```
#include <SPI.h>
#include <MFRC522.h>
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>
#include <WiFiClientSecureBearSSL.h>
#include <LiquidCrystal_I2C.h>
//----------------------------------------
#define RST_PIN  D3
#define SS_PIN   D4
#define BUZZER   D8
//----------------------------------------
MFRC522 mfrc522(SS_PIN, RST_PIN);
MFRC522::MIFARE_Key key;
MFRC522::StatusCode status;
//----------------------------------------
/* Be aware of Sector Trailer Blocks */
int blockNum = 2;
/* Create another array to read data from Block */
/* Legthn of buffer should be 2 Bytes more than the size of Block (16 Bytes) */
byte bufferLen = 18;
byte readBlockData[18];
//----------------------------------------
String card_holder_name;
const String sheet_url = "Enter Google Script URL";  //Enter Google Script URL


//----------------------------------------
// Fingerprint for demo URL, expires on Monday, May 2, 2022 7:20:58 AM, needs to be updated well before this date
//const uint8_t fingerprint[20] = {0x9a, 0x87, 0x9b, 0x82, 0xe9, 0x19, 0x7e, 0x63, 0x8a, 0xdb, 0x67, 0xed, 0xa7, 0x09, 0xd9, 0x2f, 0x30, 0xde, 0xe7, 0x3c};
//9a 87 9b 82 e9 19 7e 63 8a db 67 ed a7 09 d9 2f 30 de e7 3c
```

```
//----------------------------------------
#define WIFI_SSID "XXXXXX"  //Enter WiFi Name
#define WIFI_PASSWORD "XXXXXX"  //Enter WiFi Password
//----------------------------------------

//Initialize the LCD display
LiquidCrystal_I2C lcd(0x3F, 16, 2);  //Change LCD Address to 0x27 if 0x3F doesnt work



/
**********************************************************************************
******************
 * setup() function

**********************************************************************************
******************/
void setup()
{
  //----------------------------------------------------
  /* Initialize serial communications with the PC */
  Serial.begin(9600);
  //Serial.setDebugOutput(true);

  lcd.begin();
  lcd.backlight();
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("  Initializing  ");
  for (int a = 5; a <= 10; a++) {
    lcd.setCursor(a, 1);
    lcd.print(".");
    delay(500);
  }
```

```
 //-------------------------------------------------
 //WiFi Connectivity
 Serial.println();
 Serial.print("Connecting to AP");
 WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
 while (WiFi.status() != WL_CONNECTED){
  Serial.print(".");
  delay(200);
 }
 Serial.println("");
 Serial.println("WiFi connected.");
 Serial.println("IP address: ");
 Serial.println(WiFi.localIP());
 Serial.println();
 //-------------------------------------------------
 /* Set BUZZER as OUTPUT */
 pinMode(BUZZER, OUTPUT);
 //-------------------------------------------------
 /* Initialize SPI bus */
 SPI.begin();
 //-------------------------------------------------

}




/
**********************************************************************************
******************
 * loop() function
```

```
**********************************************************************************
******************/
 void loop()
{
 //-------------------------------------------------
 lcd.clear();
 lcd.setCursor(0, 0);
 lcd.print(" Scan your Card ");
 /* 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..."));
 ReadDataFromBlock(blockNum, readBlockData);
 /* If you want to print the full memory dump, uncomment the next line */
 //mfrc522.PICC_DumpToSerial(&(mfrc522.uid));

 /* Print the data read from block */
 Serial.println();
 Serial.print(F("Last data in RFID:"));
 Serial.print(blockNum);
 Serial.print(F(" --> "));
 for (int j=0 ; j<16 ; j++)
 {
  Serial.write(readBlockData[j]);
  lcd.clear();
```

```
  lcd.setCursor(0, 0);
  lcd.print("Hey " + String((char*)readBlockData) + "!");

  //lcd.print(String((char*)readBlockData));
  //lcd.print("!")
 }
 Serial.println();
 //-------------------------------------------------
 digitalWrite(BUZZER, HIGH);
 delay(200);
 digitalWrite(BUZZER, LOW);
 delay(200);
 digitalWrite(BUZZER, HIGH);
 delay(200);
 digitalWrite(BUZZER, LOW);
 //-------------------------------------------------

 //
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
M
 if (WiFi.status() == WL_CONNECTED) {
  //-------------------------------------------------------------------------------
  std::unique_ptr<BearSSL::WiFiClientSecure>client(new BearSSL::WiFiClientSecure);
  //-------------------------------------------------------------------------------
  //client->setFingerprint(fingerprint);
  // Or, if you want to ignore the SSL certificate
  //then use the following line instead:
  client->setInsecure();
  //-----------------------------------------------------------------
  card_holder_name = sheet_url + String((char*)readBlockData);
  card_holder_name.trim();
  Serial.println(card_holder_name);
```

```
//-------------------------------------------------------------------
HTTPClient https;
Serial.print(F("[HTTPS] begin...\n"));
//-------------------------------------------------------------------

//
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNN
if (https.begin(*client, (String)card_holder_name)){
 //-------------------------------------------------------------------
 // HTTP
 Serial.print(F("[HTTPS] GET...\n"));
 // start connection and send HTTP header
 int httpCode = https.GET();
 //-------------------------------------------------------------------
 // httpCode will be negative on error
 if (httpCode > 0) {
  // HTTP header has been send and Server response header has been handled
  Serial.printf("[HTTPS] GET... code: %d\n", httpCode);
  // file found at server
lcd.setCursor(0, 1);
lcd.print(" Data Recorded ");
delay(2000);
 }
 //-------------------------------------------------------------------
 else
 {Serial.printf("[HTTPS] GET... failed, error: %s\n", https.errorToString(httpCode).c_str());}
 //-------------------------------------------------------------------
 https.end();
 delay(1000);
}
```

```
  //
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNN
  else {
    Serial.printf("[HTTPS} Unable to connect\n");
  }
  //
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNN
 }
 //
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
M
}




/
***********************************************************************************
******************
 * ReadDataFromBlock() function

***********************************************************************************
******************/
void ReadDataFromBlock(int blockNum, byte readBlockData[])
{
 //----------------------------------------------------------------------------
 /* Prepare the ksy for authentication */
 /* All keys are set to FFFFFFFFFFFFh at chip delivery from the factory */
 for (byte i = 0; i < 6; i++) {
  key.keyByte[i] = 0xFF;
 }
 //----------------------------------------------------------------------------
 /* Authenticating the desired data block for Read access using Key A */
```

```
 status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, blockNum, &key,
&(mfrc522.uid));
 //-------------------------------------------------------------------------s
 if (status != MFRC522::STATUS_OK){
   Serial.print("Authentication failed for Read: ");
   Serial.println(mfrc522.GetStatusCodeName(status));
   return;
 }
 //-------------------------------------------------------------------------
 else {
  Serial.println("Authentication success");
 }
 //-------------------------------------------------------------------------
 /* Reading data from the Block */
 status = mfrc522.MIFARE_Read(blockNum, readBlockData, &bufferLen);
 if (status != MFRC522::STATUS_OK) {
  Serial.print("Reading failed: ");
  Serial.println(mfrc522.GetStatusCodeName(status));
  return;
 }
 //-------------------------------------------------------------------------
 else {
  Serial.println("Block was read successfully");
 }
 //-------------------------------------------------------------------------
}
```