A Project Based Report
On

"DATA LOGGER TO TELIGRAM USING NODEMCU"

Project By

BIDARKAR NISHANT

Submitted to

"DEPARTMENT OF ELECTRONIC SCIENCE AND INSTRUMENTATION SCIENCE"

(Savitribai Phule Pune University)

ABSTRACT

This is the report of our semester final project of the System Design lab(EL-301) .This report will give the reader an overview about the steps which are involved in the making of this project. The project is about the making of Data Logger that will sent a sensor data to the telegram over the internet by using the Teligram bot. In this project the following components NodeMCU(ESP-8266), LM35 Temperature Sensor and Smartphone were used to collect the data. First the circuit was implemented on breadboard to check the basic working of the circuit . The temperature sensed by the LM35 sensor were accurate. Then after all circuit components are soldered on zero PCB . For continuously proper working of the data logger stable internet connection is required that will be provided by the WiFi Hotspot of the smartphone.

Table of Contents

Chapter 1

Introduction

Data Logging to Teligram using NodeMCU :

The Data logger is used to send data sensor data collected by the NodeMCU to the telegram over the internet. This is really convenient way of monitoring the sensor data remotely and staying up-to-date about the remote location. The main intention of carrying out this project is to transmit data to telegram using NodeMCU. While doing this Project I learned creating telegram bot ,how to use that bot to send data by sending messages via telegram.

Overview:

1.1. Purpose of the project

Basic main purpose of this project is to connect Teligram to the NodeMCU and Make a data exchange over the internet. This project will lead to the basic understanding of connecting telegram like Messaging platforms to IoT devices like NodeMCU . This idea is further can be used in so many different applications for IoT.

1.2. Applications of the Project

The idea in this project is further implemented in so many different applications that are mentioned below-

- Door Security Alert System
- CCTV camera
- Getting Security alerts on chat
- Periodic message alerts related to sensor data on chat

The application about cctv cameras mentioned can be implemented using the ESP32-cam . We know the telegram messaging platform totally works on the cloud . We can also use telegram as a cloud storage platform by storing the data in messages. This feature of telegram can be used for storing the cctv camera footages to the telegram chat. We can program the Teligram Bot to periodically send the Camera footages as a telegram message this will reduce the extra storage and money we have to spend on the data storage device that requires to store the cctv camera pootages.

1.3. Theoretical Knowledge Required

To carry out this project the main requirement is to understand the C programming language which is the main language that we have used to program the NodeMCU . Along with C language we must be familiar with the Arduino IDE and using the WiFi . Along with this we must know how to interface sensors with the microcontroller , in our case we are using the
NodeMCU.

We'll have to go through the data sheets of the NodeMCU and the sensors that we are using here.

Chapter 2

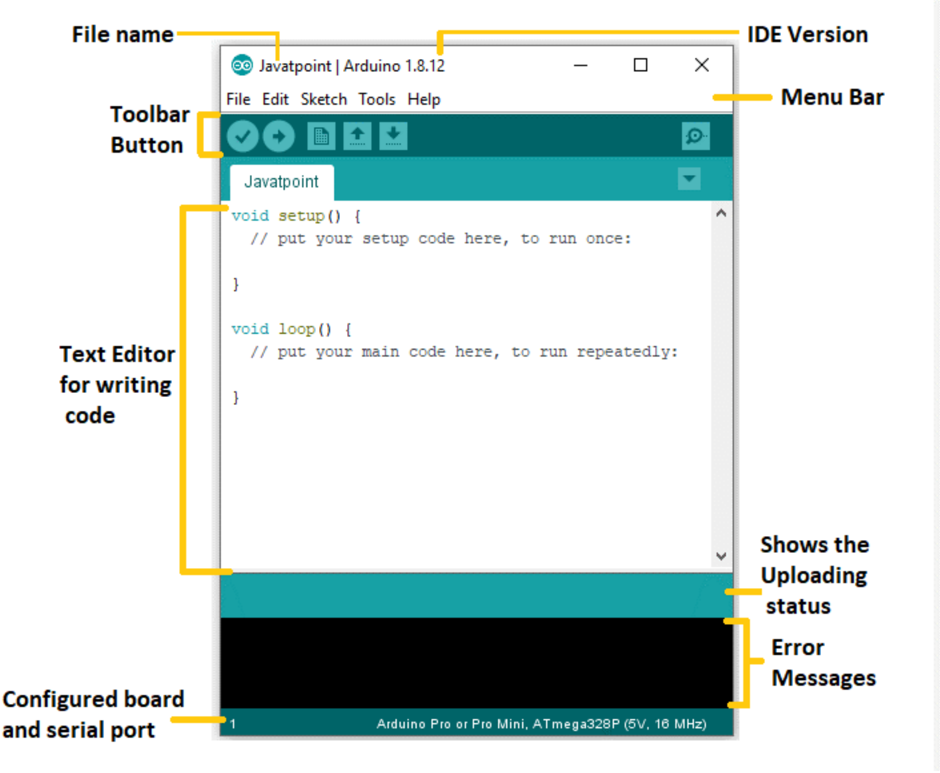
TOOLS AND ECHNIQUES

2.1 IDE(Integrated Development Environment)

IDE stands for Integrated Development Environment. It's a software application that helps programmers develop software code. IDEs combine tools for writing, building, testing, and packaging software into a single application.This increases developer productivity.

In our case we are using the **Arduino IDE**

The Arduino IDE will appear as:



File name
IDE Version
Javatpoint | Arduino 1.8.12
File Edit Sketch Tools Help
Menu Bar
Toolbar Button
Javatpoint
Text Editor for writing code
Shows the Uploading status
Error Messages
Configured board and serial port
1
Arduino Pro or Pro Mini, ATmega328P (5V, 16 MHz)

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

The Arduino IDE is an open-source software, which is used to write and upload code to the Arduino boards. The IDE application is suitable for different operating systems such as Windows, Mac OS X, and Linux. It supports the programming languages C and C++. Here, IDE stands for Integrated Development Environment.

The program or code written in the Arduino IDE is often called as sketching. We need to connect the Genuino and Arduino board with the IDE to upload the sketch written in the Arduino IDE software. The sketch is saved with the extension '.ino.'

2.2 Hardware

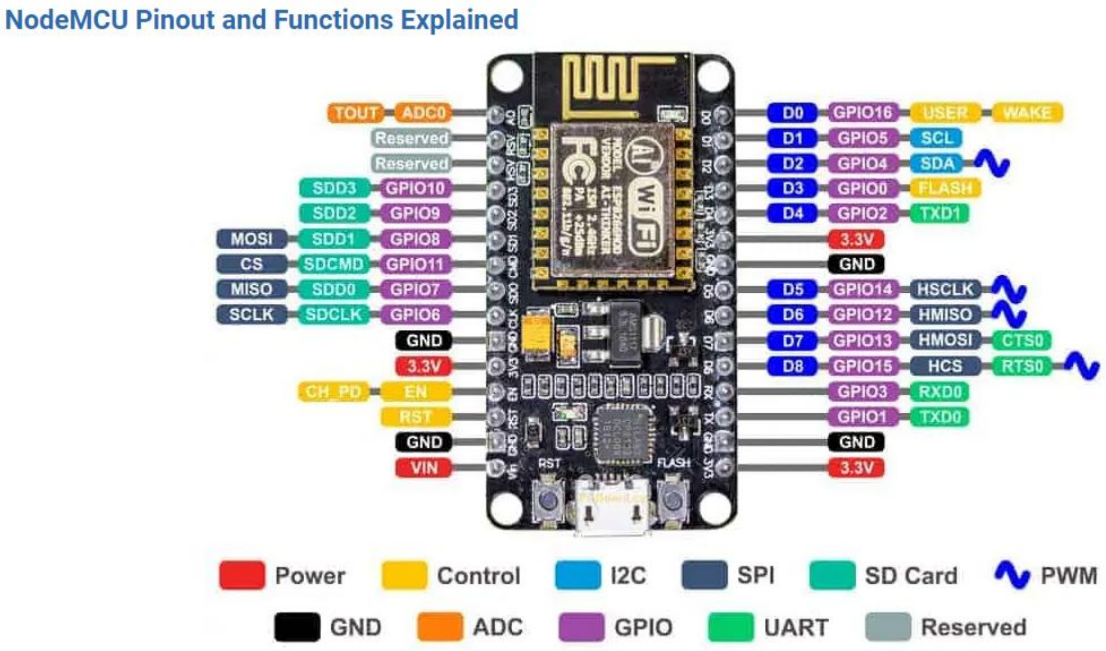
2.2.1 NodeMCU(ESP8266)

The NodeMCU (Node MicroController Unit) is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (SoC) called the ESP8266.NodeMCU is a IoT development board that has inbuilt WiFi to this board . This Board is so famous for IoT applications due to its wireless capabitlities and cheap cost. The ESP8266, designed and manufactured by Espressif Systems, contains the crucial elements of a computer: CPU, RAM, networking (WiFi), and even a modern operating system and SDK. That makes it an excellent choice for Internet of Things (IoT) projects of all kinds.

NodeMCU Technical Specifications

	Official NodeMCU	NodeMCU Carrier Board
Microcontroller	ESP-8266 32-bit	ESP-8266 32-bit
NodeMCU Model	Amica	Amica
NodeMCU Size	49mm x 26mm	49mm x 26mm
Carrier Board Size	n/a	102mm x 51mm
Pin Spacing	0.9" (22.86mm)	0.9" (22.86mm)
Clock Speed	80 MHz	80 MHz
USB to Serial	CP2102	CP2102
USB Connector	Micro USB	Micro USB
Operating Voltage	3.3V	3.3V
Input Voltage	4.5V-10V	4.5V-10V
Flash Memory/SRAM	4 MB / 64 KB	4 MB / 64 KB
Digital I/O Pins	11	11
Analog In Pins	1	1
ADC Range	0-3.3V	0-3.3V

	Official NodeMCU	NodeMCU Carrier Board
UART/SPI/I2C	1 / 1 / 1	1 / 1 / 1
WiFi Built-In	802.11 b/g/n	802.11 b/g/n
Temperature Range	-40C - 125C	-40C - 125C



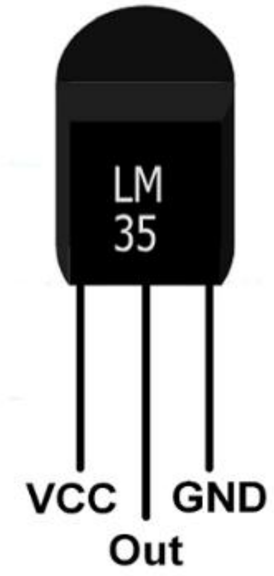
2.2.2 LM35 sensor

- LM35 is a temperature measuring device having an analog output voltage proportional to the temperature.
- It provides output voltage in Centigrade (Celsius). It does not require any external calibration circuitry.
- The sensitivity of LM35 is 10 mV/degree Celsius. As temperature increases, output voltage also increases.

E.g. 250 mV means 25°C.

- It is a 3-terminal sensor used to measure surrounding temperature ranging from -55 °C to 150 °C.
- LM35 gives temperature output which is more precise than thermistor output.

LM35 Temperature Sensor



LM35 Sensor Pinout

VCC: Supply Voltage (4V – 30V)

Out: It gives analog output voltage which is proportional to the temperature (in degree Celsius).

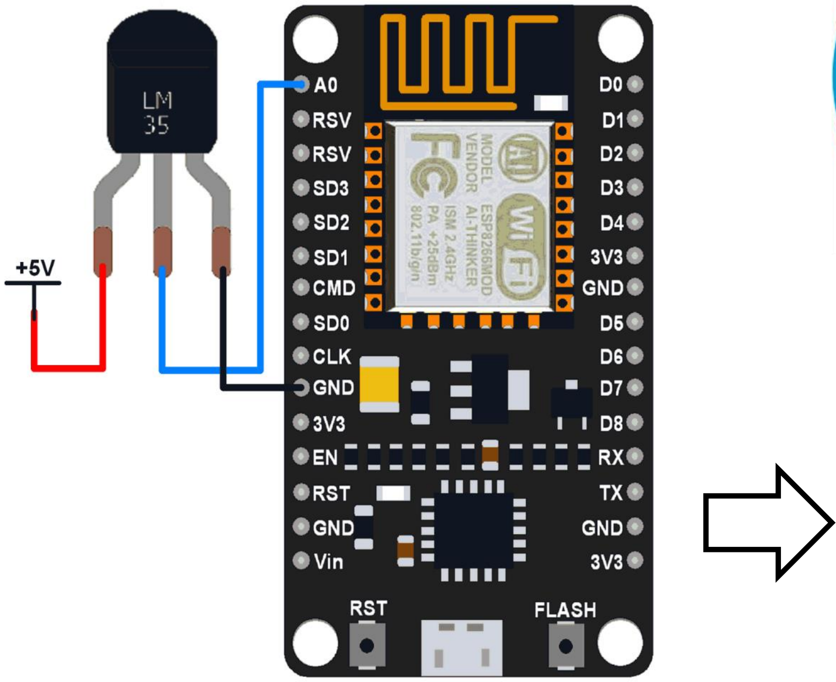
GND: Ground

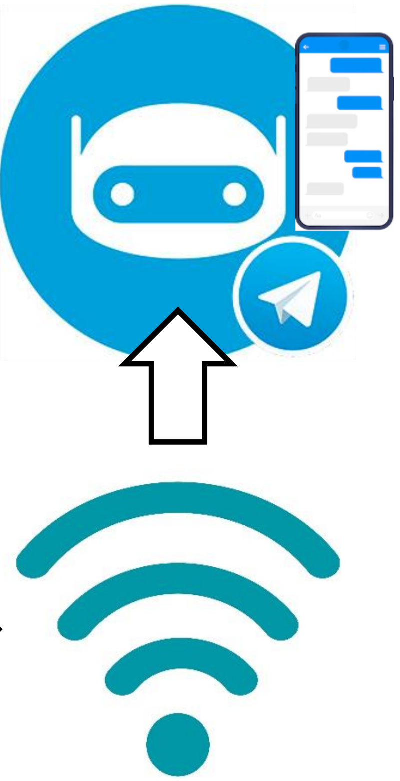
Specification of LM35 Temperature Sensor

- Operating Voltage: 4 V to 30 V
- Output Voltage: 10mV/°C
- Sensitivity: 10mV/°C
- Linearity Error: ±1°C (for 0°C to +100°C)

- Operating Temperature: -55°C to +150°C
- Output Impedance: 100 Ω
- Power Consumption: 60 µA (typical)
- Package Type: TO-92, TO-220, SOIC
- Output Type: Analog
- Accuracy: ±1°C (typical)

2.3 Circuit Diagram





2.4 IoT(Internet of Things)

The term IoT, or Internet of Things, refers to the collective network of connected devices and the technology that facilitates communication between devices and the cloud, as well as between the devices themselves.

2.3.1 How does IoT work?

A typical IoT system works through the real-time collection and exchange of data. An IoT system has three components:

Smart devices

This is a device, like a television, security camera, or exercise equipment that has been given computing capabilities. It collects data from its environment, user inputs, or usage patterns and communicates data over the internet to and from its IoT application.

IoT application

An IoT application is a collection of services and software that integrates data received from various IoT devices. It uses machine learning or artificial intelligence (AI) technology to analyze this data and make informed decisions. These decisions are communicated back to the IoT device and the IoT device then responds intelligently to inputs.

A graphical user interface

The IoT device or fleet of devices can be managed through a graphical user interface. Common examples include a mobile application or website that can be used to register and control smart devices.

Chapter 3

PROGRAMMING

3.1 Code

```
#include <string.h>
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

void tostring(char [], int);
char str[10];
char str2[]="° Centigrade";

int sensorValue = 0;
int out = 0;   // Wifi network station credentials
#define WIFI_SSID "00000000000000"
#define WIFI_PASSWORD "97662995+" // Telegram BOT Token
#define BOT_TOKEN "6514168177:AAHefslxluTFy27f4sZ_6qUfXVdrUCLnGcg"
#define CHAT_ID "812065317"

X509List cert(TELEGRAM_CERTIFICATE_ROOT);
WiFiClientSecure secured_client;
UniversalTelegramBot bot(BOT_TOKEN, secured_client);

void setup() {
  Serial.begin(115200);
  Serial.println(); // attempt to connect to Wifi network:
  Serial.print("Connecting to Wifi SSID ");
  Serial.print(WIFI_SSID);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  secured_client.setTrustAnchors(&cert); // Add root certificate for
api.telegram.org
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    delay(500);
  }
  Serial.print("\nWiFi connected. IP address: ");
  Serial.println(WiFi.localIP());
```

```
        Serial.print("Retrieving time: ");
        configTime(0, 0, "pool.ntp.org"); // get UTC time via NTP
        time_t now = time(nullptr);
        while (now < 24 * 3600)
        {
          Serial.print(".");
          delay(100);
          now = time(nullptr);
        }
        Serial.println(now);

        bot.sendMessage(CHAT_ID, "DataLoggerBot started ", "");
      }

      void loop() {
        // read the analog in value:
        sensorValue = analogRead(analogOutPin);
        out = ((sensorValue*500)/1024);

         tostring(str, out);

         strcat(str,str2);

         bot.sendMessage(CHAT_ID, str, "");

         delay(600);
      }

      void tostring(char str[], int num)
      {
          int i, rem, len = 0, n;

          n = num;
          while (n != 0)
          {
              len++;
              n /= 10;
          }
          for (i = 0; i < len; i++)
          {
              rem = num % 10;
              num = num / 10;
              str[len - (i + 1)] = rem + '0';
          }
          str[len] = '\0';
}
```

3.2 Code Description

This code is an Arduino sketch written for ESP8266 that reads an analog sensor value and converts it to temperature in Celsius, and sends the data to a Telegram bot at regular intervals. Let's break down the code and then discuss a simplified flowchart.

Code Explanation:

Include Libraries:

The code includes several libraries: **string.h**, **ESP8266WiFi**, **WiFiClientSecure**, **UniversalTelegramBot**, and **ArduinoJson**. These libraries are used for handling strings, WiFi communication, secure client connections, Telegram bot communication, and JSON parsing, respectively.

Global Variables:

str and **str2** are character arrays used to store the temperature value and the unit ("° Celsius").

sensorValue stores the analog sensor reading.

out stores the calculated temperature value.

WiFi and Telegram credentials are defined for connecting to the internet and the Telegram bot.

Setup Function:

The **setup** function initializes the serial communication, connects to the WiFi network, sets up a secure client, and configures the time using NTP.

It sends a message to the Telegram bot indicating that the DataLoggerBot has started.

Loop Function:

In the **loop** function, it reads the analog sensor value and converts it to a temperature value in Celsius.

The temperature value is converted to a string and concatenated with the unit ("° Celsius").

The temperature value is sent as a message to the Telegram bot.

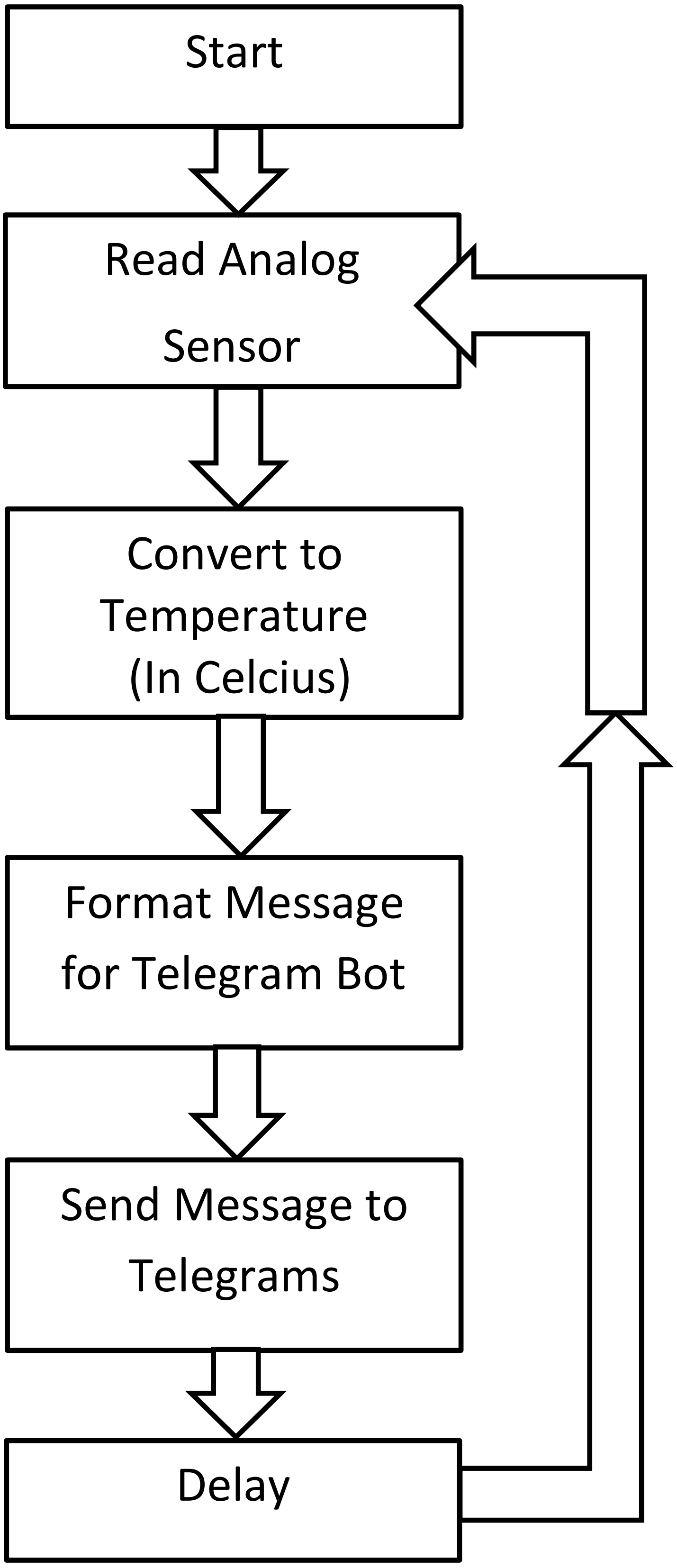
A delay of 600 milliseconds is introduced between successive readings.

tostring Function:

This function converts an integer **num** to a character array **str** using basic string manipulation.

3.3 Flowchart:

A simplified flowchart for the main loop would look like this:



Start
Read Analog Sensor
Convert to Temperature (In Celcius)
Format Message for Telegram Bot
Send Message to Telegrams
Delay

Chapter 4

TELEGRAM BOT

Telegram is one of the most popular instant messaging apps on the internet. Unlike WhatsApp, the platform is open-sourced and doesn’t require users to share their phone numbers.
Bots on Telegram are small applications that run entirely within the platform and can be designed to support any kind of task or service.

Bots are simply Telegram accounts operated by software – not people – and they'll often have AI features. They can do anything – teach, play, search, broadcast, remind, connect, integrate with other services, or even pass commands to the Internet of Things.

Just like Discord, Telegram supports third-party bots that offer additional functionality. These bots can be used to perform various tasks like converting files, checking emails and even letting users play games with others.
Here we created the bot to accept the data from NodeMCU and send it as a message to our telegram Id ,which can be seen on our smartphone.

➢ Creating Telegram Bot:

To create a telegram bot we have used the pre-existing bot available on telegram named “BotFather” which can be used to create new telegram bot.

We’ll have to send the commands to the bot according to the instructions given by bot itself. Following the instructions will create the bot.

Later we can use another bot available on telegram to redesign the functionality of our bot . By redesigning it we can make it more functional depending on our application.

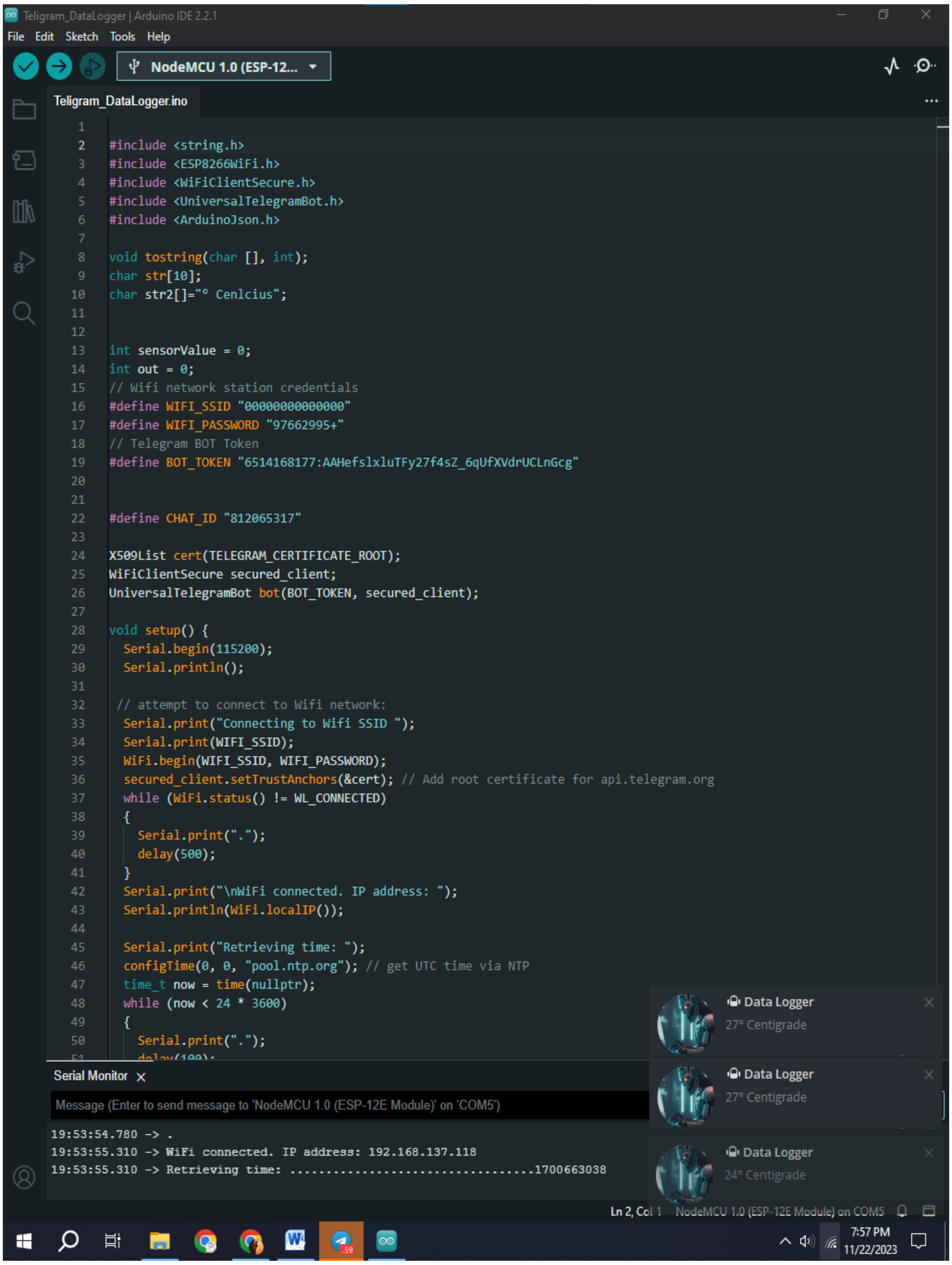
Here we are just receiving the telegram data so we don’t need any further modification in bot we’ll only create the bot and put bot’s details like token , chat ID in our code.

To control the bot we use the BOT’s HTTP API that is provided by the telegram . To access this API we require the token that we got from bot father at the time of creation of our bot.

The token used for accessing the HTTP API is so important to control the bot.

Chapter 5

RESULTS



```

#include <string.h>
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

void tostring(char [], int);
char str[10];
char str2[]="° Cenlcius";


int sensorValue = 0;
int out = 0;
// Wifi network station credentials
#define WIFI_SSID "00000000000000"
#define WIFI_PASSWORD "97662995+"
// Telegram BOT Token
#define BOT_TOKEN "6514168177:AAHefslxluTFy27f4sZ_6qUfXVdrUCLnGcg"


#define CHAT_ID "812065317"

X509List cert(TELEGRAM_CERTIFICATE_ROOT);
WiFiClientSecure secured_client;
UniversalTelegramBot bot(BOT_TOKEN, secured_client);

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

 // attempt to connect to Wifi network:
  Serial.print("Connecting to Wifi SSID ");
  Serial.print(WIFI_SSID);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  secured_client.setTrustAnchors(&cert); // Add root certificate for api.telegram.org
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    delay(500);
  }
  Serial.print("\nWiFi connected. IP address: ");
  Serial.println(WiFi.localIP());

  Serial.print("Retrieving time: ");
  configTime(0, 0, "pool.ntp.org"); // get UTC time via NTP
  time_t now = time(nullptr);
  while (now < 24 * 3600)
  {
    Serial.print(".");
    delay(100);
```

Message (Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)' on 'COM5')

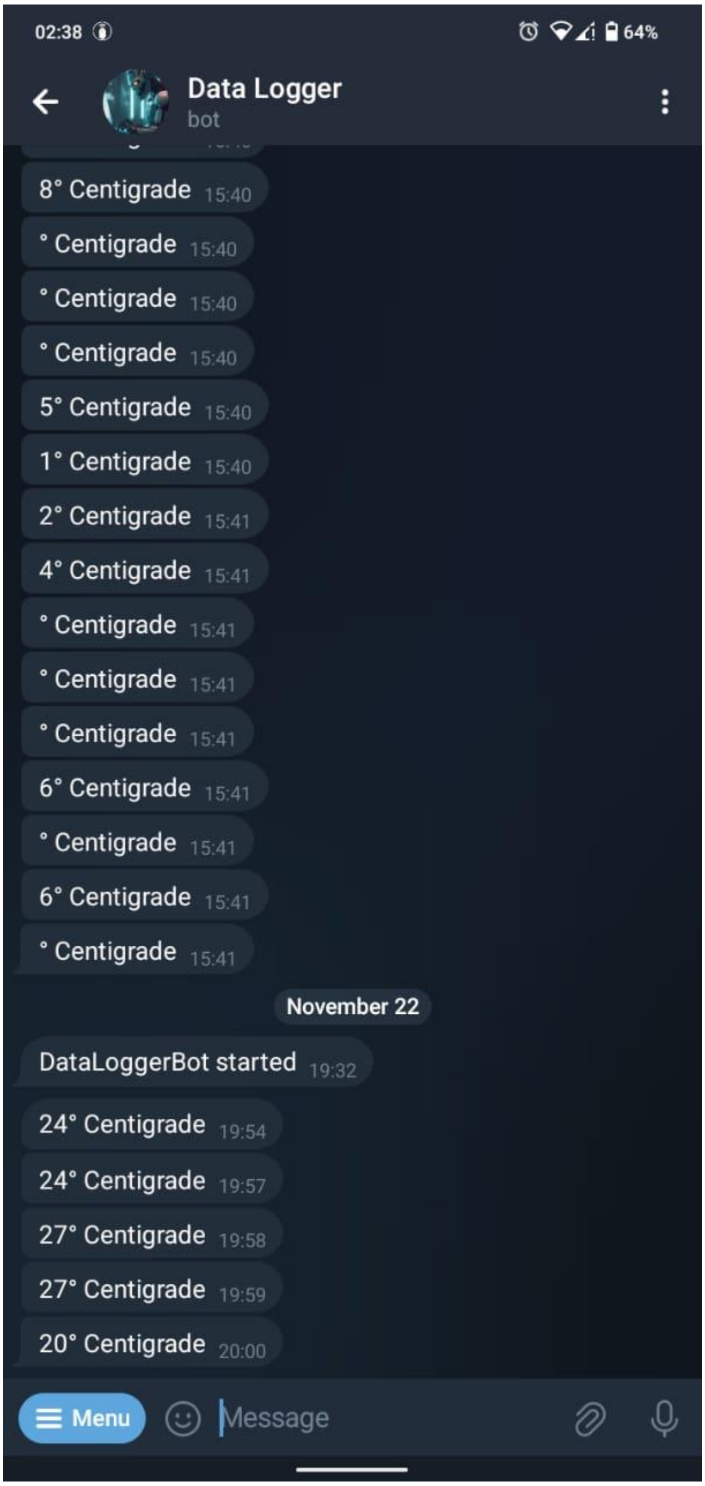
```
19:53:54.780 -> .
19:53:55.310 -> WiFi connected. IP address: 192.168.137.118
19:53:55.310 -> Retrieving time: ................................1700663038
```

Data Logger
27° Centigrade

Data Logger
27° Centigrade

Data Logger
24° Centigrade

Img: Computer screen with Arduino IDE AND Telegram running in background and receiving continuously sensor data.



8° Centigrade 15:40

° Centigrade 15:40

° Centigrade 15:40

° Centigrade 15:40

5° Centigrade 15:40

1° Centigrade 15:40

2° Centigrade 15:41

4° Centigrade 15:41

° Centigrade 15:41

° Centigrade 15:41

° Centigrade 15:41

6° Centigrade 15:41

° Centigrade 15:41

6° Centigrade 15:41

° Centigrade 15:41

November 22

DataLoggerBot started 19:32

24° Centigrade 19:54

24° Centigrade 19:57

27° Centigrade 19:58

27° Centigrade 19:59

20° Centigrade 20:00

Img: Smartphone screen of Telegram Application where we are continuously receiving the sensor Data.

Chapter 6

CONCLUSION

The main intention of this project is to implement the method for sending data collected by the microcontroller to the telegram bot . This data can be remotely accessed from anywhere through smartphone.

This example give an idea of so many other nice applications which can be implemented using this. Telegram is free cloud based messaging platform so it can be used as a free cloud server by some modification. This project give an basic example to do something related to this idea.

Throughout the implementation of this project we get a nice idea of what is the capabilities of a IoT applications based on NodeMCU .

Chapter 7

REFERENCES

- Universal Telegram Bot Library By **Brian Lough :**

https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot

- *Book :* Internet of Things with ESP8266 - By *Marco Schwartz*