

```

1  //-----Including the libraries.
2  #include <Wire.h>
3  #include <LiquidCrystal_I2C.h>
4  #include <SPI.h>
5  #include <MFRC522.h>
6  #include <WiFi.h>
7  #include <HTTPClient.h>
8  #include <vector>
9
10
11  #define SS_PIN 5
12  #define RST_PIN 4
13  #define BTN_PIN 15
14
15  const char* ssid = "Atchyuth Kalla";
16  const char* password = "00000000";
17
18
19
20
21
22  void http Req(String str_modes, String str_uid);
23  String getValue(String data, char separator, int index);
24  int getUID();
25  void byteArray_to_string(byte array[], unsigned int len, char buffer[]);
26
27  String Web_App_URL =
  "https://script.google.com/macros/s/AKfycbyktSZdylSqJACPWeeeKbWfeBbXgXmpkMPmyH7wQJD1h5NPN
  Oe2Ht_0rpC760xJ9sA3oA/exec";
28
29  String reg_Info = "";
30  String atc_Info = "";
31  String atc_Name = "";
32  String atc_Date = "";
33  String atc_Time_In = "";
34  String atc_Time_Out = "";
35
36  int lcdColumns = 16;
37  int lcdRows = 2;
38
39  int readsuccess;
40  char str[32] = "";
41  String UID_Result = "-----";
42  String modes = "atc";
43
44  LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);
45  MFRC522 mfrc522(SS_PIN, RST_PIN);
46
47  void setup() {
48    Serial.begin(115200);
49    Serial.println();
50    delay(1000);
51
52    pinMode(BTN_PIN, INPUT_PULLUP);
53
54    lcd.init();
55    lcd.backlight();
56
57    lcd.clear();
58
59    delay(500);
60
61    SPI.begin();
62    mfrc522.PCD_Init();
63
64    delay(500);
65
66
67    lcd.setCursor( 0, 0);

```

```

68     lcd.print("Google  Sheets");
69     lcd.setCursor( 0, 1);
70     lcd.print("AttendanceSystem");
71
72     delay(3000);
73     lcd.clear();
74
75     Serial.println();
76     Serial.println("-----");
77     Serial.println("WIFI mode : STA");
78     WiFi.mode(WIFI_STA);
79     Serial.println("-----");
80
81     Serial.println();
82     Serial.println("-----");
83     Serial.print("Connecting to ");
84     Serial.println(ssid);
85     WiFi.begin(ssid, password);
86
87     int connecting_process_timed_out = 20;
88     connecting_process_timed_out = connecting_process_timed_out * 2;
89     while (WiFi.status() != WL_CONNECTED) {
90         Serial.print(".");
91         lcd.setCursor( 0, 0);
92         lcd.print("Connecting WIFI");
93         delay(250);
94         lcd.clear();
95         delay(250);
96
97         if (connecting_process_timed_out > 0) connecting_process_timed_out--;
98         if (connecting_process_timed_out == 0) {
99             delay(1000);
100             ESP.restart();
101         }
102     }
103
104     Serial.println();
105     Serial.println("WiFi connected");
106     Serial.println("-----");
107
108     lcd.clear();
109     lcd.setCursor( 0, 0);
110     lcd.print("WiFi connected");
111     delay(2000);
112     lcd.clear();
113     delay(500);
114
115 }
116
117
118 void loop() {
119     int BTN_State = digitalRead(BTN_PIN);
120
121     if (BTN_State == LOW) {
122         lcd.clear();
123         if (modes == "atc") {
124             modes = "reg";
125         } else if (modes == "reg") {
126             modes = "atc";
127         }
128         delay(500);
129     }
130
131     readsuccess = getUID();
132
133     if (modes == "atc") {
134
135         lcd.setCursor(0, 0);
136         lcd.print("ATTENDANCE");

```

```

137     lcd.setCursor( 0, 1);
138     lcd.print("Please tap card");
139
140
141
142     if (readsuccess) {
143         lcd.clear();
144         delay(500);
145         lcd.setCursor( 0, 0);
146         lcd.print("Getting UID");
147         lcd.setCursor( 0, 1);
148         lcd.print("Successfully");
149         delay(1000);
150
151         http Req(modes, UID_Result);
152     }
153 }
154
155 if (modes == "reg") {
156     lcd.setCursor( 0, 0);
157     lcd.print("REGISTRATION");
158     lcd.setCursor( 0, 1);
159     lcd.print("Tap your card");
160
161     if (readsuccess) {
162         lcd.clear();
163         delay(500);
164         lcd.setCursor( 0, 0);
165         lcd.print("Getting UID");
166         lcd.setCursor( 0, 1);
167         lcd.print("Successfully");
168
169         delay(1000);
170
171         http Req(modes, UID_Result);
172     }
173 }
174
175 delay(10);
176 }
177
178
179 void http Req(String str_modes, String str_uid) {
180     if (WiFi.status() == WL_CONNECTED) {
181         String http_req_url = Web_App_URL + "?sts=" + str_modes + "&uid=" + str_uid;
182
183         Serial.println();
184         Serial.println("-----");
185         Serial.println("Sending request to Google Sheets...");
186         Serial.print("URL : ");
187         Serial.println(http_req_url);
188
189         HTTPClient http;
190         http.begin(http_req_url);
191
192         int httpCode = http.GET();
193         Serial.print("HTTP Status Code : ");
194         Serial.println(httpCode);
195
196         String payload;
197         if (httpCode == 30) {
198             String newUrl = http.header("Location");
199             Serial.println("Redirecting to: " + newUrl);
200
201             http.end(); // close current connection
202
203             // follow the redirect
204             http.begin(newUrl);
205             httpCode = http.GET();

```

```

206     Serial.print("HTTP Status Code (after redirect) : ");
207     Serial.println(httpCode);
208 }
209
210 if (httpCode > 0) {
211     payload = http.getString();
212     Serial.println("Payload : " + payload);
213 }
214
215 Serial.println("-----");
216 http.end();
217
218 // The rest of your code remains unchanged...
219 String sts_Res = getValue(payload, ',', 0);
220
221 if (sts_Res == "OK") {
222     if (str_modes == "atc") {
223         atc_Info = getValue(payload, ',', 1);
224
225         if (atc_Info == "TI_Successful" || atc_Info == "TO_Successful") {
226             atc_Name = getValue(payload, ',', 2);
227             atc_Date = getValue(payload, ',', 3);
228             atc_Time_In = getValue(payload, ',', 4);
229
230             if (atc_Info == "TO_Successful") {
231                 atc_Time_Out = getValue(payload, ',', 5);
232             }
233
234             int name_Length = atc_Name.length();
235             int pos = 0;
236             if (name_Length > 0 && name_Length <= lcdColumns) {
237                 pos = map(name_Length, 1, lcdColumns, 0, (lcdColumns / 2) - 1);
238                 pos = ((lcdColumns / 2) - 1) - pos;
239             } else if (name_Length > lcdColumns) {
240                 atc_Name = atc_Name.substring(0, lcdColumns);
241             }
242
243             lcd.clear();
244             delay(500);
245
246             lcd.setCursor( 0, 0);
247
248             lcd.print("IN: ");
249             lcd.print(atc_Time_In);
250             lcd.setCursor( 0, 1);
251             lcd.print("OUT: ");
252             if (atc_Info == "TO_Successful") {
253                 lcd.print(atc_Time_Out);
254             }
255             delay(5000);
256             lcd.clear();
257             delay(500);
258         } else if (atc_Info == "atcInf01") {
259             lcd.clear();
260             delay(500);
261             lcd.setCursor( 0, 0);
262             lcd.print("Attendance");
263             lcd.setCursor( 0, 1);
264             lcd.print("Completed");
265             delay(5000);
266             lcd.clear();
267             delay(500);
268         } else if (atc_Info == "atcErr01") {
269             lcd.clear();
270             delay(500);
271             lcd.setCursor( 0, 0);
272             lcd.print("Error!");
273             lcd.setCursor( 8, 0);
274             lcd.print("Card not");

```

```

275         lcd.setCursor( 0, 1);
276         lcd.print("registered");
277         delay(5000);
278         lcd.clear();
279         delay(500);
280     }
281
282     atc_Info = "";
283     atc_Name = "";
284     atc_Date = "";
285     atc_Time_In = "";
286     atc_Time_Out = "";
287 }
288
289 if (str_modes == "reg") {
290     reg_Info = getValue(payload, ',', 1);
291
292     if (reg_Info == "R_Successful") {
293         lcd.clear();
294         delay(500);
295         lcd.setCursor( 0, 0);
296         lcd.print("Registration");
297         lcd.setCursor( 0, 1);
298         lcd.print("Successful");
299         delay(5000);
300         lcd.clear();
301         delay(500);
302     } else if (reg_Info == "regErr01") {
303         lcd.clear();
304         delay(500);
305         lcd.setCursor( 0, 0);
306         lcd.print("Error!");
307         lcd.setCursor( 8, 0);
308         lcd.print("Already");
309         lcd.setCursor( 0, 1);
310         lcd.print("Registered");
311         delay(5000);
312         lcd.clear();
313         delay(500);
314     }
315
316     reg_Info = "";
317 }
318 }
319 } else {
320     lcd.clear();
321     delay(500);
322     lcd.setCursor( 0, 0);
323     lcd.print("Error!");
324     lcd.setCursor( 0, 1);
325     lcd.print("WiFi disconnected");
326     delay(3000);
327     lcd.clear();
328     delay(500);
329 }
330 }
331
332 String getValue(String data, char separator, int index) {
333     int found = 0;
334     int strIndex[] = { 0, -1 };
335     int maxIndex = data.length() - 1;
336
337     for (int i = 0; i <= maxIndex && found <= index; i++) {
338         if (data.charAt(i) == separator || i == maxIndex) {
339             found++;
340             strIndex[0] = strIndex[1] + 1;
341             strIndex[1] = (i == maxIndex) ? i + 1 : i;
342         }
343     }

```

```

344     return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
345 }
346 void byteArray_to_string(byte array[], unsigned int len, char buffer[]) {
347     for (unsigned int i = 0; i < len; i++) {
348         byte nib1 = (array[i] >> 4) & 0x0F;
349         byte nib2 = (array[i] >> 0) & 0x0F;
350         buffer[i * 2 + 0] = nib1 < 0xA ? '0' + nib1 : 'A' + nib1 - 0xA;
351         buffer[i * 2 + 1] = nib2 < 0xA ? '0' + nib2 : 'A' + nib2 - 0xA;
352     }
353     buffer[len * 2] = '\0';
354 }
355 int getUID() {
356     if (!mfr522.PICC_IsNewCardPresent()) {
357         return 0;
358     }
359     if (!mfr522.PICC_ReadCardSerial()) {
360         return 0;
361     }
362
363     byteArray_to_string(mfr522.uid.uidByte, mfr522.uid.size, str);
364     UID_Result = str;
365
366     mfr522.PICC_HaltA();
367     mfr522.PCD_StopCrypto1();
368
369     return 1;
370 }

```