```

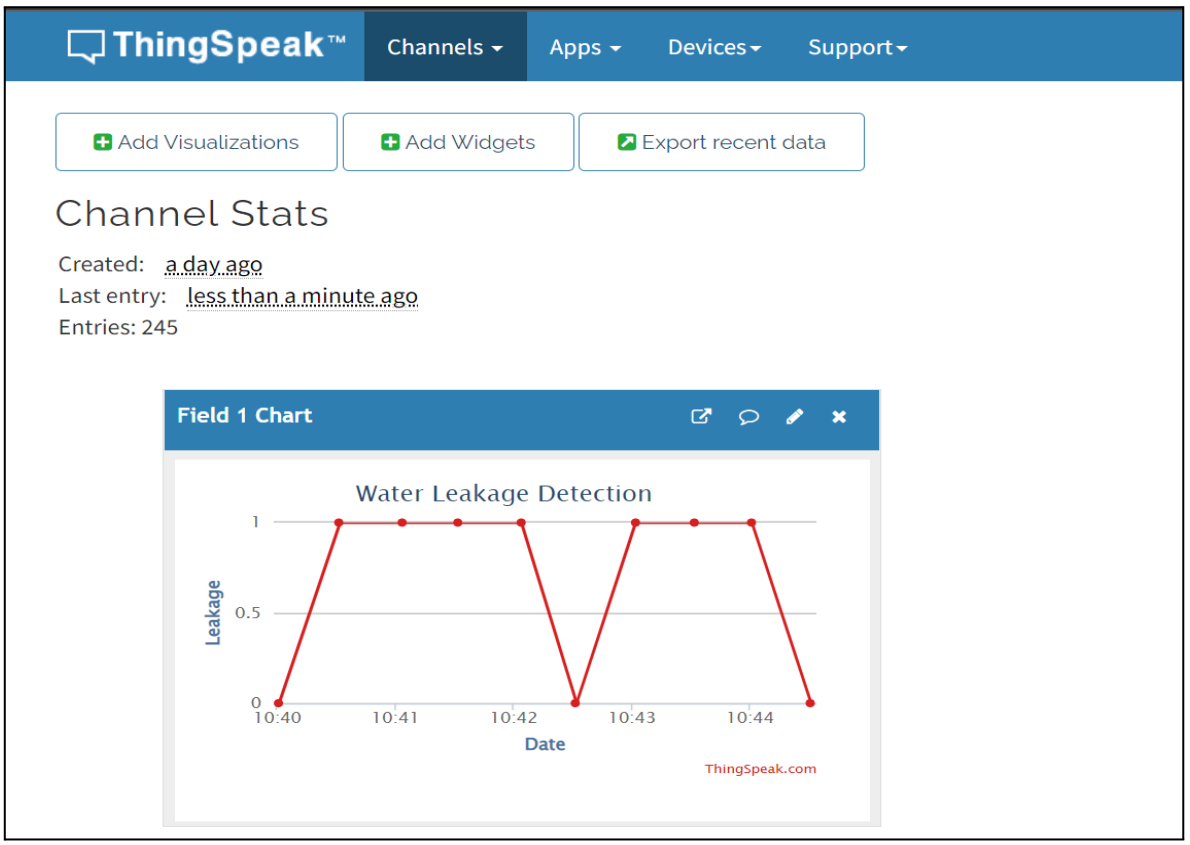
```
  Serial.println(distanceCm);
  if(distanceCm <= 10){
    counter1++;
    counter++;
  }
  Serial.print("Counter: ");
  Serial.println(counter);
  if(!getLocalTime(&timeinfo)){
    Serial.println("Failed to obtain time");
    return;
  }
  int s = timeinfo.tm_sec;
  Serial.println(s);

  if(timeinfo.tm_sec % 30 == 0){
    if(timeinfo.tm_sec ==0){
      if(counter1>=40){
        bot.sendMessage(CHAT_ID, "Leakage Alert!!!", "");
      }
     counter1=0;
    }
    if(counter <= 15){
      leakage = 0;
    }
    else{
      buzztime = 10;
      leakage = 1;
    }
    Serial.print("Leakage:");
    Serial.println(leakage);
    int x = ThingSpeak.writeField(CHANNEL_ID ,1, leakage, CHANNEL_API_KEY);
    if(x == 200){
      Serial.println("Channel update successful.");
    }
    else{
      Serial.println("Problem updating channel. HTTP error code " + String(x));
    }
    counter = 0;
  }
  if(buzztime > 0){
    digitalWrite(buzpin, HIGH);
    buzztime--;
  }
  if(buzztime == 0){
    digitalWrite(buzpin, LOW);
  }
  delay(1000);
}
```

CHAPTER 5
OBSERVATIONS

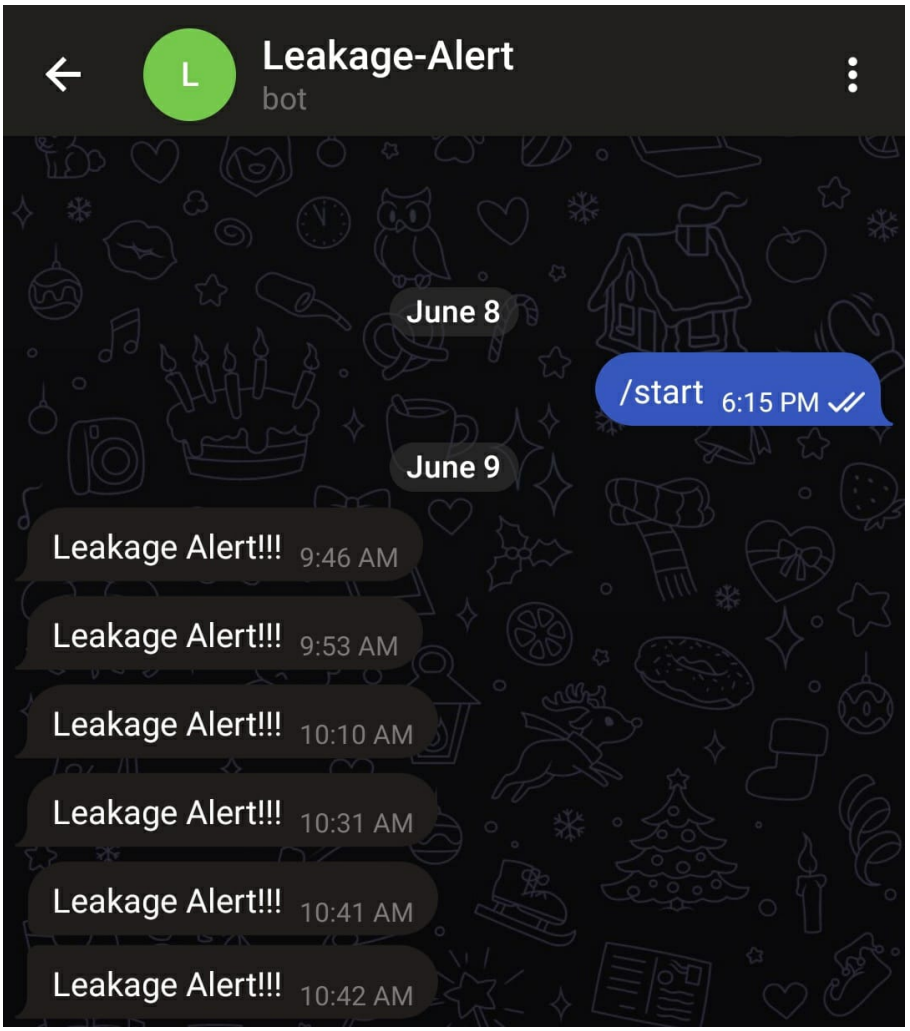
5.1: CONSOLE



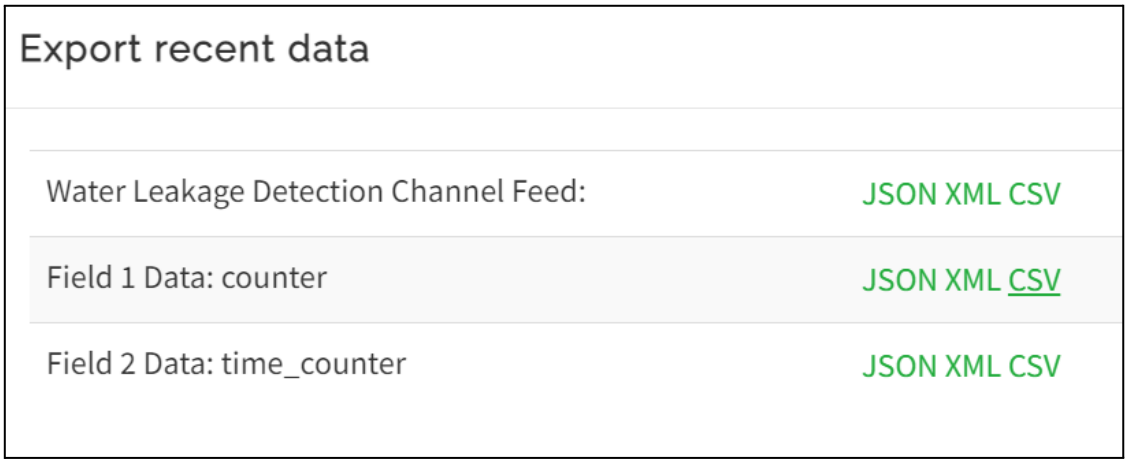
5.2 THINGSPEAK INTERFACE



5.3 TELEGRAM BOT NOTIFICATION



5.4 EXPORTING DATA



	A	B	C	D
1	created_at	entry_id	field1	
2	2022-06-09 04:10:03 UTC	170	0	
3	2022-06-09 04:11:02 UTC	171	0	
4	2022-06-09 04:11:32 UTC	172	0	
5	2022-06-09 04:12:01 UTC	173	0	
6	2022-06-09 04:12:31 UTC	174	0	
7	2022-06-09 04:13:01 UTC	175	0	
8	2022-06-09 04:13:32 UTC	176	0	
9	2022-06-09 04:14:02 UTC	177	0	
10	2022-06-09 04:16:04 UTC	178	1	
11	2022-06-09 04:16:31 UTC	179	0	
12	2022-06-09 04:22:32 UTC	180	1	
13	2022-06-09 04:24:02 UTC	181	1	
14	2022-06-09 04:26:31 UTC	182	0	
15	2022-06-09 04:27:32 UTC	183	0	
16	2022-06-09 04:28:02 UTC	184	0	
17	2022-06-09 04:28:32 UTC	185	1	
18	2022-06-09 04:29:02 UTC	186	0	
19	2022-06-09 04:30:01 UTC	187	1	

CHAPTER 6
RESULT AND CONCLUSION

5.1 RESULT

The module detected the water leakages successfully and helped to stop them via the buzzer and telegram notification modules. At the same time, the data of the leakages with timestamps were constantly uploaded on the ThingSpeak cloud database which opens future analysis opportunities upon that data.

5.2 CONCLUSION

The proposed system will help save a lot of water that is being wasted in the department due to water tap leakages. With this system, the future water crisis can be averted if multiplied on a large scale.

ACKNOWLEDGEMENT

We would like to thank our professors Dr.Selvi Ravindran and Dr.Narasimhan for helping and guiding us throughout our project and ensuring that it is successfully implemented in the department.