GOOGLE SHEET SCRIPT

var Sheet = SpreadsheetApp.openById('1eKGB\_QcyqeMzGoEqtUHzdYabA1jLwElsu0pHDtrE9cg');

var timezone = "asia/kolkata"

var hours = +6

var str = "";

function doPost(e) {

  var parsedData;

  var result = {};

  try {

    parsedData = JSON.parse(e.postData.contents);

  }

  catch(f){

    return ContentService.createTextOutput("Error in parsing request body: " + f.message);

  }

     if (parsedData !== undefined){

    var flag = parsedData.format;

    if (flag === undefined){

      flag = 0;

    }

        var sheet = Sheet.getSheetByName(parsedData.sheet\_name);

    var dataArr = parsedData.values.split(",");

    var Current\_Date = new Date(new Date().setHours(new Date().getHours() + hours));

    var Current\_Time = Utilities.formatDate(Current\_Date, timezone, 'HH:mm:ss');

    var value0 = dataArr [0];

    var value1 = dataArr [1];

    var value2 = dataArr [2];

    var value3 = dataArr [3];

    var value4 = dataArr [4]; s

    switch (parsedData.command) {

      case "insert\_row":

         sheet.insertRows(2);

         sheet.getRange('A2').setValue(Current\_Date);

         sheet.getRange('B2').setValue(Current\_Time);

         sheet.getRange('C2').setValue(value0);

         sheet.getRange('D2').setValue(value1);

         sheet.getRange('E2').setValue(value2);

         sheet.getRange('F2').setValue(value3);

         sheet.getRange('G2').setValue(value4);

         str = "Success";

         SpreadsheetApp.flush();

         break;

      case "append\_row":

         var publish\_array = new Array();

         publish\_array [0] = date\_now;

         publish\_array [1] = time\_now;

         publish\_array [2] = value0;

         publish\_array [3] = value1;

         publish\_array [4] = value2;

         sheet.appendRow(publish\_array);

         str = "Success";

         SpreadsheetApp.flush();

         break;

    }

    return ContentService.createTextOutput(str);

  }

  else {

    return ContentService.createTextOutput("Error! Request body empty or in incorrect format.");

  }

}

DATA TRANSFER CODE

#include <Arduino.h>

#include <ESP8266WiFi.h>

#include <SPI.h>

#include <MFRC522.h>

#include <HTTPSRedirect.h>

#include<Wire.h>

const char \*GScriptId = "AKfycbyKXUoWH8XWuLrYyP5kbxoxPxv-whEieBFf0XszErxIGpO7Tirx\_Sv7Lw7k4FCRCGbQuA";

const char\* ssid = "12345678";

const char\* password = "12345678";

String payload\_base = "{\"command\": \"insert\_row\", \"sheet\_name\": \"Sheet1\", \"values\": ";

String payload = "";

const char\* host = "script.google.com";

const int httpsPort = 443;

const char\* fingerprint = "";

String url = String("/macros/s/") + GScriptId + "/exec";

HTTPSRedirect\* client = nullptr;

String student\_id;

int blocks[] = {4,5,6,8,9};

#define total\_blocks (sizeof(blocks) / sizeof(blocks[0]))

#define RST\_PIN 0 //D3

#define SS\_PIN 2 //D4

#define BUZZER 4 //D2

MFRC522 mfrc522(SS\_PIN, RST\_PIN);

MFRC522::MIFARE\_Key key;

MFRC522::StatusCode status;

int blockNum = 2;

byte bufferLen = 18;

byte readBlockData[18];

void setup() {

Serial.begin(9600);

delay(10);

Serial.println('\n');

SPI.begin();

WiFi.begin(ssid, password);

Serial.print("Connecting to ");

Serial.print(ssid); Serial.println(" ...");

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

delay(1000);

Serial.print(".");

}

Serial.println('\n');

Serial.println("Connection established!");

Serial.print("IP address:\t");

Serial.println(WiFi.localIP());

client = new HTTPSRedirect(httpsPort);

client->setInsecure();

client->setPrintResponseBody(true);

client->setContentTypeHeader("application/json");

Serial.print("Connecting to ");

Serial.println(host);

bool flag = false;

for(int i=0; i<5; i++){

int retval = client->connect(host, httpsPort);

if (retval == 1){

flag = true;

String msg = "Connected. OK";

Serial.println(msg);

delay(2000);

break;

}

else

Serial.println("Connection failed. Retrying...");

}

if (!flag){

Serial.print("Could not connect to server: ");

Serial.println(host);

delay(5000);

return;

}

delete client; // delete HTTPSRedirect object

client = nullptr; // delete HTTPSRedirect object

}

void loop() {

static bool flag = false;

if (!flag){

client = new HTTPSRedirect(httpsPort);

client->setInsecure();

flag = true;

client->setPrintResponseBody(true);

client->setContentTypeHeader("application/json");

}

if (client != nullptr){

if (!client->connected())

{client->connect(host, httpsPort);}

}

else{Serial.println("Error creating client object!");}

/\* Initialize MFRC522 Module \*/

mfrc522.PCD\_Init();

/\* Look for new cards \*/

/\* Reset the loop if no new card is present on RC522 Reader \*/

if ( ! mfrc522.PICC\_IsNewCardPresent()) {return;}

/\* Select one of the cards \*/

if ( ! mfrc522.PICC\_ReadCardSerial()) {return;}

/\* Read data from the same block \*/

Serial.println();

Serial.println(F("Reading last data from RFID..."));

String values = "", data;

for (byte i = 0; i < total\_blocks; i++) {

ReadDataFromBlock(blocks[i], readBlockData);

if(i == 0){

data = String((char\*)readBlockData);

data.trim();

student\_id = data;

values = "\"" + data + ",";

}

else if(i == total\_blocks-1){

data = String((char\*)readBlockData);

data.trim();

values += data + "\"}";

}

else{

data = String((char\*)readBlockData);

data.trim();

values += data + ",";

}

}

payload = payload\_base + values;

Serial.println("Publishing data...");

Serial.println(payload);

if(client->POST(url, host, payload)){

}

else{

Serial.println("Error while connecting");

}

delay(5000);

}

void ReadDataFromBlock(int blockNum, byte readBlockData[])

{

for (byte i = 0; i < 6; i++) {

key.keyByte[i] = 0xFF;

}

status = mfrc522.PCD\_Authenticate(MFRC522::PICC\_CMD\_MF\_AUTH\_KEY\_A, blockNum, &key, &(mfrc522.uid));

if (status != MFRC522::STATUS\_OK){

Serial.print("Authentication failed for Read: ");

Serial.println(mfrc522.GetStatusCodeName(status));

return;

}

else {

Serial.println("Authentication success");

}

status = mfrc522.MIFARE\_Read(blockNum, readBlockData, &bufferLen);

if (status != MFRC522::STATUS\_OK) {

Serial.print("Reading failed: ");

Serial.println(mfrc522.GetStatusCodeName(status));

return;

}

else {

readBlockData[16] = ' ';

readBlockData[17] = ' ';

Serial.println("Block was read successfully");

}

}