An

Internship Assessment Report

on

# **“Developing IoT Applications”**

Submitted for partial fulfillment of award of

### BACHELOR OF TECHNOLOGY

Degree

IN

**Electronics and Communication Engineering**

By

HARSH RAJ

ROLL NO : 1813331049

Submitted to

 **Md. Sazid Mr. Ravi Pandey Assistant Professor, ECE Assistant Professor, ECE**

**NOIDA INSTITUTE OF ENGINEERING& TECHNOLOGY, GREATER NOIDA**

**January, 2021**

**DECLARATION**

I hereby declare that the internship assessment report which is being presented in the entitled, “**Developing IoT Applications**” in partial fulfillment of the requirements for the award of degree of **Bachelor of Technology** in Electronics and Communication Engineering in the Department of Electronics and Communication Engineering from Noida Institute of Engineering and Technology, Greater Noida, is an authentic record of my own carried out under the supervision of **Md. Sazid and Mr. Ravi Pandey** Assistant Professor, Department of ECE.

.

Date: 25/01/2021  **Harsh Raj**

1813331049

**Internship Assessment certificate**



**ABSTRACT**

“Cisco thingQbator” is a network of makerspaces in partner Universities where students can learn about digital technologies in a hands-on environment, turn their ideas into working prototypes and in the process come up with local solutions to local problems.

The name “thingQbator” is a combination of ‘Internet of Things’ and ‘Incubator’. thingQbator was started as an internal incubator of IoT ideas coming from Cisco’s engineer community. With the central belief that none of us is smarter than all of us, the aim was to create makerspaces where IoT enthusiasts could learn more about digital technologies in a hands-on environment and turn their ideas into working prototypes.

Inspired by the success of our internal makerspaces Cisco has come up with the concept of “thingQbator for Universities” to help universities and academic institutions to set up IoT makerspaces to help them accelerate innovation around IoT and digital technologies in their campus.

**ACKNOWLEDGEMENT**

I have taken efforts in this Internship Assessment. However, it would not have been possible without the kind support and help of many individuals and organizations. I am grateful to the staff and management for letting me pursue my four weeks industrial training at the department of Internet Of Things.

I am grateful for the explanation given to me by the tutor of **Cisco thingQbator program** and helping me become aware of the technology in the real world. The training I had received here has helped me immensely in understanding the practical concepts of IOT. I would like to extend my sincere thanks to all of them.

**TABLE OF CONTENT**

**Page No.**

Declaration ii

Abstract iii

Certificate iv

Acknowledgement v

Table of content vi

List of figures viii

**1 AN OVERVIEW ABOUT COMPANY**

**2 INTRODUCTIONS TO IOT**

* 1. Introduction
  2. DEFITNITION OF INTERNET OF THINGS (IoT)
  3. ARCHITECTURE OF INTERNET OF THINGS
  4. APPLICATIONS
  5. BENEFITS OF INTERNET OF THINGS

1. **ESP8266 NodeMCU** 
   1. Introduction to ESP-12E Module
   2. Peripherals and I/O
   3. ESP8266 NodeMCU Pinout

**4 DEVELOPING AN IOT APPLICATION**

4.1 Creating a channel to transmit data

4.2 Installing Arduino IDE and configuring for NodeMCU

4.3 Coding on the Arduino IDE

**5 CONCLUSION**

**6 BIBLIOGRAPHY**

**AN OVERVIEW ABOUT COMPANY**

“Cisco thingQbator” is a network of makerspaces in partner Universities where students can learn about digital technologies in a hands-on environment, turn their ideas into working prototypes and in the process come up with local solutions to local problems.

Inspired by the success of our internal makerspaces Cisco has come up with the concept of “thingQbator for Universities” to help universities and academic institutions to set up IoT makerspaces to help them accelerate innovation around IoT and digital technologies in their campus

**INTRODUCTION TO INTERNET OF THINGS (IoT)**

Anyone who says that the Internet has fundamentally changed society may be right, but at the same time, the greatest transformation actually still lies ahead of us. Several new technologies are now converging in a way that means the Internet is on the brink of a substantial expansion as ob­jects large and small get connected and assume their own web identity.  
Following on from the Internet of computers, when our servers and personal computers were connected to a global network, and the Internet of mobile telephones, when it was the turn of telephones and other mobile units, the next phase of development is the Internet of things, when more or less anything will be connected and managed in the virtual world. This revolution will be the Net’s largest enlargement ever and will have sweeping effects on every industry — and all of our everyday lives.  
Smart connectivity with existing networks and context-aware computation using network resources is an indispensable part of IoT. With the growing presence of Wi-Fi and 4G-LTE wireless Internet access, the evolution towards ubiquitous information and communication networks is already evident. However, for the Internet of Things vision to successfully emerge, the computing paradigm will need to go beyond traditional mobile computing scenarios that use smart phones and portables, and evolve into connecting everyday existing objects and embedding intelligence into our environment.

For technology to disappear from the consciousness of the user, the Internet of Things demands: a shared understanding of the situation of its users and their appliances, software architectures and pervasive communication networks to process and convey the contextual information to where it is relevant, and the analytics tools in the Internet of Things that aim for autonomous and smart behavior. With these three fundamental grounds in place, smart connectivity and context-aware computation can be accomplished.  
A radical evolution of the current Internet into a Network of interconnected objects that not only harvests information from the environment (sensing) and interacts with the physical world (actuation/ command/control), but also uses existing Internet standards to provide services for information transfer, analytics, applications, and communications. Fueled by the prevalence of devices enabled by open wireless technology such as Bluetooth, radio frequency identification (RFID), Wi-Fi, and telephonic data services as well asembedded sensor and actuator nodes, IoT has stepped out of its infancyand is on the verge of transforming the current static Internet into a fully integrated Future Internet.

**DEFITNITION OF INTERNET OF THINGS (IoT)**

“Today computers and the Internet are almost wholly dependent on human beings for information. Nearly all of the roughly 50 petabyte (1 petabyte=1015 bytes) of data available on the Internet were first captured and created by human beings by typing, pressing a record button, taking a digital picture, or scanning a bar code. Conventional diagrams of the Internet leave out the most numerous and important routers of all - people. The problem is, people have limited time, attention and accuracy all of which means they are not very good at capturing data about things in the real world. And that's a big deal. We're physical, and so is our environment … You can't eat bits, burn them to stay warm or put them in your gas tank. Ideas and information are important, but things matter much more. Yet today's information technology is so dependent on data originated by people that our computers know more about ideas than things. If we had computers that knew everything there was to know about things using data they gathered without any help from us we would be able to track and count everything, and greatly reduce waste, loss and cost. We would know when things needed replacing, repairing or recalling, and whether they were fresh or past their best. The Internet of Things has the potential to change the world, just as the Internet did or even more.

**ARCHITECTURE OF INTERNET OF THINGS**

Architecture of internet Of Things contains basically 4 layers:

* Application Layer
* Gateway and the network layer
* Management Service layer
* Sensor layer

APPLICATION LAYER:

* Lowest Abstraction Layer
* With sensors we are creating digital nervous system.
* Incorporated to measure physical quantities
* Interconnects the physical and digital world
* Collects and process the real time information

GATEWAY AND THE NETWORK LAYER:

* Robust and High-performance network infrastructure
* Supports the communication requirements for latency, bandwidth or security
* Allows multiple organizations to share and use the same network independently

MANAGEMENT LAYER:

* Capturing of periodic sensory data
* Data Analytics (Extracts relevant information from massive amount of raw data)
* Streaming Analytics (Process real time data)
* Ensures security and privacy of data.

SENSOR LAYER:

* Provides a user interface for using IoT.
* Different applications for various sectors like Transportation, Healthcare, Agriculture, Supply chains, Government, Retail etc.

**APPLICATIONS**

There are several application domains which will be impacted by the emerging Internet of Things. The applications can be classified based on the type of network availability, coverage, scale, heterogeneity, repeatability, user involvement and impact.

We categorize the applications into four application domains:

(1) Personal and Home

(2) Enterprise

(3) Utilities

(4) Mobile.

There is a huge crossover in applications and the use of data between domains. For instance, the Personal and Home IoT produces electricity usage data in the house and makes it available to the electricity (utility) company which can in turn optimize the supply and demand in the Utility IoT. The internet enables sharing of data between different service providers in a seamless manner creating multiple business opportunities.

**BENEFITS OF INTERNET OF THINGS**

* Improved citizen's quality of life

Healthcare from anywhere

Better safety, security and productivity

* New business opportunities

IoT can be used in every vertical for improving the efficiency

Creates new businesses, and new and better jobs

* Economic growth

Billions of dollars in savings and new services

* Better environment

Saves natural resources and trees

Helps in creating a smart, greener and sustainable planet

**ESP-12E Module**

The development board equips the ESP-12E module containing ESP8266 chip having Tensilica Xtensa® 32-bit LX106 RISC microprocessor which operates at 80 to 160 MHz adjustable clock frequency and supports RTOS.

ESP-12E Chip

* Tensilica Xtensa® 32-bit LX106
* 80 to 160 MHz Clock Freq.
* 128kB internal RAM
* 4MB external flash
* 802.11b/g/n Wi-Fi transceiver

ESP8266 NodeMCU Hardware Specifications - ESP-12E Chip

There’s also 128 KB RAM and 4MB of Flash memory (for program and data storage) just enough to cope with the large strings that make up web pages, JSON/XML data, and everything we throw at IoT devices nowadays.

The ESP8266 Integrates 802.11b/g/n HT40 Wi-Fi transceiver, so it can not only connect to a WiFi network and interact with the Internet, but it can also set up a network of its own, allowing other devices to connect directly to it. This makes the ESP8266 NodeMCU even more versatile.

Power Requirement

As the operating voltage range of ESP8266 is 3V to 3.6V, the board comes with a LDO voltage regulator to keep the voltage steady at 3.3V. It can reliably supply up to 600mA, which should be more than enough when ESP8266 pulls as much as 80mA during RF transmissions. The output of the regulator is also broken out to one of the sides of the board and labeled as 3V3. This pin can be used to supply power to external components.

Power Requirement

* Operating Voltage: 2.5V to 3.6V
* On-board 3.3V 600mA regulator
* 80mA Operating Current
* 20 µA during Sleep Mode

ESP8266 NodeMCU Hardware Specifications - Power Supply

Power to the ESP8266 NodeMCU is supplied via the on-board MicroB USB connector. Alternatively, if you have a regulated 5V voltage source, the VIN pin can be used to directly supply the ESP8266 and its peripherals.

**Peripherals and I/O**

The ESP8266 NodeMCU has total 17 GPIO pins broken out to the pin headers on both sides of the development board. These pins can be assigned to all sorts of peripheral duties, including:

* ADC channel – A 10-bit ADC channel.
* UART interface – UART interface is used to load code serially.
* PWM outputs – PWM pins for dimming LEDs or controlling motors.
* SPI, I2C & I2S interface – SPI and I2C interface to hook up all sorts of sensors and peripherals.
* I2S interface – I2S interface if you want to add sound to your project.

Multiplexed I/Os

* 1 ADC channels
* 2 UART interfaces
* 4 PWM outputs
* SPI, I2C & I2S interface

ESP8266 NodeMCU Hardware Specifications - Multiplexed GPIO pins

Thanks to the ESP8266’s pin multiplexing feature (Multiple peripherals multiplexed on a single GPIO pin). Meaning a single GPIO pin can act as PWM/UART/SPI.

On-board Switches & LED Indicator

The ESP8266 NodeMCU features two buttons. One marked as RST located on the top left corner is the Reset button, used of course to reset the ESP8266 chip. The other FLASH button on the bottom left corner is the download button used while upgrading firmware.

Switches & Indicators

* RST – Reset the ESP8266 chip
* FLASH – Download new programs
* Blue LED – User Programmable

ESP8266 NodeMCU Hardware Specifications - Reset Flash Buttons & LED Indicators

The board also has a LED indicator which is user programmable and is connected to the D0 pin of the board.

Serial Communication

The board includes CP2102 USB-to-UART Bridge Controller from Silicon Labs, which converts USB signal to serial and allows your computer to program and communicate with the ESP8266 chip.

**ESP8266 NodeMCU Pinout**

The ESP8266 NodeMCU has total 30 pins that interface it to the outside world. The connections are as follows:

ESP-12E Development Board ESP8266 NodeMCU Pinout

For the sake of simplicity, we will make groups of pins with similar functionalities.

Power Pins :

There are four power pins viz. one VIN pin & three 3.3V pins. The VIN pin can be used to directly supply the ESP8266 and its peripherals, if you have a regulated 5V voltage source. The 3.3V pins are the output of an on-board voltage regulator. These pins can be used to supply power to external components.

GND :

is a ground pin of ESP8266 NodeMCU development board.

12C Pins:

are used to hook up all sorts of I2C sensors and peripherals in your project. Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

GPIO Pins ESP8266 NodeMCU :

has 17 GPIO pins which can be assigned to various functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button programmatically. Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.

ADC Channel the NodeMCU :

is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC viz. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.

UART Pins ESP8266 NodeMCU :

has 2 UART interfaces, i.e. UART0 and UART1, which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps.

SPI Pins ESP8266:

features two SPIs (SPI and HSPI) in slave and master modes. These SPIs also support the following general-purpose SPI features:

* 4 timing modes of the SPI format transfer
* Up to 80 MHz and the divided clocks of 80 MHz
* Up to 64-Byte FIFO

SDIO Pins ESP8266 :

features Secure Digital Input/Output Interface (SDIO) which is used to directly interface SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.

PWM Pins:

The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 μs to 10000 μs, i.e., between 100 Hz and 1 kHz.

Control Pins:

are used to control ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

EN pin:

The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.

RST pin:

RST pin is used to reset the ESP8266 chip.

WAKE pin:

Wake pin is used to wake the chip from deep-sleep.

**DEVELOPING AN IOT APPLICATION**

Objectives

* Learn about the cloud
* Create a channel to transmit data
* Visualize data graphically
* Things

For this project, Components needs

* NodeMCU board (quantity: 1 no.)
* Micro USB cable A to B (quantity: 1 no.)
* A Cloud platform (quantity: 1 no.)
* WiFi router connected to the internet (quantity: 1 no.)

Let’s begin!

To understand the cloud and visualize the data, we will use NodeMCU and transmit data to the cloud using a readily available IoT analytics platform service. Here, we use ThingSpeak, a service that helps transmit data to the cloud without having to set up servers or develop a web software.

NodeMCU is a form of ESP8266 using the ESP-12 module, which has an on-board microcontroller. It can directly be programmed using the Arduino IDE.

ThingSpeak is an open source IoT framework and API (Application Program Interface) to store and retrieve data from “things” using the HTTP protocol over the Internet or via a Local Area Network (LAN). It enables the creation of applications handling sensor logging, location tracking, social network of things with status updates and data visualisation in the form of graphs.

ThingSpeak has integrated support from the numerical computing software MATLAB from MathWorks, allowing users to analyze and visualize uploaded data using MATLAB without requiring the purchase of the software license. To get started with ThingSpeak, we need to create an account.

* Open the official website of ThingSpeak and click on “Get Started For Free”. You will be directed to the Sign Up page. Fill in your details and click on continue.
* Once this is done, you’ll receive a verification link on your registered email-id.
* After verification, you can Sign In to your personal ThingSpeak account. It’s time to transmit data to the CLOUD!

**Creating a channel to transmit data**

To transmit data to the cloud, we need to create a transmitting channel. For this project, we will use sample data and transmit it to the cloud.

* To start, navigate to Channels -> MyChannels. Click on ‘New Channel’ to create your first channel to sent data to.
* Add a suitable name, description and assign the required number of fields as shown in the screenshot below.
* Each ‘Field’ will contain a specific kind of data.
* For example, to monitor temperature and humidity of a greenhouse, temperature data is logged to one field and humidity to another field.
* The number of fields corresponds to the number of different types of data you want to send to your channel.
* Click on ‘Save Channel‘. You have now created a channel. Explore to see what is possible!

**Installing Arduino IDE and configuring for NodeMCU**

To install the required files to program the NodeMCU using the Arduino IDE, follow the steps below.

Step 1: Download Arduino IDE from the official Arduino website: Download

Step 2: Install the software and connect the NodeMCU board to your computer using the micro-USB cable.

Step 3: Start Arduino IDE and navigate to File -> Preferences.

Step 4: Copy the link below and paste it in the Additional Boards Manager URLs tab.

http://arduino.esp8266.com/stable/package\_esp8266com\_index.json

Click OK to close the preferences window.

Step 5: Navigate to Tools -> Board -> Boards Manager.

Step 6: In the Boards Manager window, type esp8266 by ESP8266 Community. Select it and install it onto the Arduino IDE

Step 7: Navigate back to Tools -> Board, find and select NodeMCU 1.0 (ESP-12E module) board.

Step 8: Go to Tools -> Port and choose the port that corresponds to your NodeMCU.

**CODING on the Arduino IDE**

In order to execute this project, we will use the following functions to code in the Arduino IDE.

void setup ()

* Any command you type in the void setup () function will only run once.
* Syntax:

void setup ()

{

Type your command here

void loop ()

* Any command you type in the void loop () starts executing after void setup () is finished. The commands here will run infinite number of times unless you specify an exit condition to it.
* Syntax:

void loop ()

{

Type your command here;

}

int

* Integers (int) are your primary data-type for pin numbers on Arduino. They can also be used for data storage.
* Syntax: int var = val
* ‘var’ indicates the int variable name.
* ‘val’ indicates the value you assign to that variable (a pin number in most cases).

For this project: the following command is used to initialize all the pins and variables in the code.

* int val1=1;
* int val2=0;
* int state = 1;

Note:

* These variables are defined as int before void setup () in the Arduino IDE. Such variables which are defined outside void setup () and void loop () are called global variables

delay ()

* A command delay () pauses the program for a definite period of time.
* Syntax: delay(ms)
* ‘ms’ is the number of milliseconds you want the delay for.
* For this project: the following command pauses the code for 0.5 second.

delay (500);

while ()

* A command while () will loop continuously, and infinitely, until the expression inside the parenthesis () becomes false.
* Syntax: while(condition)
* ‘condition’ is a Boolean expression that evaluates to true or false.

For this project: the following command will loop continuously, and infinitely, until the expression inside the parenthesis, () becomes false.

* while (WIFI. Status ()!= WL\_CONNECTED)

if ()

* The if () function checks for a condition and executes the proceeding statement or set of statements if the condition is ‘true’.
* Syntax:

if(condition)

{

Type your command here;

}

* ‘condition’ is a Boolean expression i.e., can be true or false.

For this project: the following command checks for a condition and executes the proceeding statement or set of statements if the condition is ‘true’.

if (state==1) {

else ()

* An else () clause (if at all exists) will be executed if the condition in the if statement results in false.
* The else () can proceed another if () test, so that multiple, mutually exclusive tests can be run at the same time.
* Syntax: else ()

For this project: the following command the clause will be executed if the condition in the if statement results in false.

else {

ThingSpeak.writeField(myChannelNumber, 2, val2, myWriteAPIKey);

Serial.begin()

* A command Serial.begin() sets the data rate in bits per second (baud) for serial data transmission.
* Syntax: Serial.begin(speed)
* ‘speed’ indicates the speed in bits per second (baud) – long.
* For this project: the following command set the baud rate to 115200.

Serial.begin(115200);

Serial.print()

* A command Serial.print() prints data to the serial port as human-readable ASCII text.
* Syntax: Serial.print(val)
* ‘val’ indicates the value to print – any data type.

For this project: the following command prints the ldr value to the serial port.

Serial.print("Connecting to ");

Serial.println()

A command Serial.println() prints data to the serial port as human-readable ASCII text followed by a carriage return character (ASCII 13, or ‘\r’) and a newline character (ASCII 10, or ‘\n’).

Syntax: Serial.println(val)

‘val’ indicates the value to print – any data type.

For this project: the following command prints the map\_ldr value to the serial port.

Serial.println();

For this project: we will now summarize the above mentioned functions.

Using delay(), while(), Serial.begin(), Serial.print() and Serial.println(), the void setup() will look like this.

void setup() {

Serial.begin(115200);

Serial.println();

Serial.println();

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

while(WiFi.status() != WL\_CONNECTED)

{

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

Serial.println(WiFi.localIP());

ThingSpeak.begin(client);

}

Using delay(), if(), else() and Serial.println(), the void loop()will look like this.

void loop() {

if(state==1) {

ThingSpeak.writeField(myChannelNumber, 1, val1, myWriteAPIKey);

state = state+1;

val1=val1+2;

Serial.println(state);

Serial.println("Channel 1 transmitted");

}

else {

ThingSpeak.writeField(myChannelNumber, 2, val2, myWriteAPIKey);

Serial.println(state);

Serial.println("channel 2 transmitted");

state = 1;

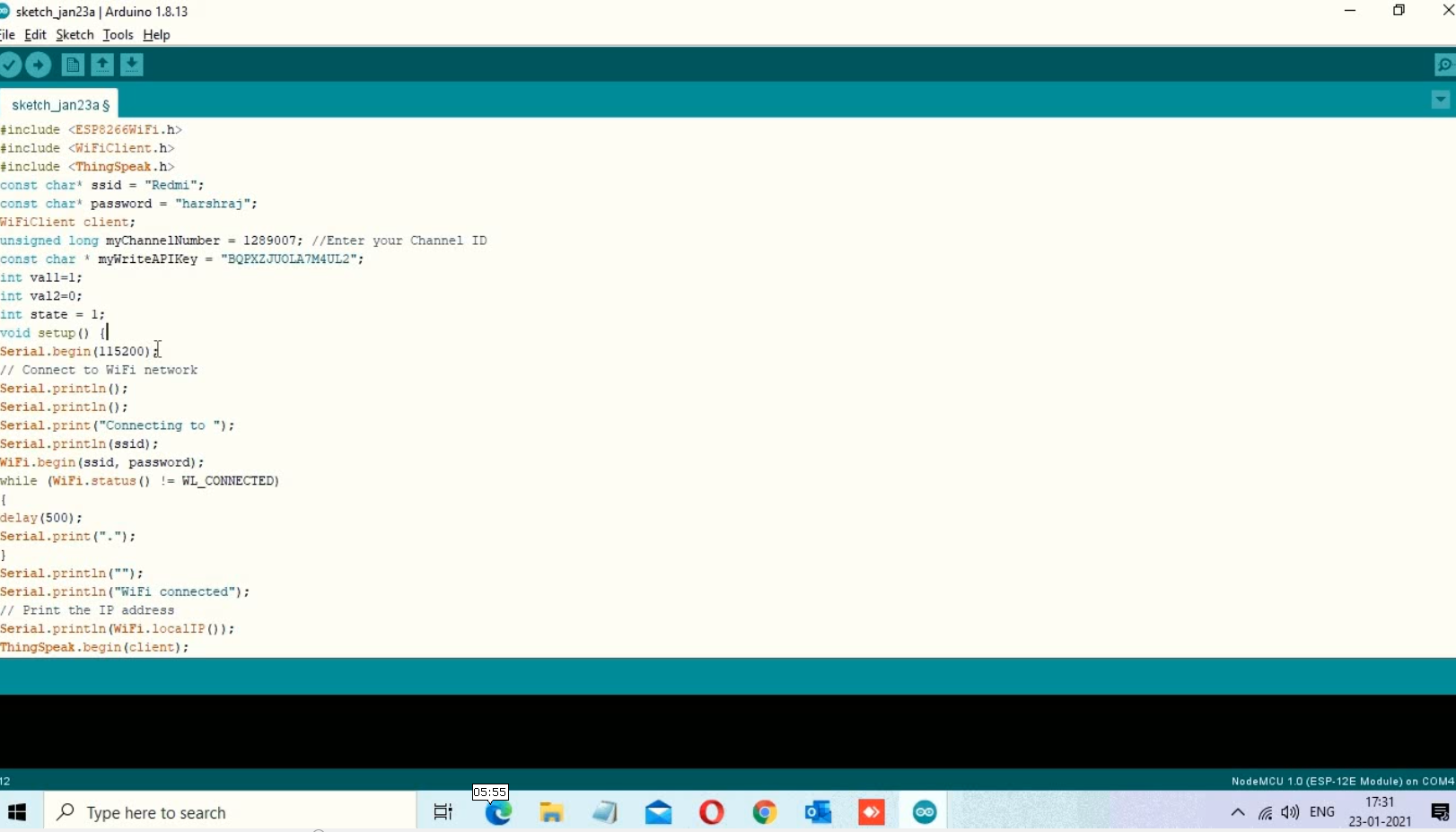
val2=val2+2;

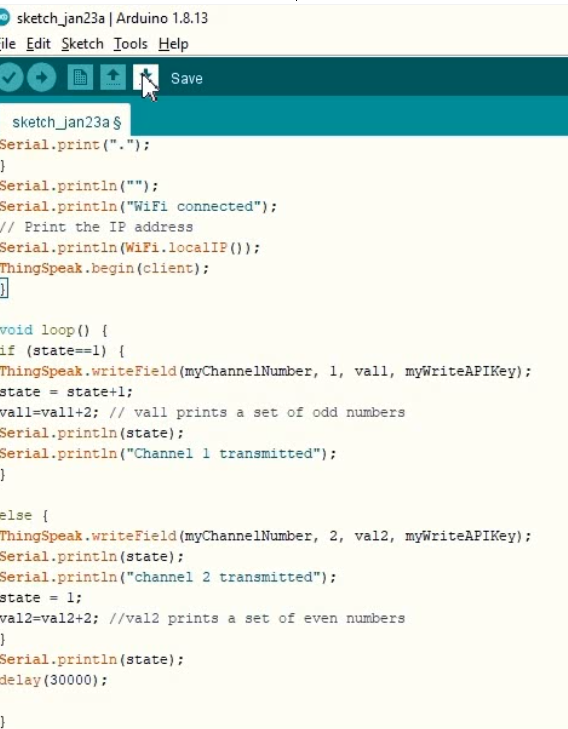
}

Serial.println(state);

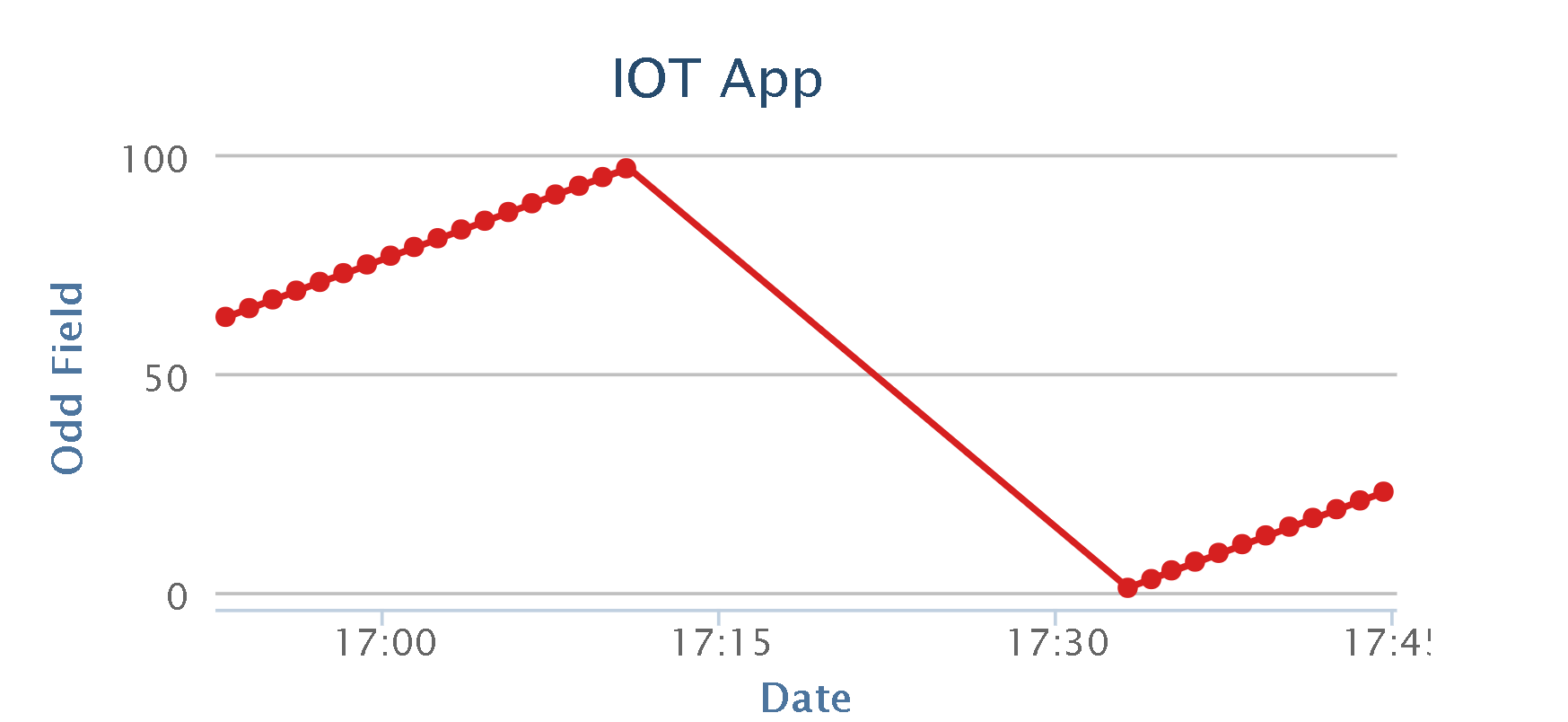
delay(30000);

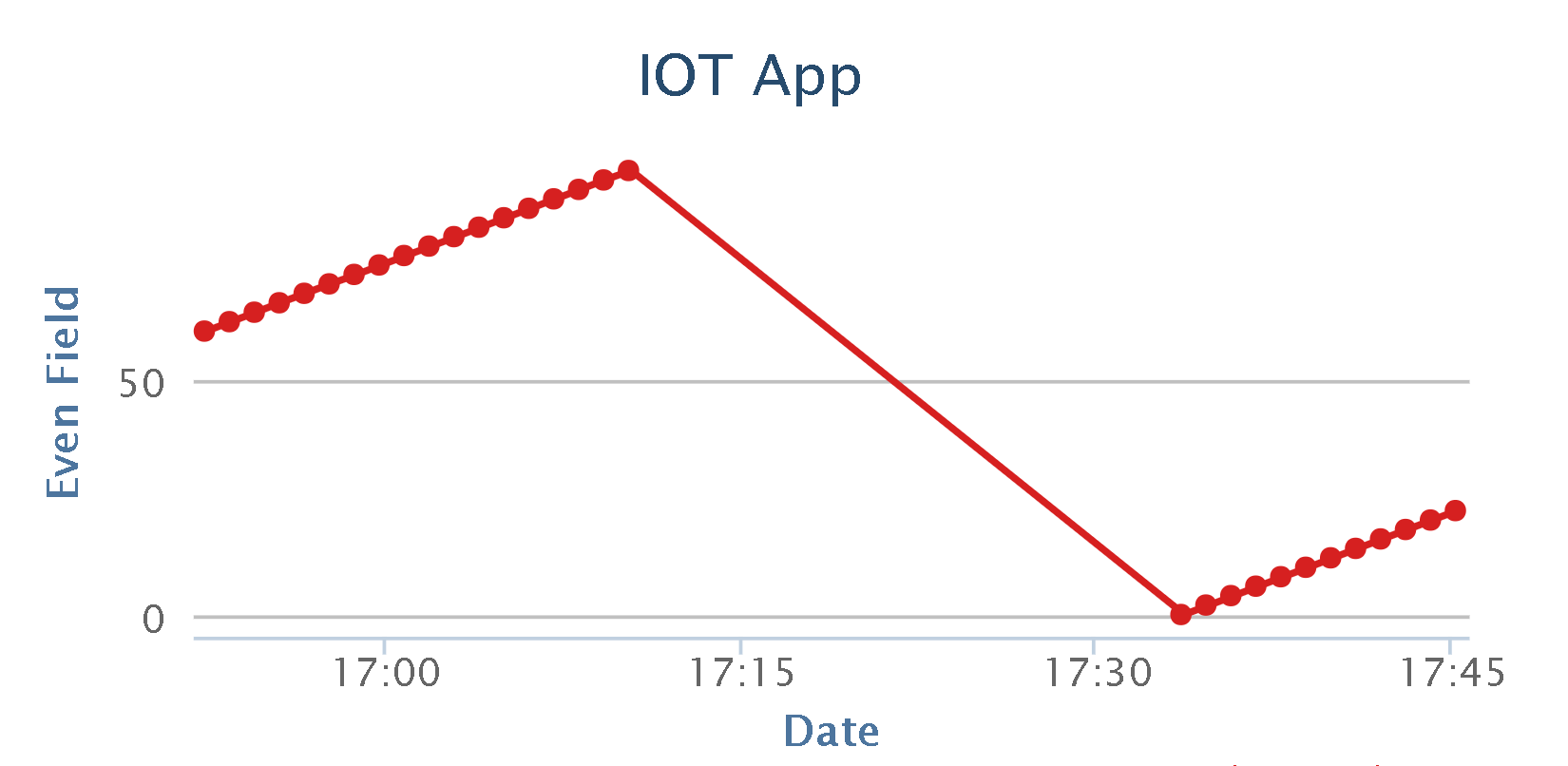
}

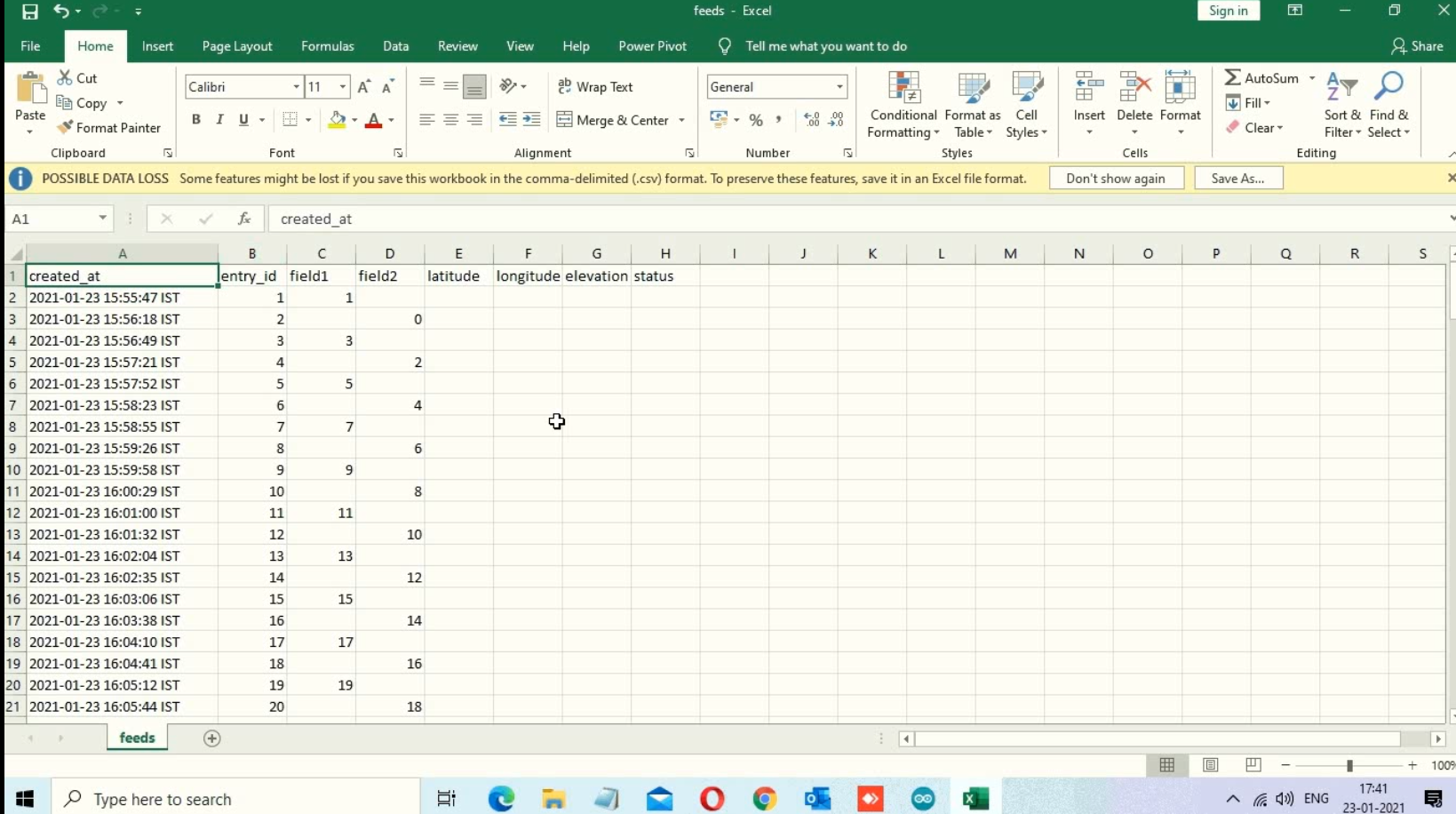




Graph : Data Analysis







**CONCLUSION**

By Enabling ESP8266 We Can upload signal data to the cloud. Access the data on your cloud and visualize the outputs. Use internet communication protocols like REST to retrieve data from cloud and develop web application.

We will know about how to plot signal readings on a web server using the ESP8266 with Arduino IDE. The ESP will host a web page with real time charts that have new readings added every 30 seconds.

Building an IoT based Cloud Application.

Access the data on your cloud and visualize the outputs.

Use internet communication protocols like REST.

Make your web application portable with your ESP8266’s web server.

**BIBLIOGRAPHY**

1. *Circuits4you.com*
2. *https://community.thingspeak.com*
3. *Cisco thingQbator – Cisco*
4. *www.electronicshub.org/connect-esp8266-to-thingspeak/*