CS3242 - Micro-controllers and Applications

Environment Monitoring IoT device

S.K Ekanayake

170158F

Content

1. Scope of the project ……………………………………………………………. 3
2. Special features…………………………………………………………………… 3
3. High level design diagram ………………………………………………………. 4
4. List of components and their costs ……………………………………………….. 5
5. Schematic diagram ……………………………………………………………. 6
6. Fault recovery option implemented …………………………………………….. 7
7. Project circuit photo ……………………………………………………………. 7
8. How the device work
9. Join to any Wi-Fi through configuration setup …………………………… 9
10. Sensor data real-time upload to the google excel sheet in every 15min…. 10
11. Sending sensor data to local web server using CAP protocol. …………… 11
12. SD card saved data ……………………………………………………….. 11
13. Pseudo code algorithm ……………………………………………………………. 12
14. Used libraries for the project ……………………………………………………… 13
15. Full source code
16. Environment\_Monitoring\_System.ino code ……………………………. 14
17. PageIndex.h code ………………………………………………………… 21
18. Google Sheet Script code ……………………………………………….. 23
19. References……………………………………………………………………….. 24

Scope of the project

The Environment Monitoring IoT device is capable of measuring the following parameters.

1. Temperature of the environment (in 0C)
2. Relative humidity of the environment (in %)
3. Rainfall of the environment (in %)
4. Air pressure (in Pa)
5. Amplitude (in m)

The device collects data every 2 seconds via the sensors. Then the data real-time sending to the google excel sheet (which can access remotely) every 15 minutes when devices is ON and connected to the Wi-Fi. These data are formatted to have the CAP format and sent to the web server which is locally connected through the Node MCU network. The data gets updated real time in the local web server and can be monitored via an interface.

Special features

1. Power saving – Buck converter ensure that power saving. Which is done by Buck converters are highly efficient (often higher than 90%), making them useful for tasks such as converting a computer's main supply voltage (often 12 V) down to lower voltages needed by USB.
2. Different types of power inputs
3. Using DC power socket – variable voltage input can be given (in range of 7V to 35V)
4. Micro USB port – USB power input(5V)
5. Inbuild storage memory – using SD card to store data as backup of sensor read data.
6. Real time clock – To make sure which data is related which date and time.
7. Can connect to any Wi-Fi router – using network configuration UI

High level design diagram

SD Card Module

DHT11 Temperature and humidity sensor

ESP8266 V

*ESP8266 NodeMCU*

Local Web server

(locally access)

Microcontroller

Raindrop Detection Sensor

Wi-Fi module

Wi-Fi configuration end-point

BMP180 Pressure Sensor

Our home/workplace Router

Google Excel Sheet

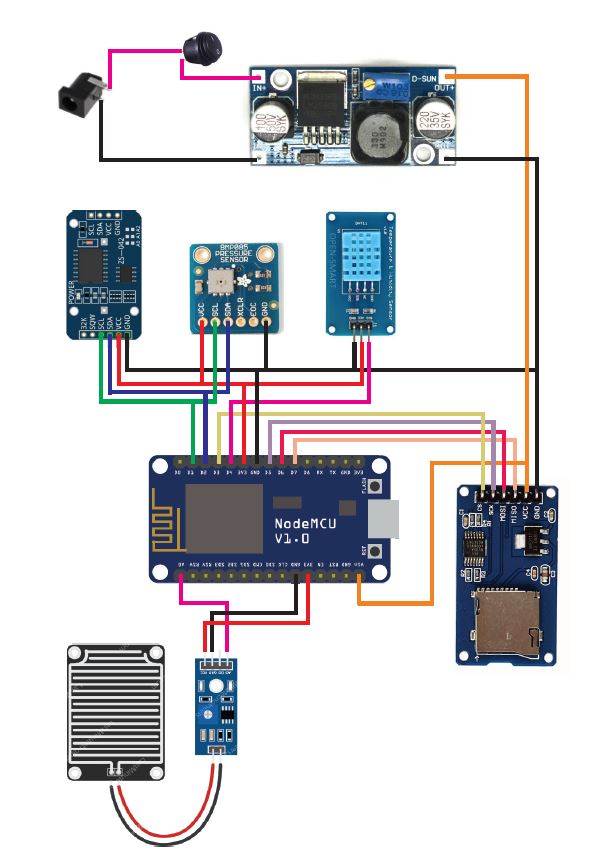
(Remotely access)

List of components and their cost

*Components*  *price (Rs.)*

1. ESP8266 (NodeMCU) 650
2. DHT11 200
3. BMP180 330
4. DS1307 clock RTC module 180
5. Micro SD card module 160
6. Raindrop Detection sensor 190
7. DC-DC Buck Converter 250
8. CR2032 Watch Clock battery 150
9. DC-005 DC power socket 40
10. Switch 20
11. Voltage regulator 20
12. Jumping Wires 200
13. Dot board 60

Schematic Diagram

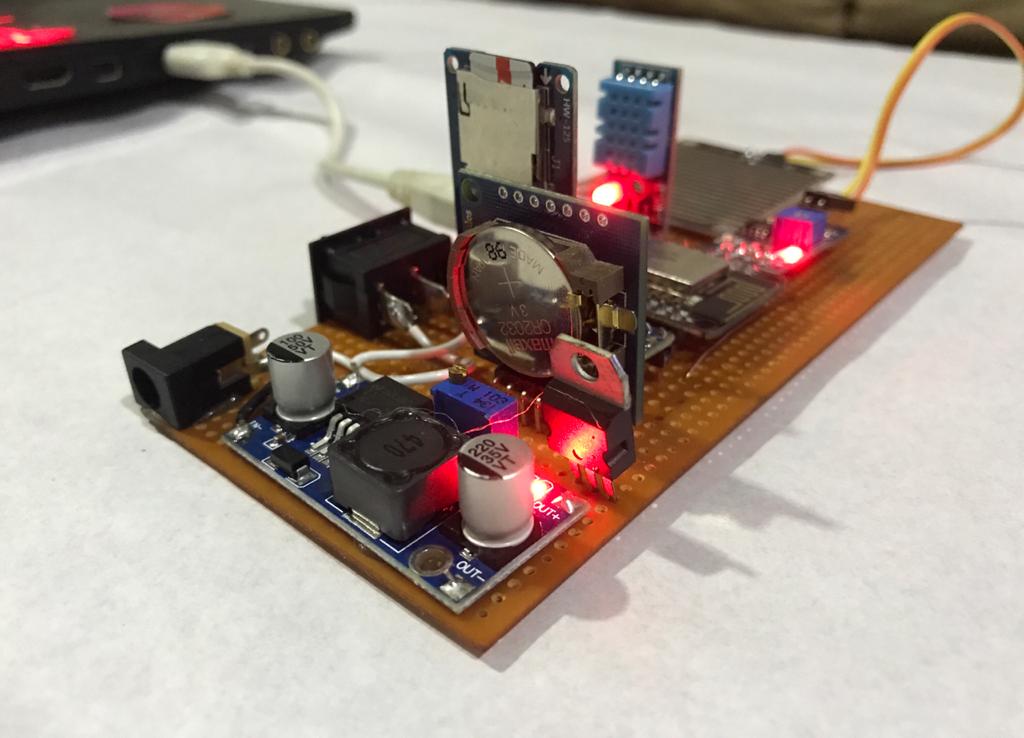


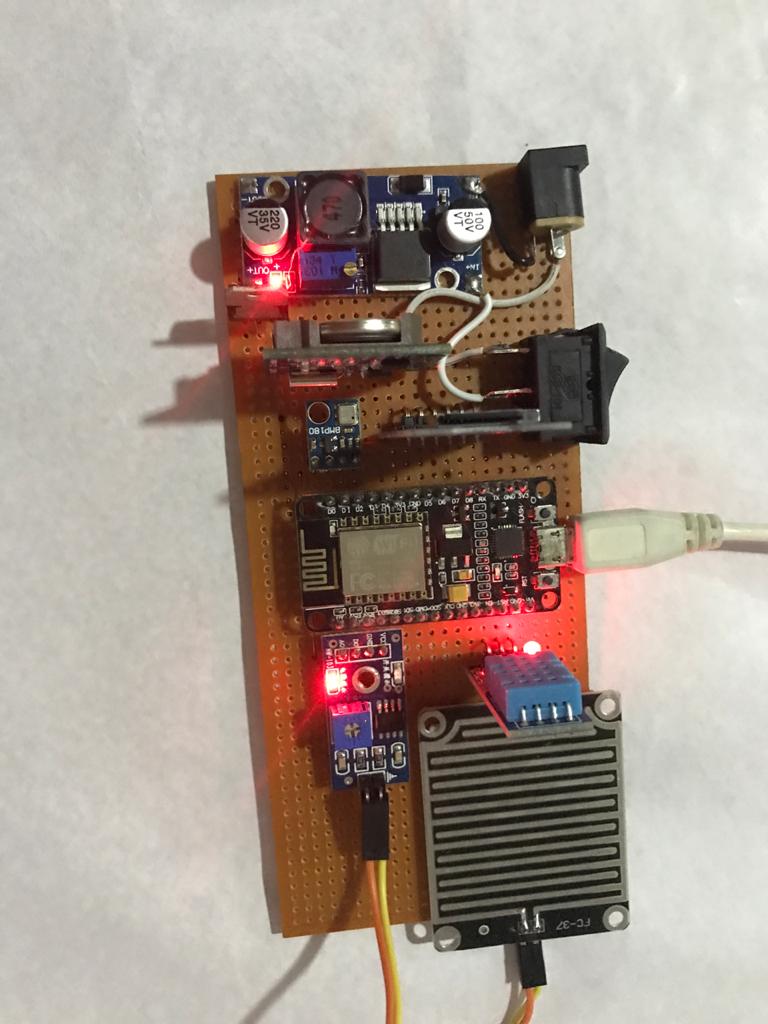
Fault recovery option implemented

1. Save the sensor read data into a SD card
2. Make temporary array and sending data when the connection stablishes again

When every 15 minutes data is sending to the google excel sheet. When data sending to excel sheet, I used SD card module to store all data in a micro SD card. So, if connection is lost but the data is safely kept in the SD card. In addition, in coding I implemented arrays which are collect sensor data only if there is no Wi-Fi connection. When the internet connection stablishes again all the data in the temp arrays are send again to the excel sheet. For update the sheet with real sensor read time, I used DS1307 clock RTC module. That will help to recovery the data in fault with correct sensor read time.

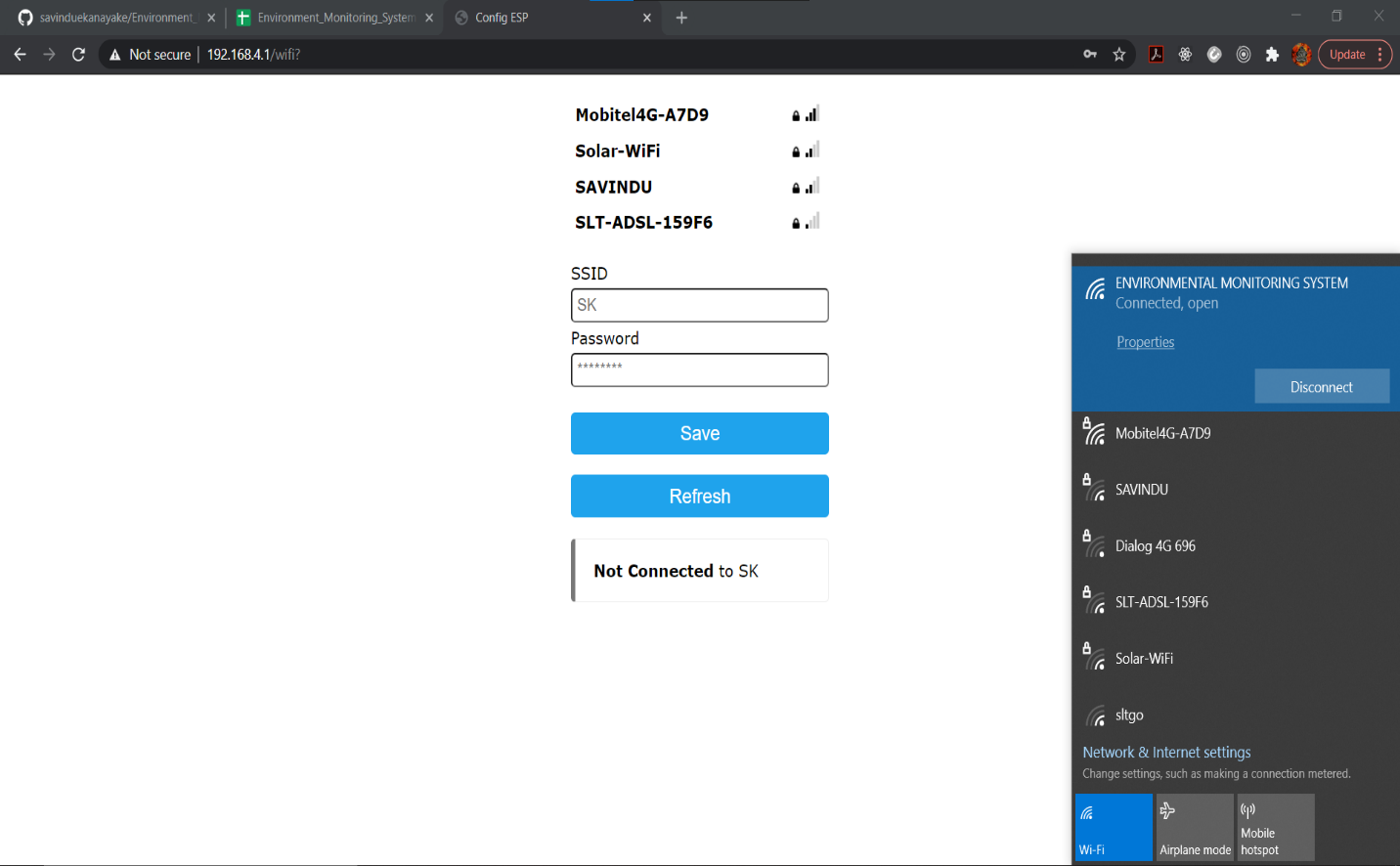
Project circuit photo





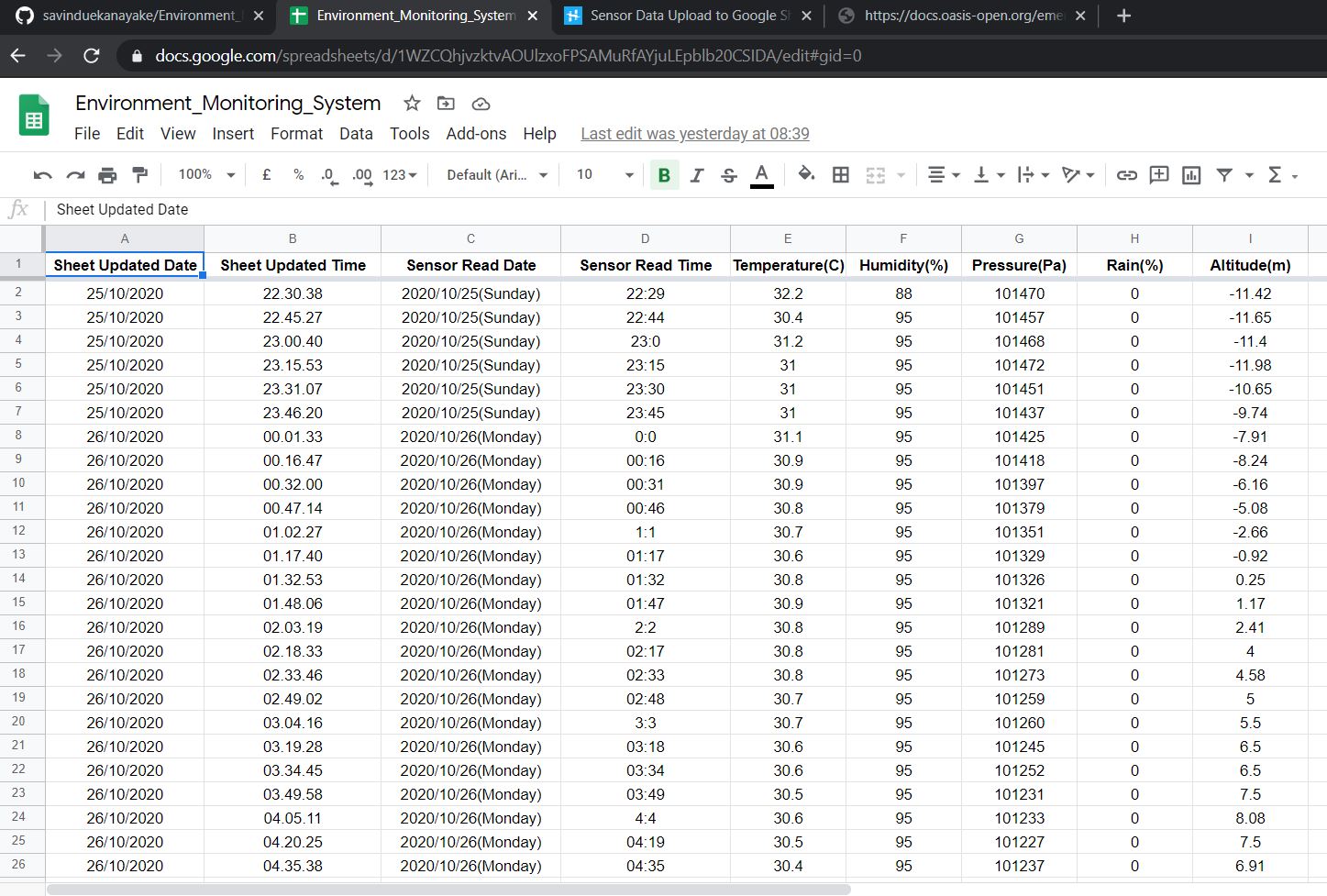
How the Device work

1. Join to any Wi-Fi through configuration setup



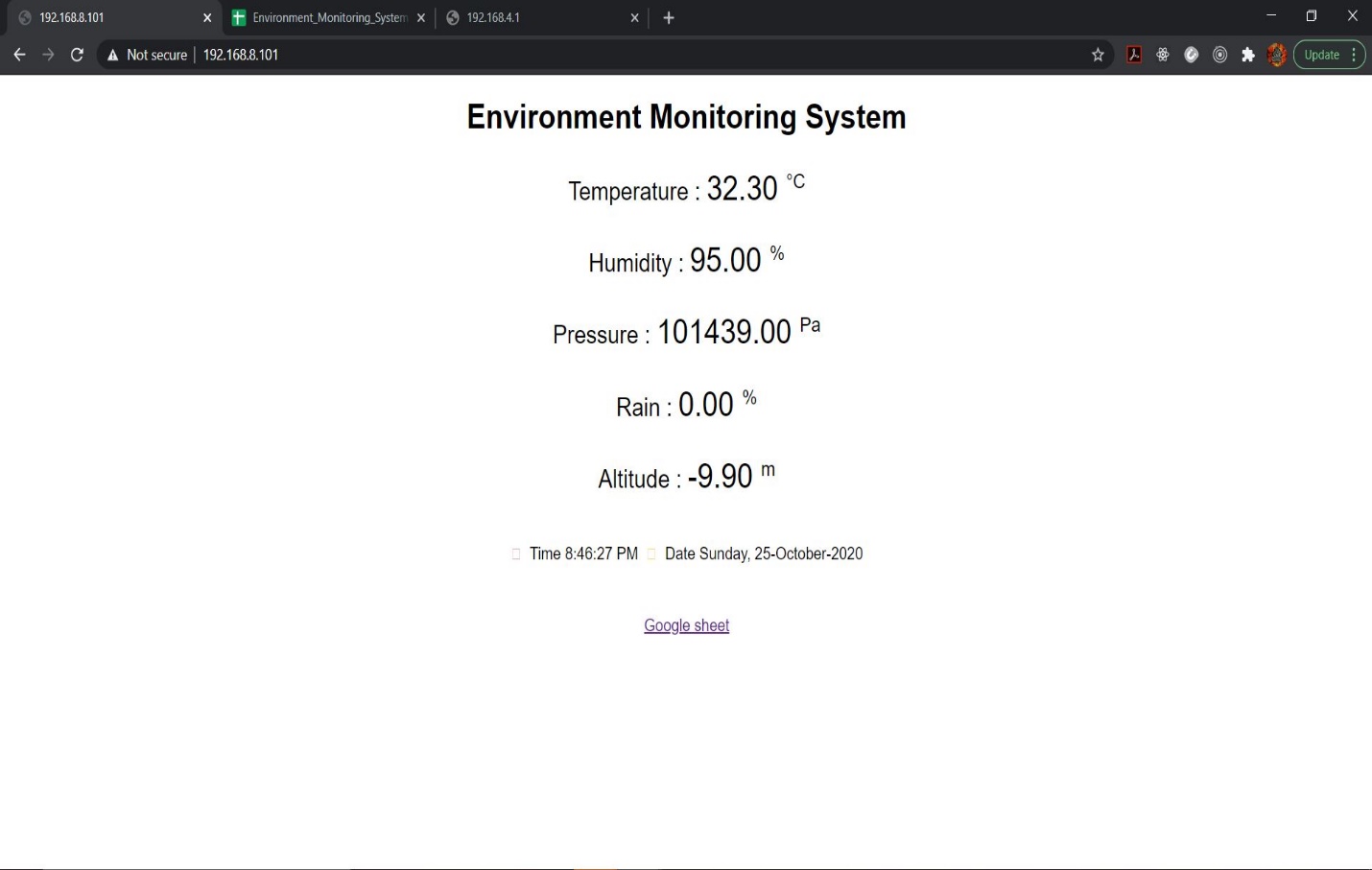
* First switch ON the device.
* Search the available Wi-Fi connections using Laptop.
* Then connect to the Wi-Fi ‘ENVIRONMENT\_MONITORING\_SYSTEM’.
* Then the configuration setup will auto popup in the laptop. (if not, we can go to configuration setup using IP- 192.168.4.1)
* Then connect to your Wi-Fi using entering correct credentials to the configuration interface.
* Now you are connected to the internet through the router.

1. Sensor data real-time upload to the google excel sheet in every 15min



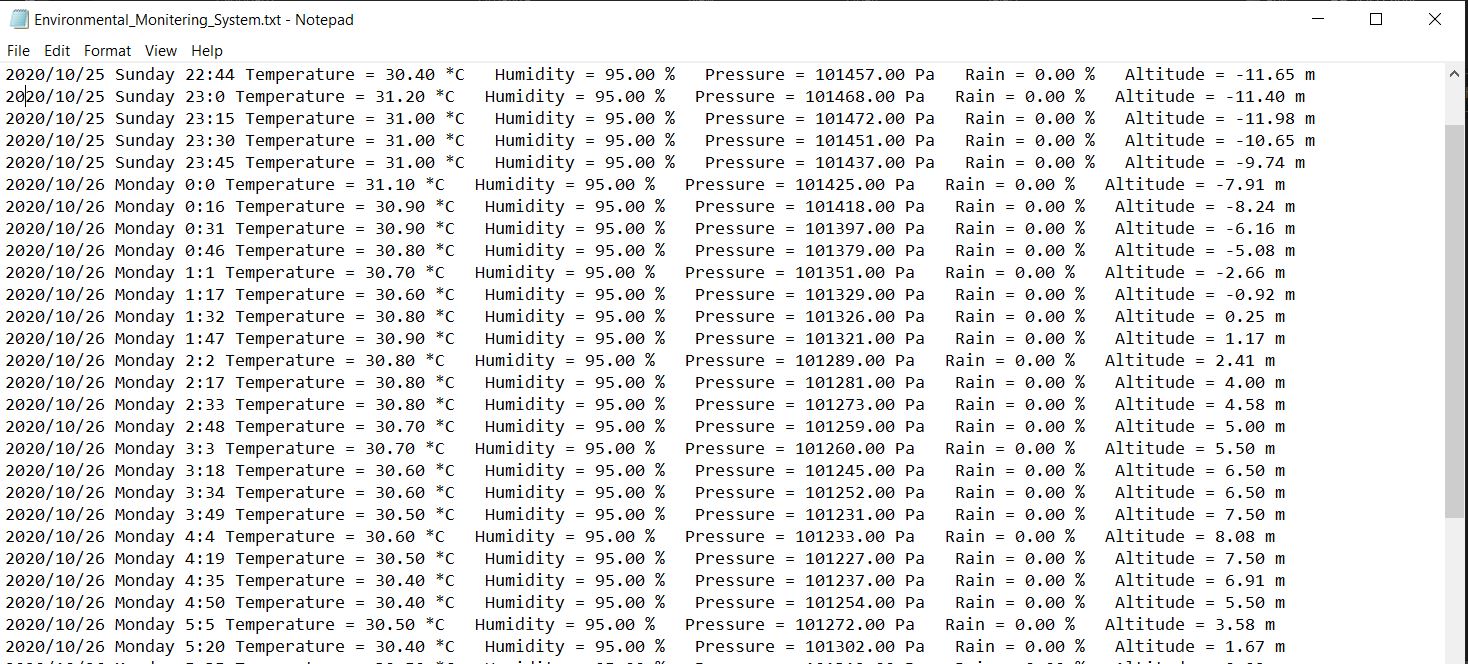
* Then IoT device is sending senor data to google excel sheet in every 15min.
* This Google Excel Sheet can be remotely access by anywhere using the link.
* Link - <https://docs.google.com/spreadsheets/d/1WZCQhjvzktvAOUlzxoFPSAMuRfAYjuLEpblb20CSIDA/edit#gid=0>

1. Sending sensor data to local web server using CAP protocol.



* Latest values of sensor data are display in a locally available web server
* For that need to connect to the Node MCU Wi-Fi using IP 192.168.8.1(dynamic IP)
* So, entering Ip 192.168.4.1 also redirected to this web page.
* For this NodeMCU sending data using Common Alerting Protocol (CAP) to this locally available server.
* Then the server read the data from CAP body and display in this page.

1. Sensor data also securely save in the SD card as a backup



Algorithm used for device and server (Pseudo code)

## **Device**

function setup():

connect to NodeMCU Wi-Fi module

configuration your home Wi-Fi using credintails

initialize DHT11, Rainfall and BMP180 sensors

function loop():

time1 =millis()

time2 = millis()

while( (time2-time1) < 900000): // this for loop collects data for 15 minutes.

time2 = millis();

get sensor readings and update the 5 lists

delay 2000ms

calculate average of each parameter and save them to 5 different variables

calculate standard deviation of each parameter

sending calculate data to Google Excel Sheet

format the sensor read data to CAP format

send CAP formatted data to Local Web Server

save the sensor collected data into SD card

## **Server backend**

function connection():

connect to local Wi-Fi

function getData():

try:

fetch all data in the Parameters

render fetched data to the Page

except any error:

print the error

## **Google excel sheet**

function doGet(e):

if(e.parameter == ‘undifined’):

print(‘No Params’)

else:

rawData = [];

get the timestamp of current time

for( param in e.parameter): // fetch all data in the Parameters

value <= stripQuotes(e.parameter[param])

switch case:

rawData[column] <= added that value

// value added to the right column using switch case

write the values to the new row in Excel Sheet

function stripQuotes(value):

value.replace(/^[“’]|[‘”]$/g, “”)

Used libraries for the project

* ESP8266WiFi.h – Connect with router
* ESP8266WebServer.h – Create a web server
* DNSServer.h – Create a domain name system
* WifiManager.h – Enter SSID & password of Wi-Fi without hard cording a credentials
* WiFiClientSecure.h – Keep connection
* Dht11.h – For DHT11 module library
* SPI.h – Start SPI communication with SD card
* SD.h – SD card library
* RTClib.h – Real time clock library
* Wire.h – To make i2c communication
* Adafruit\_BMP085.h – BMP085 barometric sensor library

Full Source code

* GitHub link for full source code and libraries–

<https://github.com/savinduekanayake/Environment_Monitoring_System>

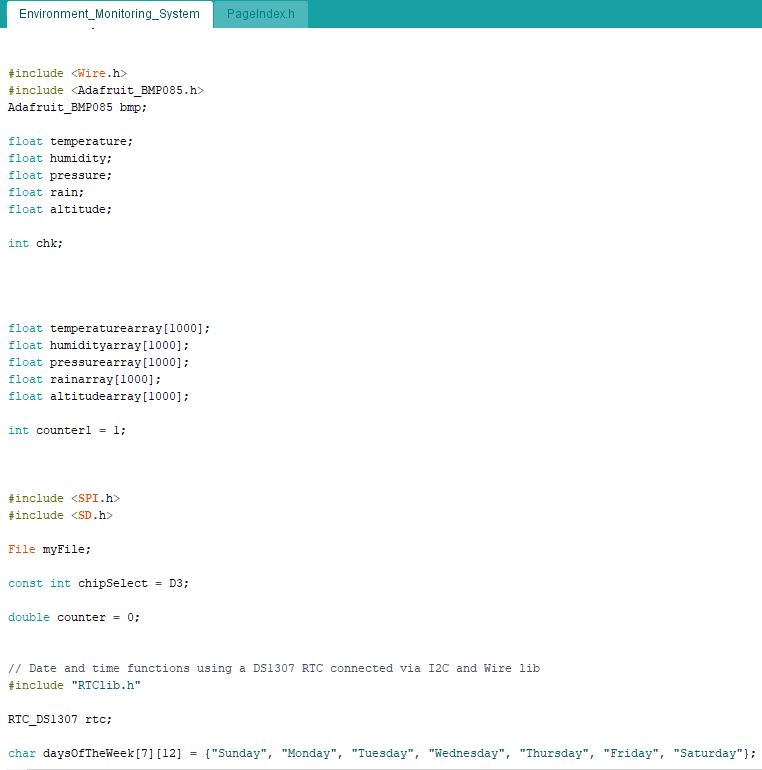
* Google sheet which upload sensor data in 15min in real-time –

<https://docs.google.com/spreadsheets/d/1WZCQhjvzktvAOUlzxoFPSAMuRfAYjuLEpblb20CSIDA/edit#gid=0>

* Node MCU updated code

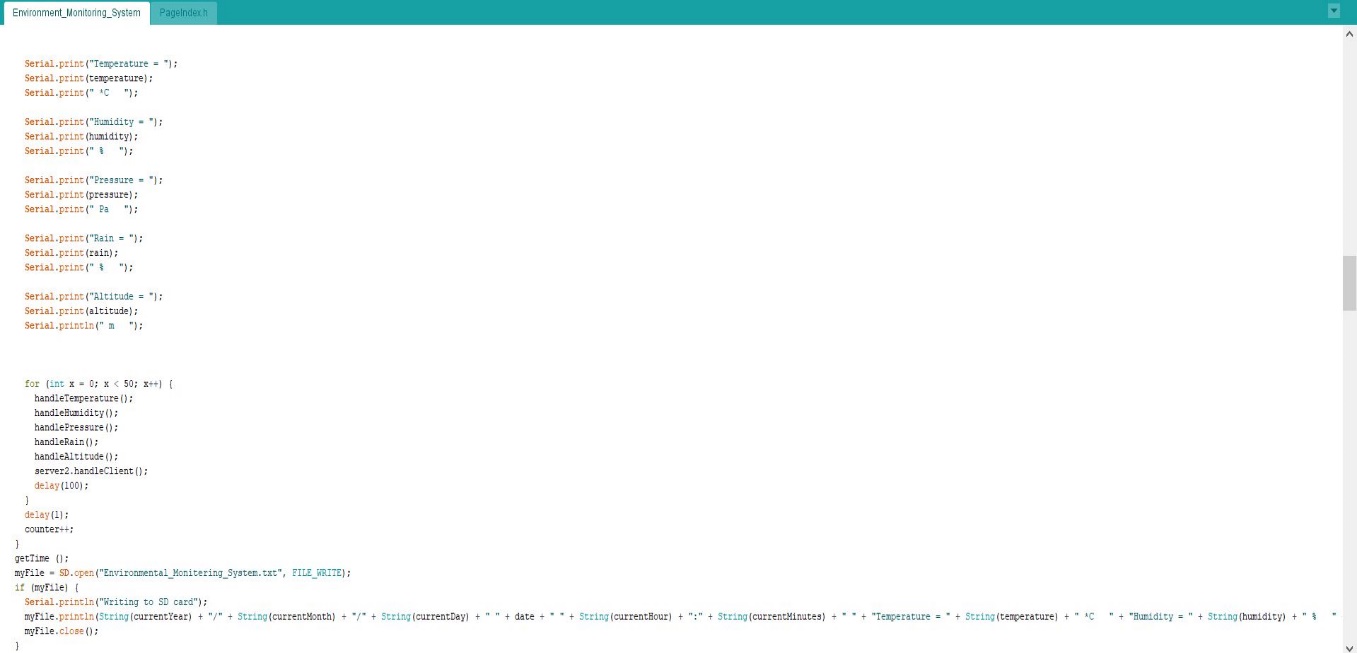
1. Environment\_Monitoring\_System.ino

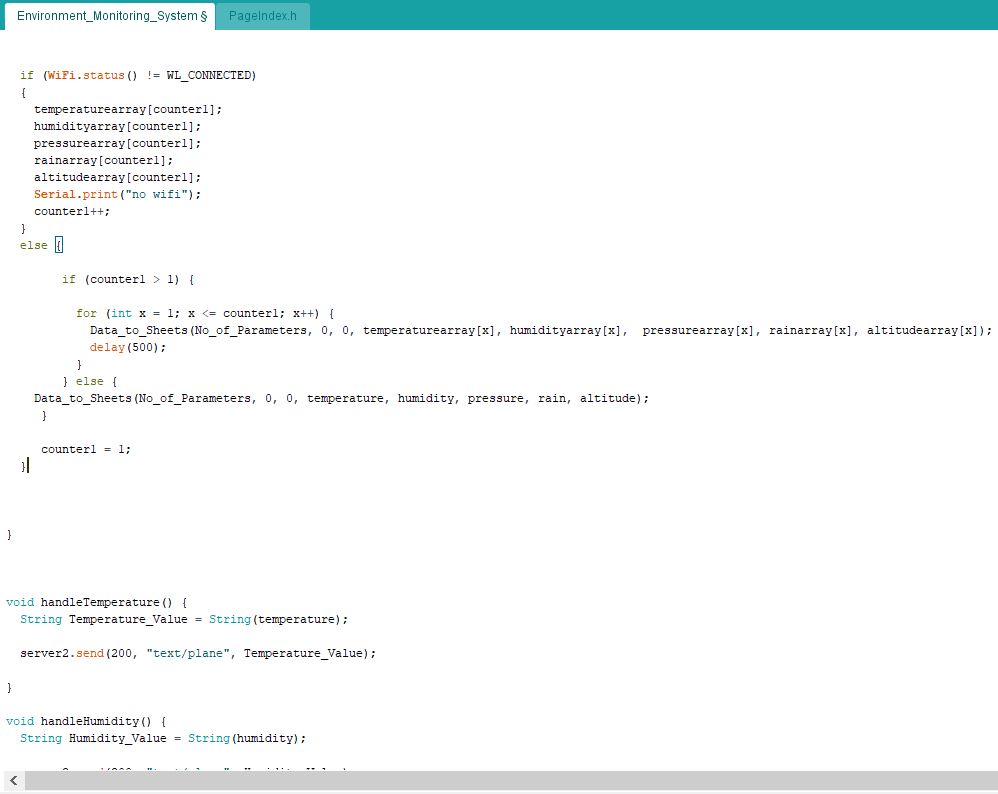






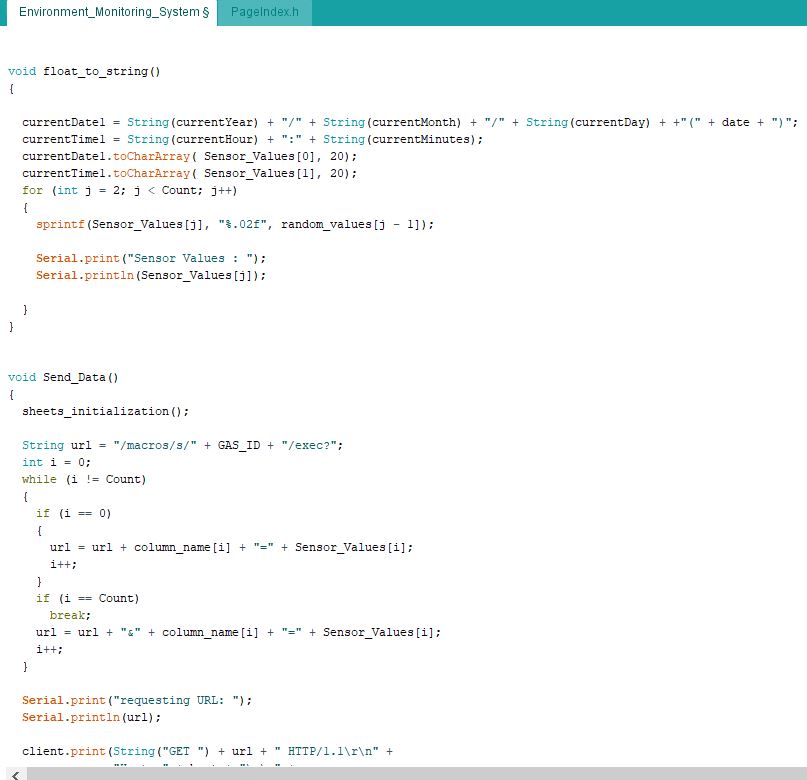






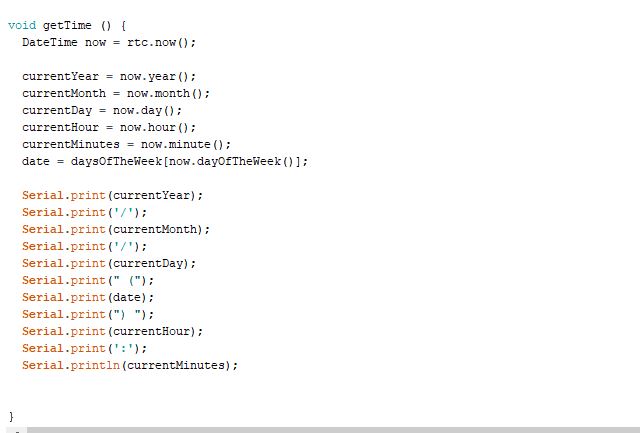






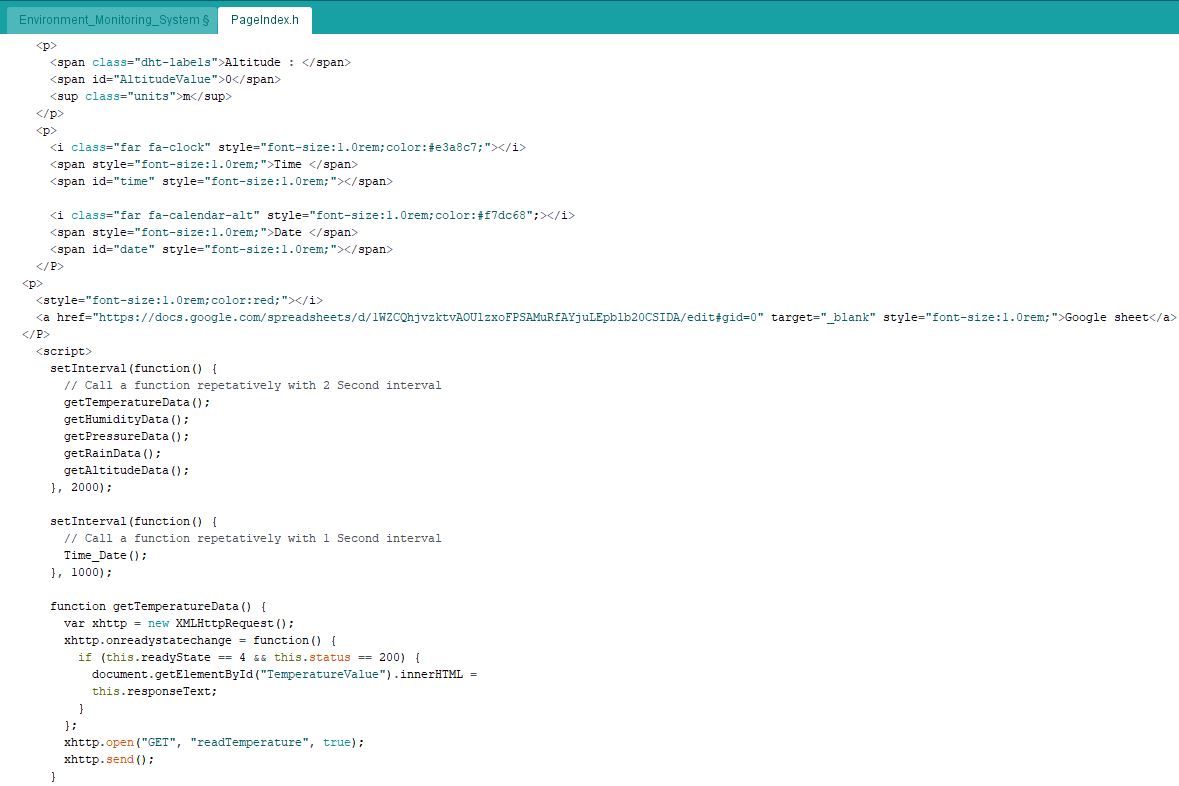




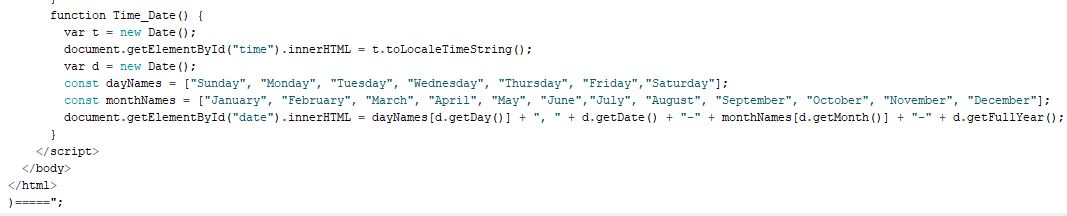


1. PageIndex.h

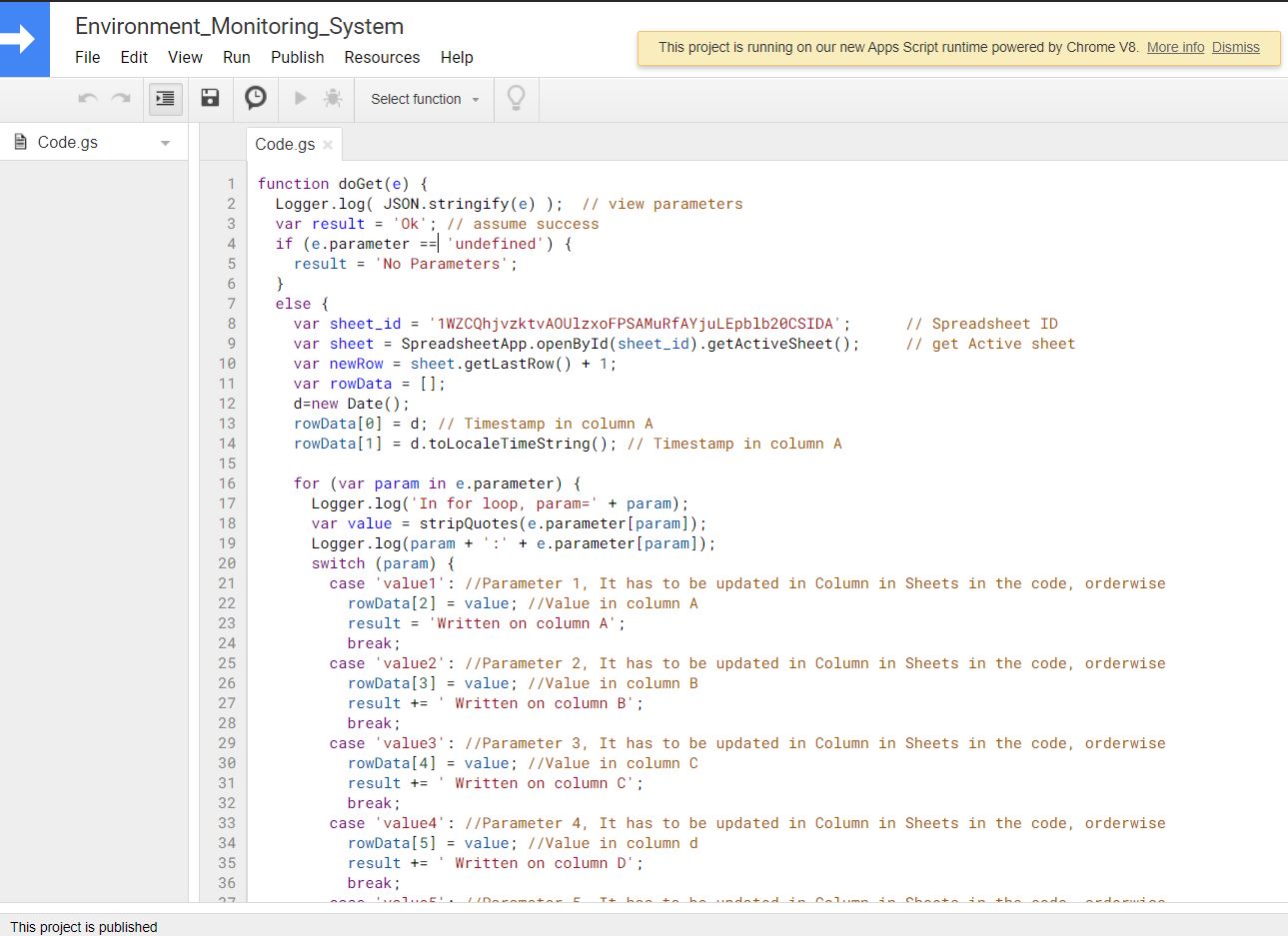


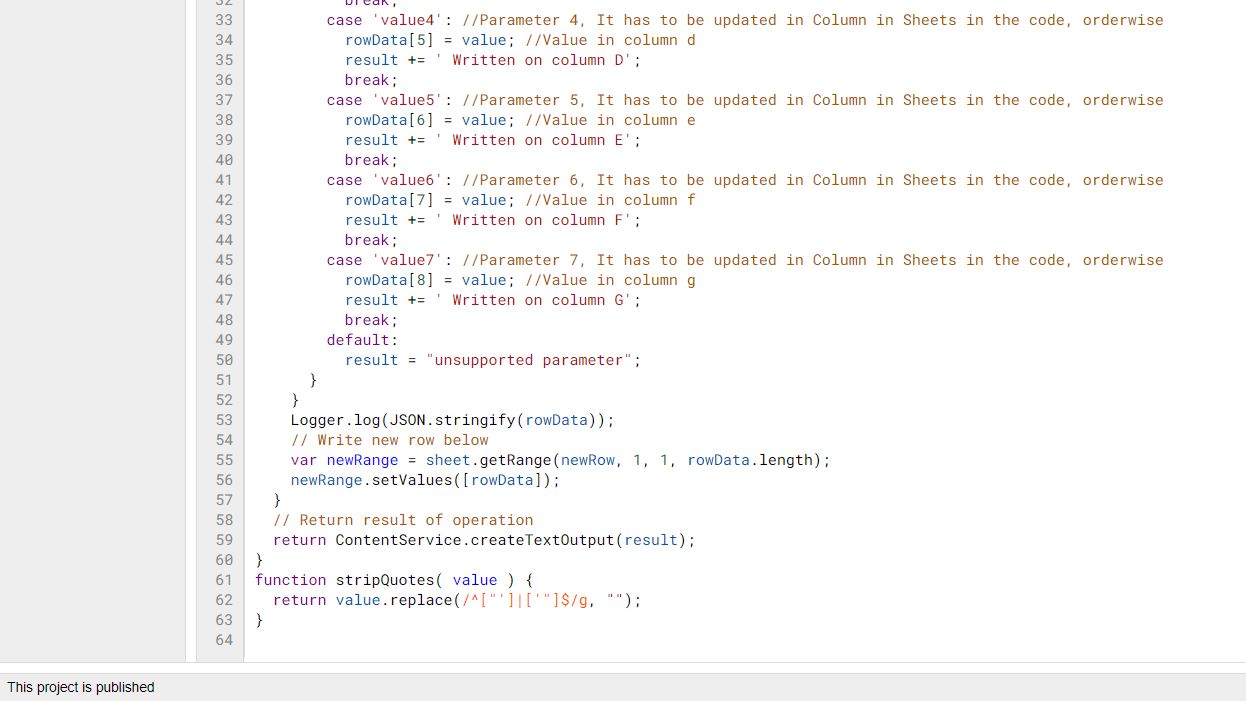






* Google Sheet Script





References

* CAP Scheme – <https://docs.oasis-open.org/emergency/cap/v1.2/CAP-v1.2.xsd>
* make a local server from NodeMCU <https://www.youtube.com/watch?v=cx4S_rcMrZc&t=571s>
* Google sheet update with NodeMCU - <https://www.hackster.io/thatiotguy/sensor-data-upload-to-google-sheets-through-nodemcu-632358>