**SOURCE CODE:**

#include<LiquidCrystal.h>

LiquidCrystal lcd(7, 6, 5, 4, 3, 2);

#include <SoftwareSerial.h> //Software Serial library

SoftwareSerial espSerial(11, 10); //Pin 2 and 3 act as RX and TX. Connect them to TX and RX of ESP8266

#define DEBUG true

String mySSID = "Ravi1"; // WiFi SSID

String myPWD = "12345678"; // WiFi Password

String myAPI = "N5OQA82W1E1F5Q4X"; // API Key

String myHOST = "api.thingspeak.com";

String myPORT = "80";

String myFIELD = "field1";

String myFIELD2 = "field2";

int sendVal;

#include <Wire.h>

#include "MAX30105.h"

#include <avr/wdt.h>

#include "heartRate.h"

MAX30105 particleSensor;

const byte RATE\_SIZE = 4; //Increase this for more averaging. 4 is good.

byte rates[RATE\_SIZE]; //Array of heart rates

byte rateSpot = 0;

long lastBeat = 0; //Time at which the last beat occurred

float beatsPerMinute;

int beatAvg;

int count = 0;

const int lm35\_pin = A0;

void setup()

{

lcd.begin(16, 2);

Serial.begin(115200);

Serial.println("Initializing...");

lcd.setCursor(0,0); //sets the cursor at row 0 column 0

lcd.print("Initializing"); // prints 16x2 LCD MODULE

// Initialize sensor

if (!particleSensor.begin(Wire, I2C\_SPEED\_FAST)) //Use default I2C port, 400kHz speed

{

Serial.println("MAX30105 was not found. Please check wiring/power. ");

lcd.clear();

lcd.setCursor(0,0); //sets the cursor at row 0 column 0

lcd.print("PO NOT FOUND"); // prints 16x2 LCD MODULE

while (1);

}

Serial.println("Place your index finger on the sensor with steady pressure.");

lcd.clear();

lcd.setCursor(0,0);

lcd.print("MAX30105 Found");

particleSensor.setup(); //Configure sensor with default settings

particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is running

particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED

espSerial.begin(115200);

espData("AT+RST", 1000, DEBUG); //Reset the ESP8266 module

espData("AT+CWMODE=1", 1000, DEBUG); //Set the ESP mode as station mode

espData("AT+CWJAP=\""+ mySSID +"\",\""+ myPWD +"\"", 1000, DEBUG); //Connect to WiFi network

delay(1000);

}

void loop()

{

count = count+1;

if(count<=500)

{

long irValue = particleSensor.getIR();

if (checkForBeat(irValue) == true)

{

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Calibrating...");

//We sensed a beat!

long delta = millis() - lastBeat;

lastBeat = millis();

beatsPerMinute = 60 / (delta / 1000.0);

if (beatsPerMinute < 255 && beatsPerMinute > 20)

{

rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array

rateSpot %= RATE\_SIZE; //Wrap variable

//Take average of readings

beatAvg = 0;

for (byte x = 0 ; x < RATE\_SIZE ; x++)

beatAvg += rates[x];

beatAvg /= RATE\_SIZE;

}

}

Serial.print("IR=");

Serial.print(irValue);

Serial.print(", BPM=");

Serial.print(beatsPerMinute);

Serial.print(", Avg BPM=");

Serial.print(beatAvg);

lcd.setCursor(0,1);

lcd.print("Heart Rate:");

lcd.setCursor(11,1);

lcd.print(beatAvg);

if (irValue < 50000)

Serial.print(" No finger?");

Serial.println();

}

else

{

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Calibrated");

int temp\_adc\_val;

float temp\_val;

temp\_adc\_val = analogRead(lm35\_pin); /\* Read Temperature \*/

temp\_val = (temp\_adc\_val \* 4.88); /\* Convert adc value to equivalent voltage \*/

temp\_val = (temp\_val/10); /\* LM35 gives output of 10mv/°C \*/

Serial.print("Temperature = ");

Serial.print(temp\_val);

Serial.print(" Degree Celsius\n");

delay(1000);

lcd.setCursor(0,1);

lcd.print("HR=");

lcd.setCursor(4,1);

lcd.print(beatAvg);

lcd.setCursor(8,1);

lcd.print("T=");

lcd.setCursor(10,1);

lcd.print(temp\_val);

lcd.setCursor(14,1);

lcd.print("'C");

String sendData = "GET /update?api\_key="+ myAPI +"&"+ myFIELD +"="+String(beatAvg) +"&"+ myFIELD2 +"="+String(temp\_val);

espData("AT+CIPMUX=1", 1000, DEBUG); //Allow multiple connections

espData("AT+CIPSTART=0,\"TCP\",\""+ myHOST +"\","+ myPORT, 1000, DEBUG);

espData("AT+CIPSEND=0," +String(sendData.length()+4),1000,DEBUG);

espSerial.find(">");

espSerial.println(sendData);

espData("AT+CIPCLOSE=0",1000,DEBUG);

delay(10000);

count=0;

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Data Sent");

//reboot();

}

}

String espData(String command, const int timeout, boolean debug)

{

Serial.print("AT Command ==> ");

Serial.print(command);

Serial.println(" ");

String response = "";

espSerial.println(command);

long int time = millis();

while ( (time + timeout) > millis())

{

while (espSerial.available())

{

char c = espSerial.read();

response += c;

}

}

if (debug)

{

//Serial.print(response);

}

return response;

}

void reboot() {

wdt\_disable();

wdt\_enable(WDTO\_15MS);

while (1) {}

}