ABSTRACT

Taking traditional manual pen-paper attendance, which is very time-consuming, insecure and usually leads to human errors as well as prone to misconduct, as the valuable time and work gets wasted in organizing and structuring the attendance data in registers. Hence to overcome this major hectic problem, we have used relational database management system in real time with appropriate security measures to access, manipulate and represent the data on the basis of the unique RFID tags, which gets fast and easily scanned on the RFID reader. This system consists of hardware and software with most trending implementation of a lightweight MQTT protocol in IoT technology; designed to take an attendance on the basis of RFID technology with NodeMCU firmware. The main objective of this proposed system is to make the effective and efficient computerized attendance on cloud platform (Adafruit). This proposed system is an entirely green and clean way and novel approach to flourishing in the IoT era.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Attendance is a state of being present or an evidence of presence of the students in educational institutions, since penalty claimed for the students having below 80% present for that attendance record. This is a significant aspect to make future bright for students.

1.1 PROBLEM DEFINITION

Existing system is traditional and manual pen-paper work attendance, where lecturers have to take attendance every time of class or session in 3 ways, which are time wasting, ineffective, inefficient. They are

1) Calling name or roll number on the register, which is waste of teaching time.

2) Blank paper which is circulated by students during lecture. There may be chance of proxy intentionally or sometimes any student may forget to make attendance.

3) Attendance sheet which already having name, roll number needs to be signed by the students, which distract the students from the lecture concentration to find the name for making attendance.

Valuable time and work get wasted to organize and structure the attendance data in registers, needed to put and maintain that attendance data to proper location for future references. It requires more papers, registers and cost for it. Wasting valuable quality time of teaching leads to loss of education. This is not systematic as missing name of any student, proxy false attendance, chance of sheet loss or damage may possible. So, this pen paper work attendance method, impacted on four factors: time consuming, space consuming, cost consuming and data loss. This conventional attendance is insecure, inefficient and leads to human error.

1.2 LITERATURE SURVEY

It is recommended to use ESP8266 module for WIFI purpose in order to collect, record and process the data of participants of any event/conference as using MRF24WB0MA is three times more expensive **[1]**.

When the RFID reader reads the tag and transmits the tag's data to the microcontroller. After comparing the stored ID, the attendance is then showed on the LCD (Liquid Crystal Display) and the data is transferred to the PC via RS232 port **[2]**.

According to these papers **[3][4]**, There are many attendance systems designed with RFID, which improve the manual method. It captures the attendance of the students by flashing the tags over the RFID reader.

The low-cost attendance monitoring system was implemented using RFID and IOT with the Cloud. To avoid proxy, the simple image comparison technique is used. It totally works on an image comparison algorithm by considering RGB values of the pixels. The student's attendance stored in the cloud database. Wi-Fi adapter and Cloud are less expensive than the Desktop computer and need less maintenance **[5]**.

The concept of "IoT" has growing highly and attractively attention in education as well as industry while transmitting data over the network through computer communication without human interaction **[6]**. In this research project, making use of trending IoT fields, i.e. Wireless RFID technology and NodeMCU firmware has been used along with new MQTT protocol.

[7] proposed time attendance system using RFID technology and window services to made some facilities taking parents in account, sending of SMS, email, reports with concern. This system discussed various issues like proxy, weightage, probation analysis, submission warning and teaching time loss.

[2] focuses on the issue of student irregular attendance. This system used RFID technology with microcontroller and tends to implemented in school, colleges and universities. It is an automated scanning system that is represented by parallel line of different width which stores information regarding an object. The proposed system uses real time clock function to gain accurate results, and viewed using software called HyperTerminal, which is connected to computer using RS232 (USB) to store the data into the database. The system tends to be vital with physical advantages as low cost, light weight, user friendly, compact design and portable.

[8] focuses on the factors such as reliability, time saving and easy control. This system defined prototype using RFID system. This implementation only applicable for small scale setup. The further attempt will be made to modify the RFID for completing application on a large scale. The application has three modules student faculty and administrator.

[4] described web-based attendance for academic performance and progress uses MySQL database, focuses on time consuming and inefficient method of conventional manual attendance. It consists of three module RFID reader, data reporter and web server module and works on TCP/IP protocol.

[3] developed system safe and secure four-tier architecture using biometric, GSM and RFID technology, records attendance of students and staff members and tracks their location in campus. The SMS and email facilities were proposed for lagging attendances for both students and parents. It has been developed with desktop and android version for remote access. System gives student's consistency graph throughout semester.

[6] developed attendance system to reduce time consumption by traditional attendance. It's implementation of IoT through raspberry Pi 3 and RFID technology to make automated one and developed android app for students can check their attendance anywhere.

[1] developed using RFID based on the mobile communications and IT technologies, to put into use at any gatherings such as conferences, exhibitions, training courses, etc. and scaled from small to large venues depending on the need. The system is designed to generate real time consolidated reports on attendance, which collects records and processes data on participants of any conference or event. This paper suggests to use ESP8266 for low cost WIFI chip purpose as MRF24WB0MA module were used for development, which is three times costlier.

[9] presents RFID reader as established approach for recording attendance system, used SD card and the Thingspeak API cloud server, as it provides real time data collection with ESP8266 WIFI module.

The proposal of merging the IoT and RFID technology to establish attendance system is discussed in number of researches. There are many papers presented for making advance effective, efficient and reliable system.

1.3 INTERNET OF THINGS CONCEPT

The Internet of Things (IoT) is a concept of revolutionary change taking place to promises transform in world of interrelated computing devices by transmitting data and automate tasks without human intervention. IoT invokes opposing emotions. So, it is nothing but connecting physical devices over the internet to make communication simple and easier.

1.4 OVERVIEW OF RFID AND NODEMCU

RFID (Radio Frequency Identification) technology is one of the types of Automatic Identification and Data Capture (AIDC) method. As AIDC methods automatically identify objects, collect data about them, gets that data directly into computer system, with little or no human involvement. RFID technology uses radio frequency electromagnetic fields to transfer information from an RFID tag to RFID reader using 13.56MHz electromagnetic field for identification purpose.

RFID system comprises of RFID reader, RFID tags and antenna, is used to transmit and receive information through radio waves at a distance without wires. RFID reader can be grouped into three categories: low frequency, high frequency and ultra-high frequency. As per need, RFID tags maybe active or passive tags.

Compare with barcode - RFID provides fast identification, no line of sight, reusable to rewrite or update, higher data storage, higher read rate, multiple reading, durable, maximum distance reading without interference.

In comparison with other identification technologies - RFID technology offers quality, security, supply chain optimization, Better individual identification, better storage flexibility, real time information and better ergonomics.

Recently, UP state has made mandatory this technology for vehicle operating to use RFID tags along with E-way bill from 1 Nov, 2018. NodeMCU is an open source firmware and development board with ESP8266 helps to build IoT products. ESP8266 WIFI module is a low-cost microcontroller with WIFI capability and full TCP/IP stack to make easier IoT platform. NodeMCU have built-in support for wireless networks compared to Arduino and Raspberry Pi.

1.5 MQTT PROTOCOL

MQTT (message queuing telemetry transport) protocol is a lightweight, small in size, publish-subscribe network protocol that transports messages between devices with low bandwidth environments. So, its perfect solution for IoT applications due to its feature. MQTT uses transport layer security (TLS/SSL) encryption with bidirectional connection support. This simple messaging protocol allows to connect, disconnect, subscribe and publish to subscribed one on request with acknowledgment. We have three options for hosting an MQTT broker. Use own locally installed broker/server, which is we have used. Second use of cloudbased server or virtual server and third one by use of shared server application.

1.6 PROJECT OBJECTIVE

To design and build a prototype of IOT based attendance system using RFID and ESP8266, and display the data in cloud platform (Adafriut).

1.7 PROPOSED SYSTEM AND METHODOLOGY

RFID Technology – This technology permits devices to identify and capture the certain unique information recorded on a tag using radio waves. Compared to the barcodes, RFID provides faster identification, no line of sight, reusable to rewrite or update higher data storage, higher read rate, multiple tag readings at a time, toughness, extreme distance reading without intrusion, cost effective. The best features of RFID are its speed, function, performance over bar code. RFID module is used as input to the NodeMCU, which consists of RFID Reader and RFID Tags with antenna.

NodeMCU WIFI (Node Micro-Controller Unit Wireless Fidelity) Module – It is an easily available IoT firmware in addition with the development board. It is a cheap microcontroller with inbuilt ESP8266 WIFI capability to communicate and aid full TCP/IP stacks and assist to build easier on IoT platform. This module is interfaced with RFID

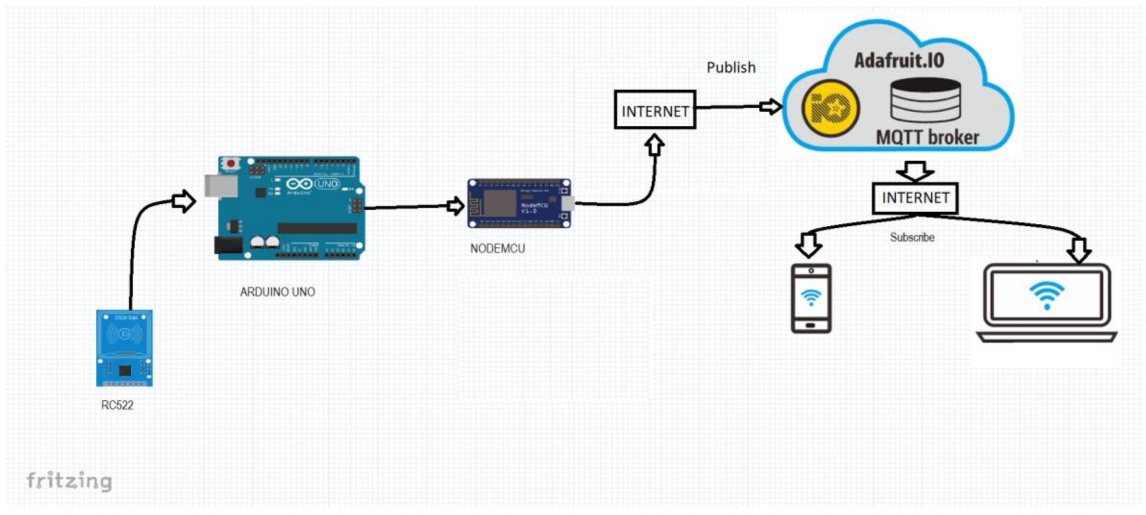


Figure 1.1 Proposed System

Arduino and RFID scanner scans the RFID cards and then log the data to Adafruit IO cloud platform with the help of ESP8266 Wi-Fi module. This information can be displayed in the Adafruit IO dashboard as we see in fig 1.1 and can be accessed by the required authorities to view and analyse the attendance over the internet from anywhere at any time.

Publish Subscribe:

MQTT was created with the goal of collecting data from many devices and then transporting that data to the IT infrastructure. It is lightweight, and therefore ideal for remote monitoring, especially in M2M connections that require a small code footprint or where network bandwidth is limited.

In the client-sever model, a client communicates directly with an endpoint. The pub/sub model decouples the client that sends a message (the publisher) from the client or clients that receive the messages (the subscribers). The publishers and subscribers never contact each other directly.

MQTT is a publish/subscribe protocol that allows edge-of-network devices to publish to a broker. Clients connect to this broker, which then mediates communication between the two devices. Each device can subscribe, or register, to particular topics. When another client publishes a message on a subscribed topic, the broker forwards the message to any client that has subscribed.

CHAPTER 2

COMPONENTS REQUIRED

2.1 HARDWARE COMPONENTS REQUIRED

- NodeMCU ESP8266
- RFID-RC522 Module
- Jumper Wires
- Micro USB Cable
- Mini Breadboard (optional)
- Arduino Uno
- RFID Tags

2.2 SOFTWARE REQUIRED

- Arduino IDE
- Adafruit IO
- Fritzing software

CHAPTER 3

IMPLEMENTATION METHODOLOGIES

3.1 HARDWARE IMPLEMENTATION

The RFID Module : - RFID stands for Radio Frequency Identification. Here digital data stored in RFID tags are captured by a reader via radio waves. It is having a reader and a tag as we see in fig 3.1. It acts as input device consists of RFID reader and RFID tags with antenna. RFID reader uses radio waves to read and capture information data stored on a tag when it gets swiped over that reader. Hence, RFID technology is applied to reduce the time involved in recurrent manual attendance.

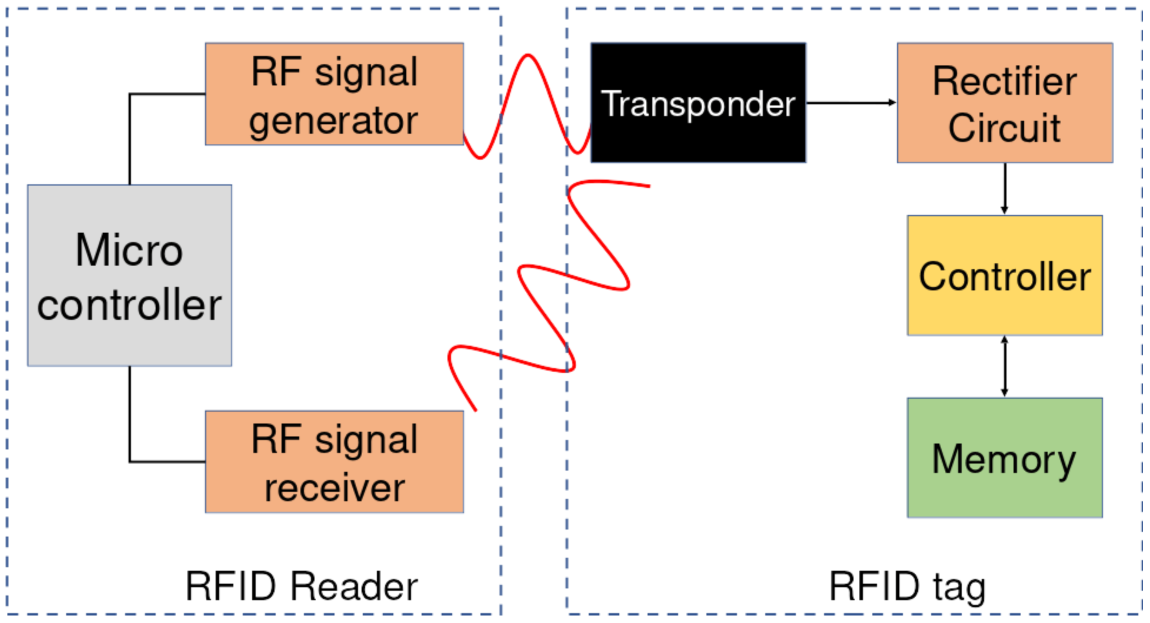


Figure 3.1 RFID module

MFRC522 reader is used with features of low cost, compact size, low power consumption, portable and installable as needed. Students must need to enter in the class, so passive tags are used, which are powered through reader’s electromagnetic fields to receive messages or ID data wirelessly from the reader. Because passive tags don’t contain battery.

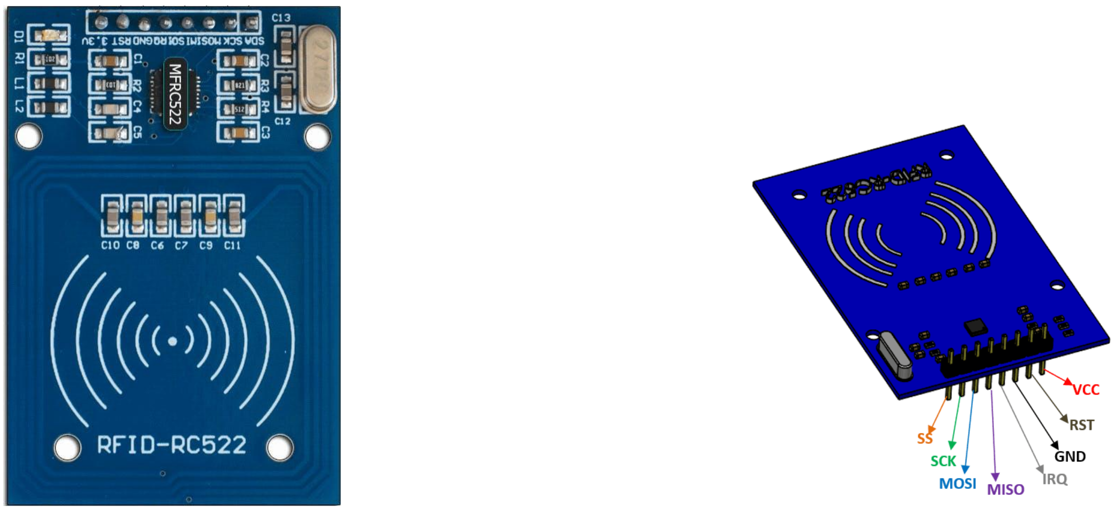


Figure 3.2   RC522 pin configuration

The RC522 is a RF Module that consists of a RFID reader fig 3.2, RFID card and a key chain. The module operates 13.56MHz which is industrial (ISM) band and hence can be used without any license problem. The module operates at 3.3V typically and hence commonly used in 3.3V designs. It is normally used in application where certain person/object has to be identified with a unique ID.

Figure 3.3   RFID Tag

The keychain shown in fig 3.3 has 1kB memory in it which can be used to stored unique data. The RC522 reader module can both read and write data into these memory elements. The reader can read data only form passive tags that operate on 13.56MHz

Specifications:

- Operating Current :13-26mA / DC 3.3V
- Idle Current :10-13mA / DC 3.3V
- Sleep Current: < 80uA
- Peak Current: < 30mA
- Operating Frequency: 13.56MHz
- Supported card types: mifare1 S50, mifare1 S70 MIFARE Ultralight, mifare Pro, MIFARE DESFire
- Environmental Operating Temperature: -20 - 80 degrees Celsius
- Environmental Storage Temperature: -40 - 85 degrees Celsius
- Relative humidity: relative humidity 5% - 95%
- Reader Distance: ≥ 50mm / 1.95" (mifare 1)
- Module Size: 40mm × 60mm
- Module interface: SPI
- Data transfer rate: Maximum 10Mbit/s

NodeMCU : - NodeMCU shown in fig 3.4 is an open source development board and firmware based in the widely used ESP8266 -12E WiFi module. It allows you to program the ESP8266 WiFi module with the simple and powerful LUA programming language or Arduino IDE.

With just a few lines of code you can establish a WiFi connection and define input/output pins according to your needs exactly like arduino, turning your ESP8266 into a web server and a lot more. It is the WiFi equivalent of ethernet module.

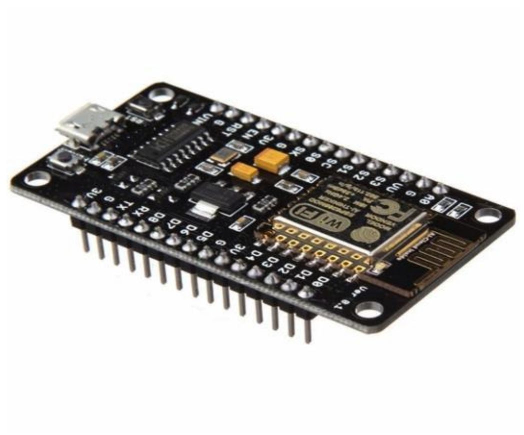


Figure 3.4 NodeMCU

Specifications:

- Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
- Operating Voltage: 3.3V
- Input Voltage: 7-12V
- Digital I/O Pins (DIO): 16
- Analog Input Pins (ADC): 1
- UARTs: 1
- SPIs: 1
- I2Cs: 1
- Flash Memory: 4 MB
- SRAM: 64 KB
- Clock Speed: 80 MHz
- USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
- PCB Antenna
- Small Sized module to fit smartly inside your IoT projects

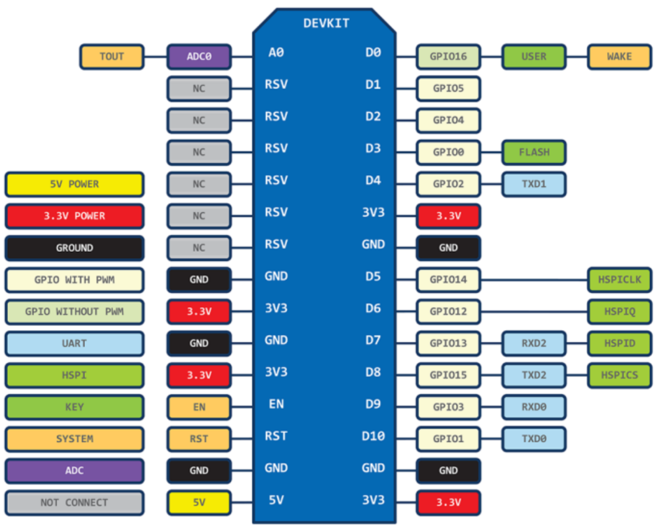


Figure 3.5   NodeMCU pin configuration

The ESP8266 chip requires 3.3V power supply voltage. It should not be powered with 5 volts like other arduino boards. We can see the whole pin configuration in fig 3.5

Arduino : - Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board. The pin diagram is in fig 3.6

Specifications:

- Microcontroller: ATmega328P
- Operating Voltage: 5V
- Input Voltage (recommended): 7-12V
- Inout Voltage (limit): 6-20V
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- PWM Digital I/O Pins: 6
- Analog Input Pins: 6
- DC Current per I/O Pin: 20 mA
- DC current for 3.3V Pin: 50 mA
- Flash Memory: 32 KB (ATmega328P) of which 0.5 KB used by bootloader
- SRAM: 2 KB (ATmega328P)
- EEPROM: 1 KB (ATmega328P)
- Clock Speed: 16 MHz
- LED_BUILTIN: 13
- Length: 68.6 mm
- Width: 58.4 mm
- Weight: 25 g

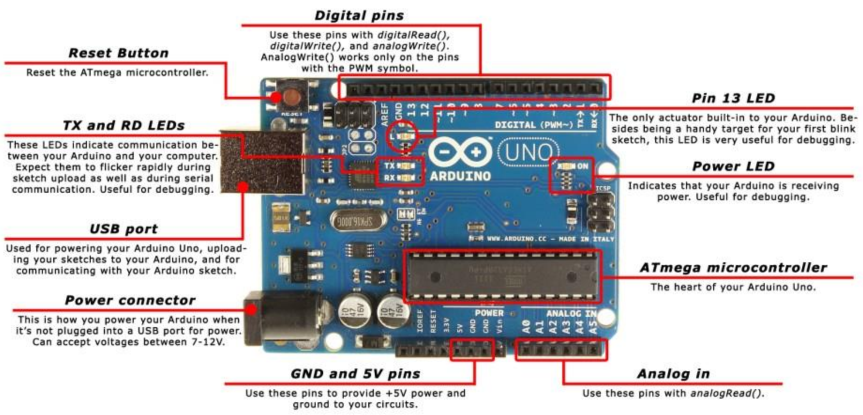
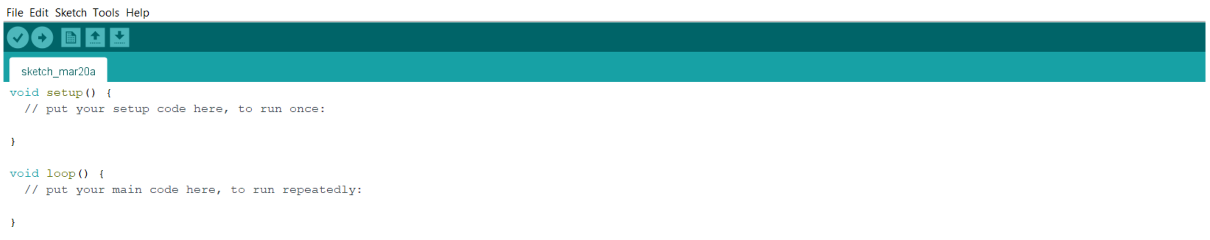

Digital pins
Use these pins with digitalRead(), digitalWrite(), and analogWrite(). AnalogWrite() works only on the pins with the PWM symbol.
Reset Button
Reset the ATmega microcontroller.
Pin 13 LED
The only actuator built-in to your Arduino. Besides being a handy target for your first blink sketch, this LED is very useful for debugging.
TX and RD LEDs
These LEDs indicate communication between your Arduino and your computer. Expect them to flicker rapidly during sketch upload as well as during serial communication. Useful for debugging.
Power LED
Indicates that your Arduino is receiving power. Useful for debugging.
USB port
Used for powering your Arduino Uno, uploading your sketches to your Arduino, and for communicating with your Arduino sketch.
ATmega microcontroller
The heart of your Arduino Uno.
Power connector
This is how you power your Arduino when it's not plugged into a USB port for power. Can accept voltages between 7-12V.
GND and 5V pins
Use these pins to provide +5V power and ground to your circuits.
Analog in
Use these pins with analogRead().

Figure 3.6 Arduino pin configuration

3.2 SOFTWARE IMPLEMENTATION

Arduino IDE : - The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards.

Use the Arduino Uno on the Arduino Desktop IDE. Install the board drivers. Open the sketch. And select the board type and port. Upload the program.



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

}

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

}
```

Figure 3.7 New Sketch

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them. The new sketch window is shown in fig 3.7.

Adafruit IO : - Adafruit.io is a cloud service - that just means we run it for you and you don't have to manage it. Adafruit.io can handle and visualize multiple feeds of data. We can view your dashboards from anywhere in the world. You need an account because we want to make sure the data you upload is available to you and only you until you are ready to make it public.

You can connect to it over the Internet. It's meant primarily for storing and then retrieving data but it can do a lot more than just that. It can

- Display your data in real-time, online
- Make your project internet-connected: Control motors, read sensor data, and more!
- Connect projects to web services like Twitter, RSS feeds, weather services, etc.
- Connect your project to other internet-enabled devices

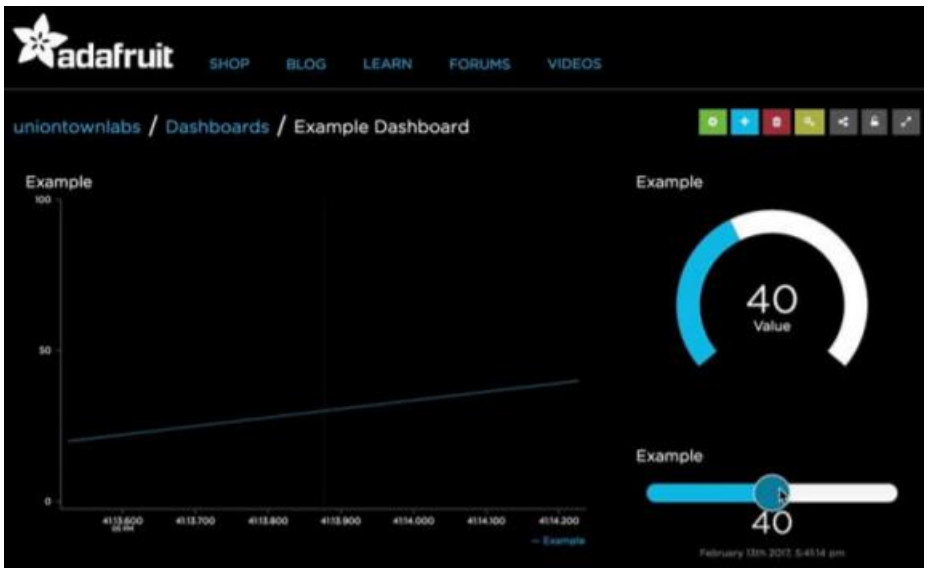
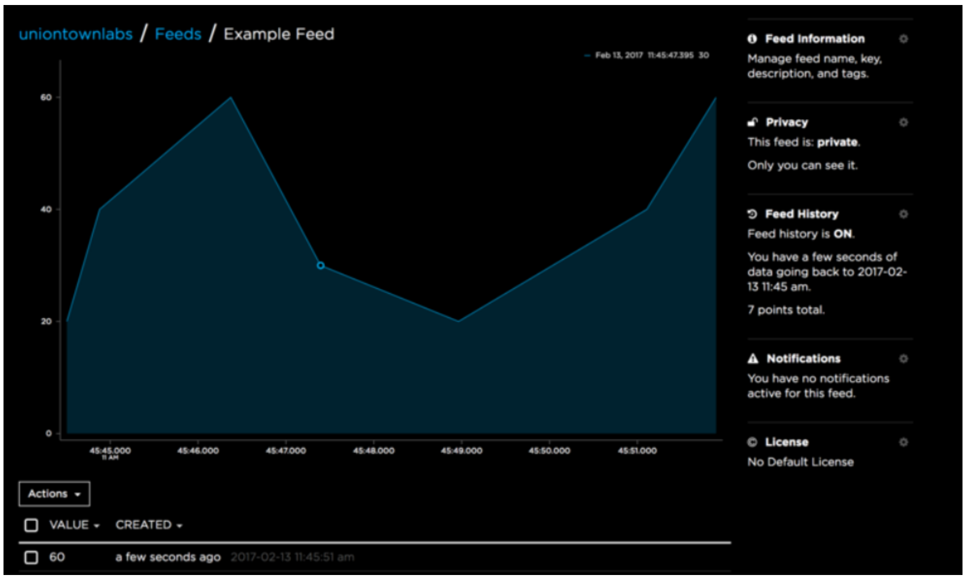


Figure 3.8 Dashboard

Dashboards shown in fig 3.8 are a feature integrated into Adafruit IO which allow you to chart, graph, gauge, log, and visualize your data. You can view your dashboards from anywhere in the world.



Feed Information
Manage feed name, key, description, and tags.

Privacy
This feed is: **private**.
Only you can see it.

Feed History
Feed history is **ON**.
You have a few seconds of data going back to 2017-02-13 11:45 am.
7 points total.

Notifications
You have no notifications active for this feed.

License
No Default License

VALUE	CREATED	
60	a few seconds ago	2017-02-13 11:45:51 am

Figure 3.9 Feed of Adafruit

Feeds shown in fig 3.9 are the core of Adafruit IO. They hold both the data you uploaded and meta-data about the data your sensors push to Adafruit IO. For example, the date and time when it was uploaded. Or, the GPS coordinates where the data came from.

MQTT, or message queue telemetry transport, is a protocol for device communication that Adafruit IO supports. Using a MQTT library or client you can publish and subscribe to a feed to send and receive feed data.

CHAPTER 4

WORKING

4.1 CIRCUIT DIAGRAM

The complete circuit diagrams shown in fig 4.1 for the RFID Smart Attendance system is shown below. The Circuit Diagram was created using Fritzing software. The RFID scanner is powered by the 3.3V pin of Arduino, the ESP is powered by the 5V pin of Arduino. And the Arduino UNO module is powered by USB port. SDA and RST pins are connected with 9 and 10 pin of Arduino as shown in circuit diagram.

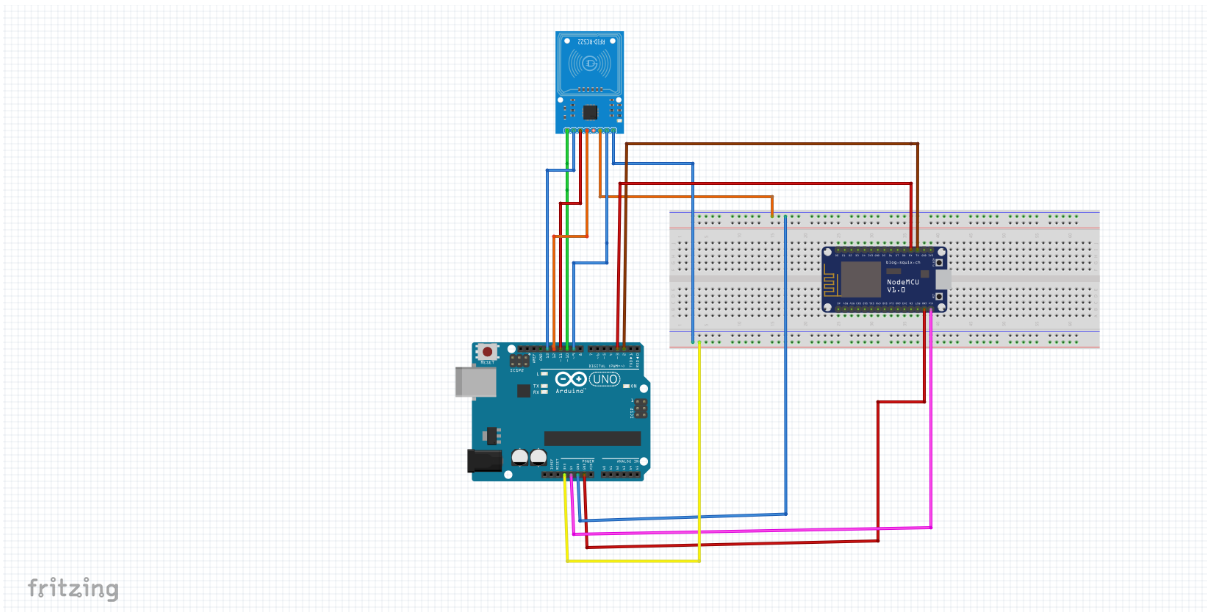


Figure 4.1 Circuit Diagram

4.2 RFID DATA READING

RFID reader (RC522), RFID tag (student's ID card) and Arduino UNO board are included in RFID data reading. To integrate the RFID tags in IoT, this publish subscribe pattern has been employed . When the RFID client wants to broadcast, send or publish data of topic ID to the broker. This needs to establish a connection with ESP8266 broker. The RFID reader acts as sensor, captures ID from tag. The RFID clients can publish the topic to the broker, after connection establishment between them. The broker then receives and publishes that available ID to destined computer client. The computer client requests to ESP8266 broker to connect, after which then broker sends the acknowledgment to the destined client for that connection request. The broker continuously sends updated current messages of topic to the subscribed computer client. We can see the hardware interfacing in fig 4.2

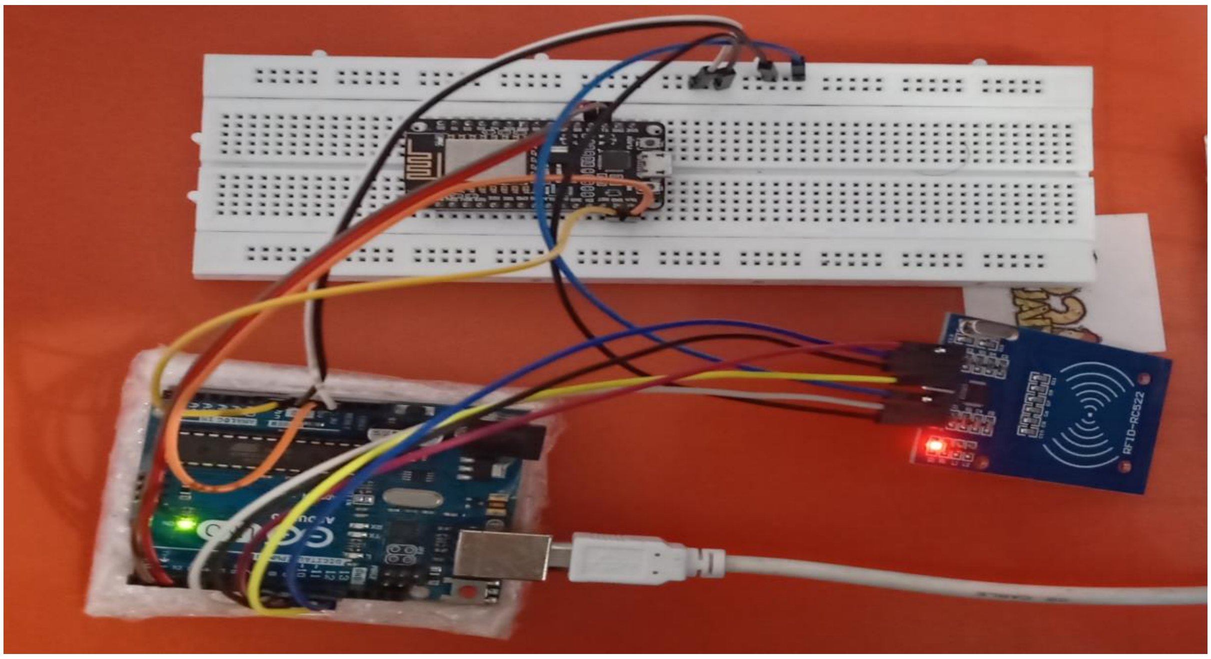


Figure 4.2 Hardware Interfacing

4.3 DATA REPORTING TO ARDUINO

The data from the RFID module is analysed in the Arduino and the correct register number according to the UID of each RFID tags is send to the Adafruit IO through NodeMCU using the MQTT protocol.

4.4 PUBLISHING TO ADAFRUIT IO

With the help of internet the NodeMCU sends the data to the cloud by using the MQTT, which is a lightweight publish-subscribe-based messaging protocol.

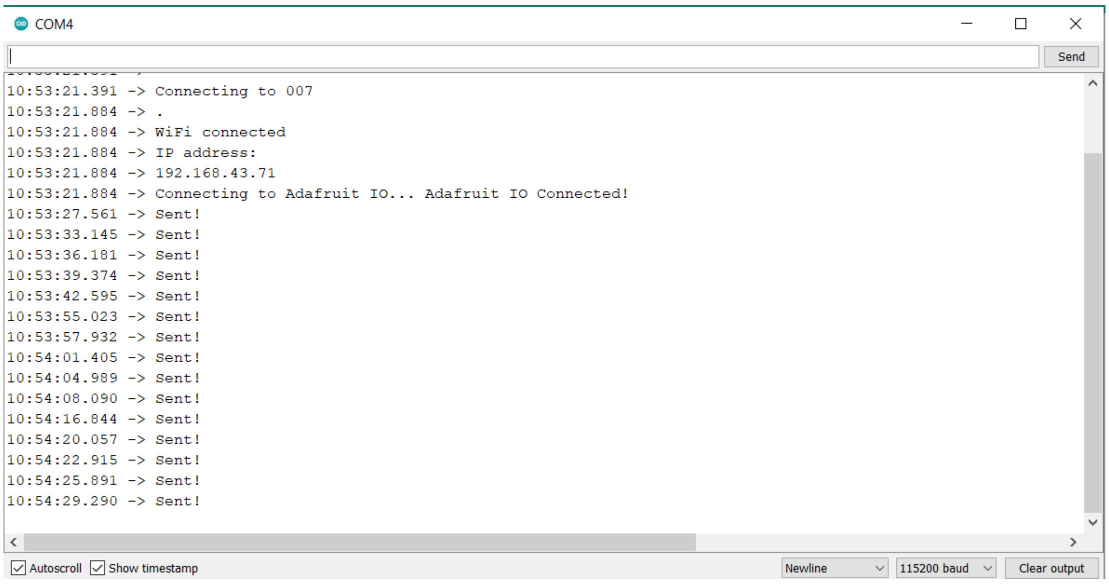
The attendance details are now visible in the dashboard of Adafruit IO.

CHAPTER 5

RESULTS

5.1 CONNECTING TO ADAFRUIT IO

The NodeMCU is connected to Adafruit IO using MQTT protocol and that is visible in the serial window output.



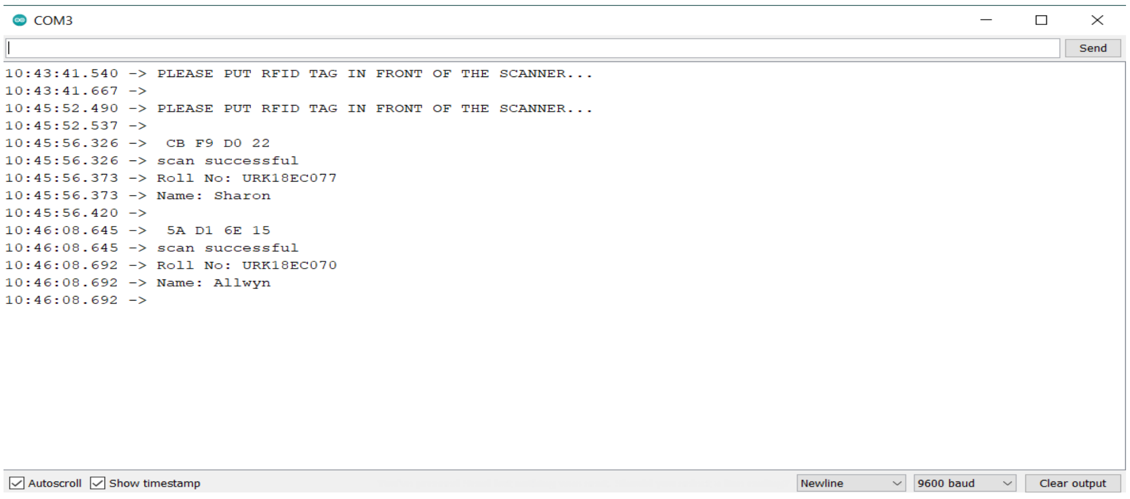
```
10:53:21.391 -> Connecting to 007
10:53:21.884 -> .
10:53:21.884 -> WiFi connected
10:53:21.884 -> IP address:
10:53:21.884 -> 192.168.43.71
10:53:21.884 -> Connecting to Adafruit IO... Adafruit IO Connected!
10:53:27.561 -> Sent!
10:53:33.145 -> Sent!
10:53:36.181 -> Sent!
10:53:39.374 -> Sent!
10:53:42.595 -> Sent!
10:53:55.023 -> Sent!
10:53:57.932 -> Sent!
10:54:01.405 -> Sent!
10:54:04.989 -> Sent!
10:54:08.090 -> Sent!
10:54:16.844 -> Sent!
10:54:20.057 -> Sent!
10:54:22.915 -> Sent!
10:54:25.891 -> Sent!
10:54:29.290 -> Sent!
```

Figure 5.1 NodeMCU Serial Window Output

In this serial window output of NodeMCU fig 5.1 , we can see the connection establishment between the NodeMCU and the Adafruit IO. Also while scanning the RFID tag, we can see the “sent!” message in the window. And the time of sending the datais also visible in the serial window output.

5.2 ARDUINO SCAN RESULT

The RFID tags are scanned using the RC522 reader and the data was detected and matched successfully with that registered ID and it is visible in the Arduino serial window output fig 5.2.

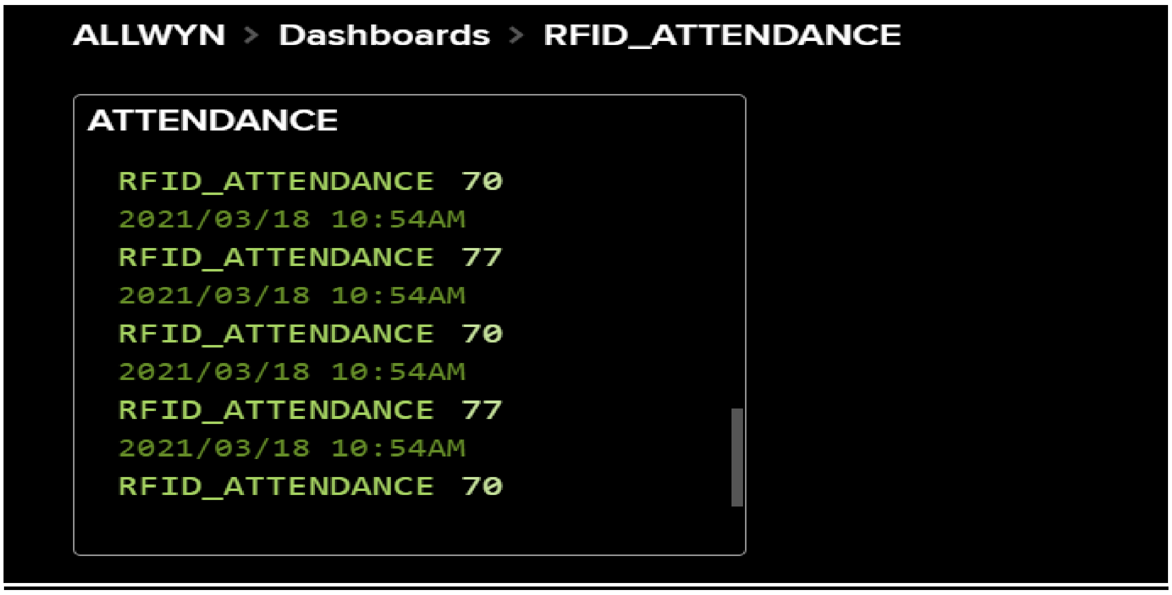


```
10:43:41.540 -> PLEASE PUT RFID TAG IN FRONT OF THE SCANNER...
10:43:41.667 ->
10:45:52.490 -> PLEASE PUT RFID TAG IN FRONT OF THE SCANNER...
10:45:52.537 ->
10:45:56.326 ->  CB F9 D0 22
10:45:56.326 -> scan successful
10:45:56.373 -> Roll No: URK18EC077
10:45:56.373 -> Name: Sharon
10:45:56.420 ->
10:46:08.645 ->  5A D1 6E 15
10:46:08.645 -> scan successful
10:46:08.692 -> Roll No: URK18EC070
10:46:08.692 -> Name: Allwyn
10:46:08.692 ->
```

Figure 5.2 Arduino Serial Window Output

The tag UID, scan result whether the card/tag is validity , the roll number and the name of the student are printed in the Arduino serial window output in real time.

5.3 DASHBOARD OUTPUT IN ADAFRUIT



ALLWYN > Dashboards > RFID_ATTENDANCE

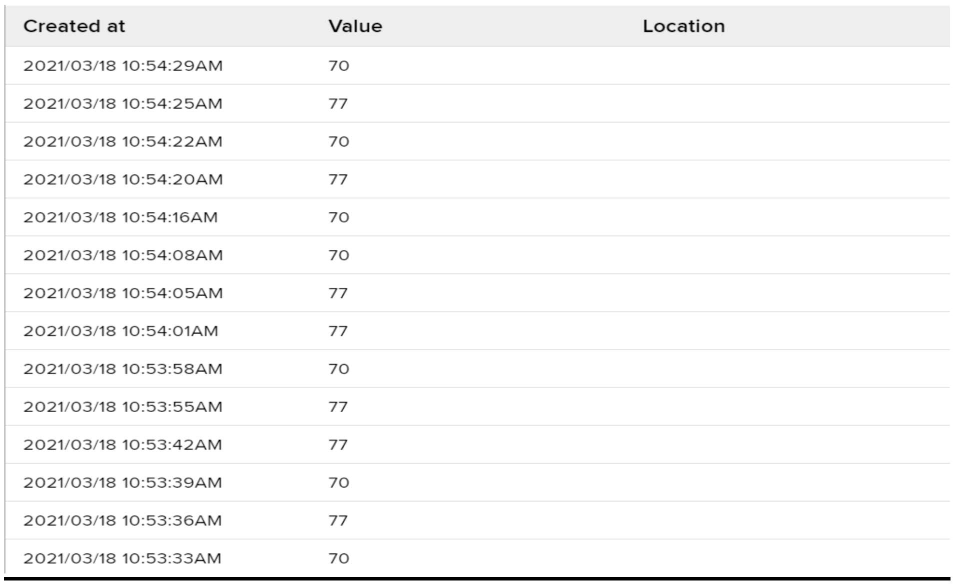
ATTENDANCE

```
RFID_ATTENDANCE 70
2021/03/18 10:54AM
RFID_ATTENDANCE 77
2021/03/18 10:54AM
RFID_ATTENDANCE 70
2021/03/18 10:54AM
RFID_ATTENDANCE 77
2021/03/18 10:54AM
RFID_ATTENDANCE 70
```

Figure 5.3 Adafruit Dashboard Output

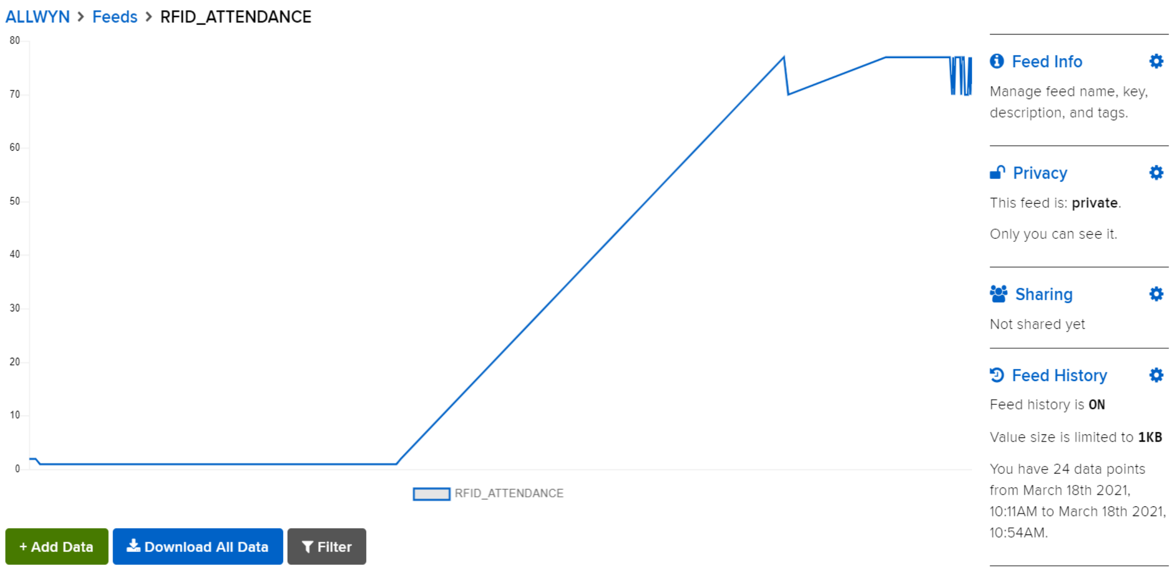
The register number is now printed in the Adafriut IO dashboard and the timing of scanning the tag is also printed in the dashboard fig 5.3.

5.4 OUTPUT FROM THE FEED



Created at	Value	Location
2021/03/18 10:54:29AM	70	
2021/03/18 10:54:25AM	77	
2021/03/18 10:54:22AM	70	
2021/03/18 10:54:20AM	77	
2021/03/18 10:54:16AM	70	
2021/03/18 10:54:08AM	70	
2021/03/18 10:54:05AM	77	
2021/03/18 10:54:01AM	77	
2021/03/18 10:53:58AM	70	
2021/03/18 10:53:55AM	77	
2021/03/18 10:53:42AM	77	
2021/03/18 10:53:39AM	70	
2021/03/18 10:53:36AM	77	
2021/03/18 10:53:33AM	70	

Figure 5.4 Feed Output



Manage feed name, key, description, and tags.

This feed is: **private**.

Only you can see it.

Not shared yet

Feed history is **ON**

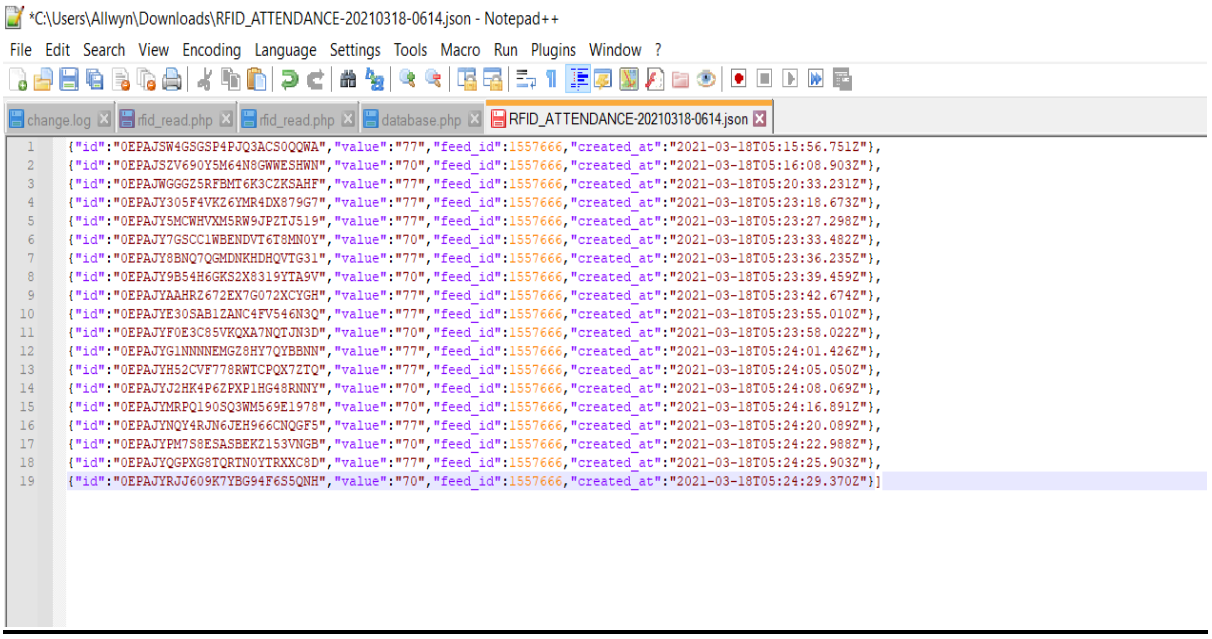
Value size is limited to **1KB**

You have 24 data points from March 18th 2021, 10:11AM to March 18th 2021, 10:54AM.

Figure 5.5 Feed Graph

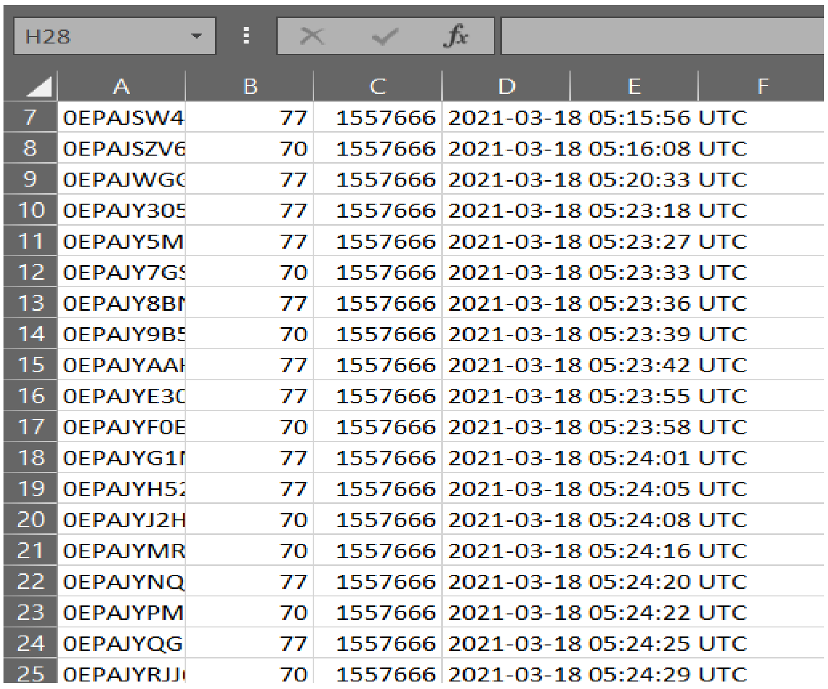
The fig 5.4 shows the feed output and fig 5.5 shows the feed graph output of the Adafruit IO.

5.5 OUTPUT AS JSON, CSV FILE


*C:\Users\Allwyn\Downloads\RFID_ATTENDANCE-20210318-0614.json - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
change.log
rfid_read.php
rfid_read.php
database.php
RFID_ATTENDANCE-20210318-0614.json

```
{"id":"0EPAJSW4GSGSP4PJQ3ACS0QQWA","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:15:56.751Z"},
{"id":"0EPAJSZV690Y5M64N8GWWESHWN","value":"70","feed_id":1557666,"created_at":"2021-03-18T05:16:08.903Z"},
{"id":"0EPAJWGGGZ5RFBMT6K3CZKSAHF","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:20:33.231Z"},
{"id":"0EPAJY305F4VKZ6YMR4DX879G7","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:23:18.673Z"},
{"id":"0EPAJY5MCWHVXM5RW9JPZTJ519","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:23:27.298Z"},
{"id":"0EPAJY7GSCC1WBENDVT6T8MN0Y","value":"70","feed_id":1557666,"created_at":"2021-03-18T05:23:33.482Z"},
{"id":"0EPAJY8BNQ7QGMDNKHDHQVTG31","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:23:36.235Z"},
{"id":"0EPAJY9B54H6GKS2X8319YTA9V","value":"70","feed_id":1557666,"created_at":"2021-03-18T05:23:39.459Z"},
{"id":"0EPAJYAAHRZ672EX7G072XCYGH","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:23:42.674Z"},
{"id":"0EPAJYE30SAB1ZANC4FV546N3Q","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:23:55.010Z"},
{"id":"0EPAJYF0E3C85VKQXA7NQTJN3D","value":"70","feed_id":1557666,"created_at":"2021-03-18T05:23:58.022Z"},
{"id":"0EPAJYG1NNNNEMGZ8HY7QYBBNN","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:24:01.426Z"},
{"id":"0EPAJYH52CVF778RWTCPQX7ZTQ","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:24:05.050Z"},
{"id":"0EPAJYJ2HK4P6ZPXP1HG48RNNY","value":"70","feed_id":1557666,"created_at":"2021-03-18T05:24:08.069Z"},
{"id":"0EPAJYMRPQ190SQ3WM569E1978","value":"70","feed_id":1557666,"created_at":"2021-03-18T05:24:16.891Z"},
{"id":"0EPAJYNQY4RJN6JEH966CNQGF5","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:24:20.089Z"},
{"id":"0EPAJYPM7S8ESASBEKZ153VNGB","value":"70","feed_id":1557666,"created_at":"2021-03-18T05:24:22.988Z"},
{"id":"0EPAJYQGPXG8TQRTN0YTRXXC8D","value":"77","feed_id":1557666,"created_at":"2021-03-18T05:24:25.903Z"},
{"id":"0EPAJYRJJ609K7YBG94F6S5QNH","value":"70","feed_id":1557666,"created_at":"2021-03-18T05:24:29.370Z"}]
```

Figure 5.6 json file



	A	B	C	D	E	F
7	0EPAJSW4	77	1557666	2021-03-18 05:15:56 UTC		
8	0EPAJSZV6	70	1557666	2021-03-18 05:16:08 UTC		
9	0EPAJWGG	77	1557666	2021-03-18 05:20:33 UTC		
10	0EPAJY305	77	1557666	2021-03-18 05:23:18 UTC		
11	0EPAJY5M	77	1557666	2021-03-18 05:23:27 UTC		
12	0EPAJY7GS	70	1557666	2021-03-18 05:23:33 UTC		
13	0EPAJY8BN	77	1557666	2021-03-18 05:23:36 UTC		
14	0EPAJY9B5	70	1557666	2021-03-18 05:23:39 UTC		
15	0EPAJYAAH	77	1557666	2021-03-18 05:23:42 UTC		
16	0EPAJYE3C	77	1557666	2021-03-18 05:23:55 UTC		
17	0EPAJYF0E	70	1557666	2021-03-18 05:23:58 UTC		
18	0EPAJYG1N	77	1557666	2021-03-18 05:24:01 UTC		
19	0EPAJYH52	77	1557666	2021-03-18 05:24:05 UTC		
20	0EPAJYJ2H	70	1557666	2021-03-18 05:24:08 UTC		
21	0EPAJYMR	70	1557666	2021-03-18 05:24:16 UTC		
22	0EPAJYNQ	77	1557666	2021-03-18 05:24:20 UTC		
23	0EPAJYPM	70	1557666	2021-03-18 05:24:22 UTC		
24	0EPAJYQG	77	1557666	2021-03-18 05:24:25 UTC		
25	0EPAJYRJJ	70	1557666	2021-03-18 05:24:29 UTC		

Figure 5.7 csv file

The json file fig 5.6 and the csv file fig 5.7 can be downloaded from the3 Adafruit feed.

CHAPTER 6

CONCLUSION

The aim was to get effective and efficient time-saving automated computerized attendance in real time with ready excel sheet to maintain attendance records in IoT trends has been done with implementing using Adafruit IO, NodeMCU, Arduino and MQTT for proper communication and interfacing. The system provides more accurate identification. Identifies candidates in seconds in quick and rapid way. I conclude that this user-friendly proposed system would prove to be easy to use and implement, cost efficient, time saving, less tedious, portable. To overcome unreliable and inaccurate manual work, this proposed system gets improved with less effort and yet generates the results with high accuracy and qualitative one. Ultimately, improves academic performance with encourage as time saves. This project proposed a system with MQTT implementation successfully accompanied with booming RFID technology successfully.

REFERENCES

[1] H. K. Nguyen, M. T. Chew, “RFID-Based Attendance Management System”, IEEE 2017, doi:10.1109/RTTR.7887874.

[2] T. S. Lim, S. C. Sim, and M. M. Mansor, “RFID based attendance system”, IEEE Symposium on Industrial Electronics and Applications, ISIEA 2009 -Proceedings, 2009, vol. 2, pp. 778–782, Oct 2009.

[3] M. B. Srinidhi and R. Roy, "A web enabled secured system for attendance monitoring and real time location tracking using Biometric and Radio Frequency Identification (RFID) technology”, International Conference on Computer Communication and Informatics, ICCCI 2015, pp. 1-5, January 2015.

[4] M. Kassim, H. Mazlan, N. Zaini, and M. K. Salleh, “Webbased student attendance system using RFID technology”, in Proceedings - 2012 IEEE Control and System Graduate Research Colloquium, ICSGRC 2012, pp. 213–218.

[5] T. Sharma, S. L. Aarthy, “An Automatic Attendance Monitoring System using RFID and IOT using Cloud”, Online International Conference on Blue Engineering and Technologies (IC-GET), IEEE, 2016

[6] Sri Madhu B.M, Kavya Kanagotagi, “IoT based Automatic Attendance Management System”, International Conference on Current Trends in Computer, Electrical, Electronics and Communication, ICCTCEEC 2017, pp. 83- 86

[7] A. Qaiser and S.A. Khan, “Automation of Time and Attendance using RFID Systems”, IEEE International Conference on Emerging Technologies, ICET 2006, pp. 60-63, Nov 2006, doi:10.1109/ICET.335928.

[8] A. Kassem, M. Hamad, Z. Chalhoub and S. El Dahdaah, “An RFID Attendance and Monitoring System for University Applications”, ICECS 2010, pp. 851-854, December 2010, doi:10.1109/ICECS.5724646.

[9] H. U Zaman, J. Hossain, T. Anika and D. Choudhury, "RFID based attendance system”, 2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT), July 2017, doi:101109/ICCCNT.8204180.

APPENDIX

ARDUINO CODE

```
#include <SPI.h>
#include <MFRC522.h>
#include "SoftwareSerial.h"
#define SS_PIN 10
#define RST_PIN 9
MFRC522 mfrc522(SS_PIN, RST_PIN);   // Create a MFRC522 object.
SoftwareSerial ser(2,3); // RX, TX
void setup()
{
  Serial.begin(9600);   // Initiate the serial communication
  ser.begin (115200);
  SPI.begin();      // Initiate the SPI bus
  mfrc522.PCD_Init();   // Initiate MFRC522
  Serial.println("PLEASE PUT RFID TAG IN FRONT OF THE SCANNER...");
  Serial.println();

}
void loop()
{
  // Look for new cards
```

```
if ( ! mfrc522.PICC_IsNewCardPresent())
{
  return;
}
// Select one of the cards
if ( ! mfrc522.PICC_ReadCardSerial())
{
  return;
}
//Show UID on serial monitor

String content= "";
byte letter;
for (byte i = 0; i < mfrc522.uid.size; i++)
{
  Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
   Serial.print(mfrc522.uid.uidByte[i], HEX);
   content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
   content.concat(String(mfrc522.uid.uidByte[i], HEX));
}
Serial.println();
Serial.print("scan successful");
Serial.print('\n');
content.toUpperCase();
if (content.substring(1) == "5A D1 6E 15" ) //change here the UID of the card
{
  Serial.println("Roll No: URK18EC070");
  Serial.println("Name: Allwyn");
  ser.write(70);
  Serial.println();
  delay(3000);
}
 if (content.substring(1) ==  "CB F9 D0 22" ) //change here the UID of the card
 { Serial.println("Roll No: URK18EC077");
  Serial.println("Name: Sharon");
  ser.write(77);
  Serial.println();
  delay(3000);
}
 if (content.substring(1) == "4E C6 2B 2B" ) //change here the UID of the card
{
  Serial.println("3-Yashika");
  ser.write(3);
  Serial.println();
  delay(3000);
}
}
```

NODEMCU CODE

```
#include <ESP8266WiFi.h>

#include "Adafruit_MQTT.h"

#include "Adafruit_MQTT_Client.h"

// WiFi parameters

#define WLAN_SSID       "007"

#define WLAN_PASS       "012345678"


// Adafruit IO

#define AIO_SERVER      "io.adafruit.com"

#define AIO_SERVERPORT  1883

#define AIO_USERNAME    "ALLWYN"

#define AIO_KEY         "aio_KaYX84SI2xpzCVTpmpwlWETl5t0m"


WiFiClient client;


// Setup the MQTT client class by passing in the WiFi client and MQTT server and
login details.

Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT,
AIO_USERNAME, AIO_KEY);


Adafruit_MQTT_Publish Attendance = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/RFID_ATTENDANCE");


char ID;
```

```
void setup() {
  Serial.begin(115200);
  Serial.println(F("Adafruit IO Example"));

  // Connect to WiFi access point.
  Serial.println(); Serial.println();
  delay(10);
  Serial.print(F("Connecting to "));
  Serial.println(WLAN_SSID);

  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(F("."));
  }
  Serial.println();

  Serial.println(F("WiFi connected"));
  Serial.println(F("IP address: "));
  Serial.println(WiFi.localIP());

  // connect to adafruit io
  connect();

}
```

```
// connect to adafruit io via MQTT
void connect() {
  Serial.print(F("Connecting to Adafruit IO... "));
  int8_t ret;
  while ((ret = mqtt.connect()) != 0) {
    switch (ret) {
      case 1: Serial.println(F("Wrong protocol")); break;
      case 2: Serial.println(F("ID rejected")); break;
      case 3: Serial.println(F("Server unavail")); break;
      case 4: Serial.println(F("Bad user/pass")); break;
      case 5: Serial.println(F("Not authed")); break;
      case 6: Serial.println(F("Failed to subscribe")); break;
      default: Serial.println(F("Connection failed")); break;
    }

    if(ret >= 0)
      mqtt.disconnect();

    Serial.println(F("Retrying connection..."));
    delay(5000);
  }
  Serial.println(F("Adafruit IO Connected!"));
}

void loop() {
  // ping adafruit io a few times to make sure we remain connected
  if(! mqtt.ping(3)) {
    // reconnect to adafruit io
    if(! mqtt.connected())
      connect();
  }
   if ( Serial.available() ) { // Update and send only after 1 seconds
   char a = Serial.read();
   ID = a;

   if (! Attendance.publish(ID)) {                    //Publish to Adafruit
      Serial.println(F("Failed"));
    } else {
      Serial.println(F("Sent!"));
    }
}
}
```