

Smart Home

```
1 // Smart Home Practical
2
3 /* Includes Section */
4 #include <Wire.h>
5 #include <ThreeWire.h>
6 #include <RtcDS1302.h>
7 #include <LiquidCrystal_I2C.h>
8 #include <Keypad.h>
9
10 /* Variables Section */
11 #define RTC_CLOCK 2
12 #define RTC_DATA 3
13 #define RTC_RESET 4
14 #define KP_R1 5
15 #define KP_R2 6
16 #define KP_R3 7
17 #define KP_R4 8
18 #define KP_C1 9
19 #define KP_C2 10
20 #define KP_C3 11
21 #define KP_C4 12
22 #define ALARM A3
23 #define DOOR A2
24 #define LDR_OUT A1
25 #define OUTDOOR_LIGHT A0
26
27 // Number of pressed keys
28 int entered_Num = 5;
29
30 // LDR Variable
31 bool LDR_Status = 0;
32
33 // ALARM
34 bool ALARM_OFF_STATUS = 0;
35
36 // Password variables
37 String pad;
38 char keyPressed;
39 int pass_Flag = 0;
40
41 // Date and Time variables
42 const long Event_Time = 30000;
43 unsigned long Previous_Time = 0;
44
45 // DOOR Timer
46 unsigned long time_for_action;
47 #define OUTDOORINTERVAL 5000
48
49 // Keypad variables
50 const byte ROWS = 4; // Set up four rows for the Keypad
51 const byte COLS = 4; // Set up four columns for the Keypad
52
53 // Set up the modules
54 LiquidCrystal_I2C lcd(0x27, 16, 2); // Setting the connection with the I2C #I2C Address: 0x27
55 ThreeWire myWire(RTC_DATA, RTC_CLOCK, RTC_RESET);
56 RtcDS1302<ThreeWire> Rtc(myWire);
57
58 char Keys[ROWS][COLS] =
59 {
60   {'1', '2', '3', 'A'},
61   {'4', '5', '6', 'B'},
62   {'7', '8', '9', 'C'},
63   {'*', '0', '#', 'D'}
64 };
65
66
```

```

67 // Address the pins of the Keypad
68 byte rowPins[ROWS] = {KP_R1,KP_R2,KP_R3,KP_R4};
69 byte colPins[COLS] = {KP_C1,KP_C2,KP_C3,KP_C4};
70
71 Keypad customKeypad = Keypad(makeKeymap(Keys), rowPins, colPins, ROWS, COLS);
72
73 void setup()
74 {
75     pinMode(LDR_OUT, INPUT);
76     pinMode(ALARM, OUTPUT);
77     pinMode(ACCESS_GRANTED, OUTPUT);
78     pinMode(OUTDOOR_LIGHT, OUTPUT);
79     pinMode(DOOR, OUTPUT);
80
81     digitalWrite(ALARM, LOW);
82     digitalWrite(DOOR, HIGH);
83
84     // Starting Serial connection
85     Serial.begin(115200);
86     int myTimeout = 100; // milliseconds for Serial.readString
87     Serial.setTimeout(myTimeout);
88
89     // Starting the LCD
90     lcd.init();
91     lcd.backlight();
92     lcd.clear();
93
94     // Starting the Real Time Clock
95     Rtc.Begin();
96     RtcDateTime currentTime = RtcDateTime(__DATE__ , __TIME__);
97     Rtc.SetDateTime(currentTime);
98 }
99
100 void loop()
101 {
102     StartUP();
103     //Set_Date_Time();
104     String password = "123";    // Set up the password
105
106     OUTDOOR();
107     readKeypad();
108     if(keyPressed == '#')
109     {
110         if(pad == password)
111         {
112             ACCESS_GRANTED();
113         }
114         else
115         {
116             ACCESS_DENIED();
117         }
118     }
119 }
120
121 void Set_Date_Time()
122 {
123     if(pad == 0)
124     {
125         RtcDateTime now = Rtc.GetDateTime();    // Get the current Date & Time
126
127         unsigned long Current_Time = millis();
128
129         if(Current_Time - Previous_Time >= Event_Time) // Timer to change every one sec
130         {
131             lcd.setCursor(0,0);
132             lcd.print("Date: ");
133             lcd.print(now.Day());
134             lcd.print("/");
135             lcd.print(now.Month());
136             lcd.print("/");
137             lcd.print(now.Year());
138
139

```

```
139     lcd.setCursor(0,1);
140     lcd.print("Time: ");
141     lcd.print(now.Hour());
142     lcd.print(":");
143     lcd.print(now.Minute());
144     lcd.print(":");
145     lcd.print(now.Second());
146 }
147 }
148 }
149
150 void OUTDOOR()
151 {
152     LDR_Status = digitalRead(LDR_OUT);
153
154     String Serial_Read;
155     Serial_Read = Serial.readStringUntil('\n');
156
157     if(1 == LDR_Status)
158     {
159         digitalWrite(OUTDOOR_LIGHT, LOW);
160     }
161     else
162     {
163         digitalWrite(OUTDOOR_LIGHT, HIGH);
164     }
165 }
166
167 void readKeypad()
168 {
169     keyPressed = customKeypad.getKey();
170     if (keyPressed != '#')
171     {
172         String typed_Pass = String(keyPressed);
173         pad += typed_Pass;
174         lcd.setCursor(0, 0);
175         lcd.print("Taking password:");
176         if(keyPressed)
177         {
178             lcd.setCursor(entered_Num++, 1);
179             lcd.print("#");
180         }
181     }
182 }
183
184 void ACCESS_GRANTED()
185 {
186     lcd.clear();
187     lcd.setCursor(0,0);
188     lcd.print("Correct Password");
189     lcd.setCursor(0,1);
190     lcd.print("Access Granted:");
191
192     String Serial_Read;
193
194     Serial_Read = Serial.readStringUntil('\n');
195     Serial.println(Serial_Read);
196
197     if(Serial_Read[0] == '0')
198     {
199         StartUP();
200     }
201
202     digitalWrite(DOOR, LOW);
203     delay(1000);
204     digitalWrite(DOOR, HIGH);
205
206     delay(5000);
207
208     pad = "";
209
210     lcd.clear();
211 }
```

```
212     pass_Flag = 0;
213     entered_Num = 5;
214 }
215
216 void ACCESS_DENIED()
217 {
218     if(2 != pass_Flag)
219     {
220         lcd.clear();
221         lcd.setCursor(0,0);
222         lcd.print(" Incorrect Pass ");
223         lcd.setCursor(0,1);
224         lcd.print("Access Denied!!");
225
226         pad = "";
227
228         delay(5000);
229         lcd.clear();
230         entered_Num = 5;
231         pass_Flag++;
232     }
233     else
234     {
235         lcd.clear();
236         while(pass_Flag != 0)
237         {
238             lcd.setCursor(0,0);
239             lcd.print(" ALERT!!!");
240             lcd.setCursor(0,1);
241             lcd.print(" Home Locked! ");
242             digitalWrite(ALARM, HIGH);
243             delay(500);
244             digitalWrite(ALARM, LOW);
245             delay(500);
246             StartUP();
247         }
248         lcd.clear();
249         entered_Num = 5;
250     }
251 }
252
253 void TURN_OFF_ALARM()
254 {
255     String Serial_Read;
256     Serial_Read = Serial.readStringUntil('\n');
257
258     if(Serial_Read[0] == '3')
259     {
260         pass_Flag == 0;
261     }
262 }
263
264 void Fire_Alert()
265 {
266     lcd.setCursor(0,0);
267     lcd.print(" FIRE ALERT!! ");
268     lcd.setCursor(0,1);
269     lcd.print("Flame Detected!!");
270     digitalWrite(ALARM, HIGH);
271     delay(500);
272     digitalWrite(ALARM, LOW);
273     delay(500);
274 }
275
276 void StartUP()
277 {
278     while(Serial.available() > 0)
279     {
280         String Serial_Read;
281         long message;
282
283         Serial_Read = Serial.readStringUntil('\n');
284     }
```

```
285 Serial.println(Serial_Read);
286
287 if(Serial_Read[0] == '2')
288 {
289     Fire_Alert();
290 }
291 else if(Serial_Read[0] == '1')
292 {
293     ACCESS_GRANTED();
294 }
295 else if(Serial_Read[0] == '0')
296 {
297     digitalWrite(DOOR, HIGH);
298 }
299 else if(Serial_Read[0] == '3')
300 {
301     pass_Flag = 0;
302 }
303 else if(Serial_Read[0] == '4')
304 {
305     ACCESS_DENIED();
306 }
307 else if(Serial_Read[0] == '7')
308 {
309     digitalWrite(OUTDOOR_LIGHT, LOW);
310     delay(2000);
311 }
312 else if(Serial_Read[0] == '8')
313 {
314     digitalWrite(OUTDOOR_LIGHT, HIGH);
315     delay(2000);
316 }
317 lcd.clear();
    break;
}
```