Project Report

Remote Health Monitoring System to Measure Body Temperature, Humidity level and Detect Falls in Older Adults

Telehealth and Virtual Hospitals INFO-10298

December 10, 2021

Fall 2021

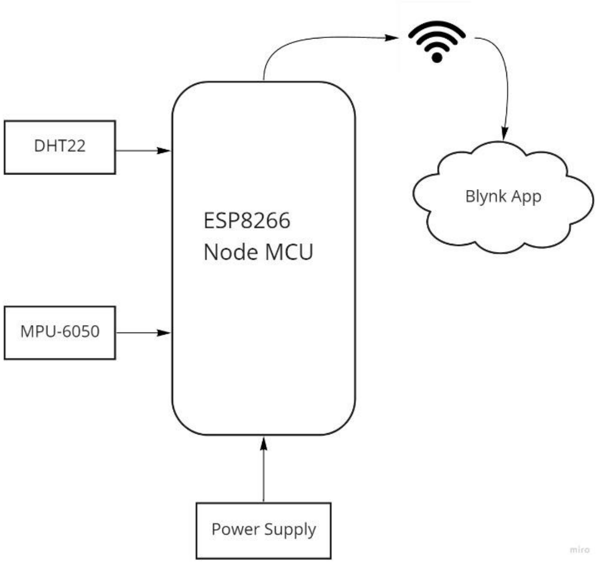
By:

Maria Hussain Rassawala

Project Objective

Elderly populations are prone to hyperthermia and hypothermia due to innate physiological changes associated with age, chronic diseases, and medications, because of this it is important to employ a remote monitoring system that detects falls and temperature of the patients. The ability to detect a fall on time and provide prompt assistance can help reduce serious injuries.

Block Diagram



Devices, Sensors, and Software used

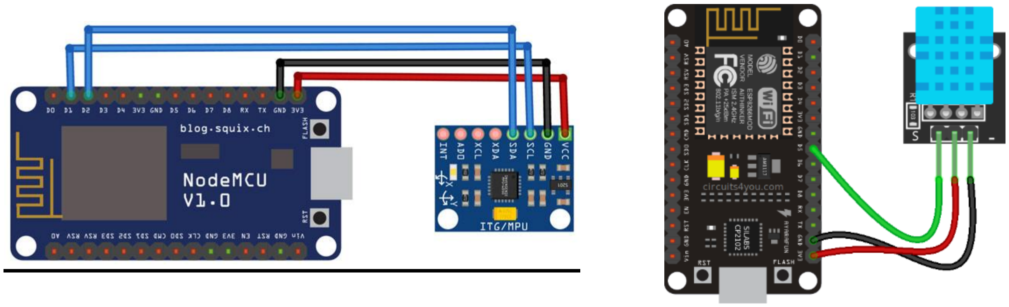
1. **Type and number of sensors:** 4 sensors will be used for the project. Accelerometer and gyroscope (MPU-6050) sensor will be used for fall detection. DHT22 sensor will be used to detect the person's body temperature and humidity level.
2. **Microcontroller and other Hardware used:**

ESP8266 NodeMCU	DHT22 sensor	MPU-6050 sensor	Breadboard	Jumper wires	Micro USB cable

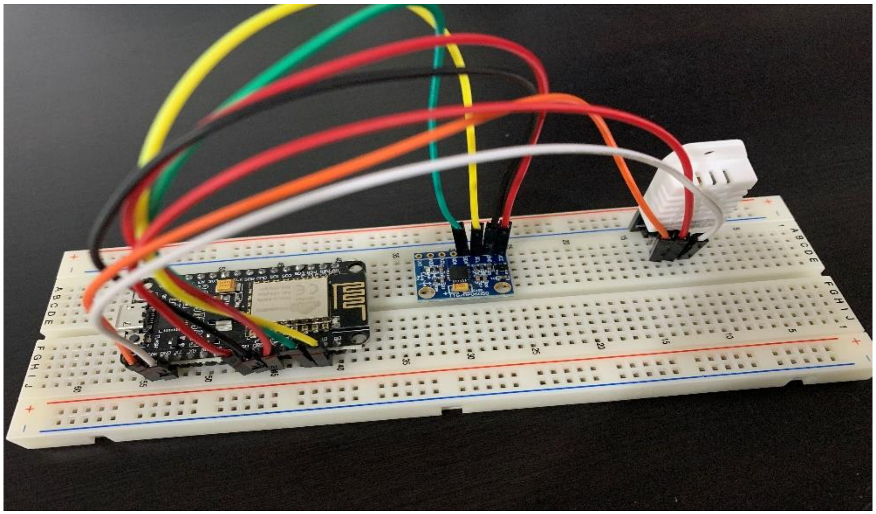
3. Network and Communication:

- Arduino UNO, programming in IDE environment
- ESP8266 NodeMCU connects to existing WiFi network & creates a hotspot to which Blynk app connects to by using wifi credentials. Then sends data from sensors to the Blynk app.
- Blynk IoT cloud/Blynk app

Wiring Diagram

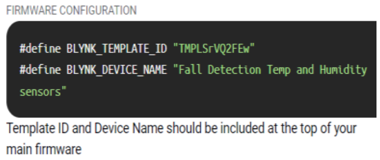


SCL pin of MPU-6050	D1 – GPI05
SDA pin of MPU-6050	D2 – GPI04
Data pin of DHT22	D5 – GPI014



Creating Blynk Dashboard

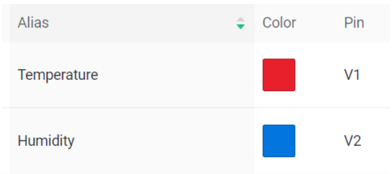
1. Create an account on https://blynk.io/ and create blynk Template with the name “Fall Detection Temp and Humidity sensors”. This will give the **BLYNK_TEMPLATE_ID** and **BLYNK_DEVICE_NAME** which will uniquely identify the device in the code.
2. Add temperature and humidity datastreams as **V1** and **V2** respectively. Add their widgets.
3. Add an event named **‘fall_detected’** and set its type to **‘warning’**.



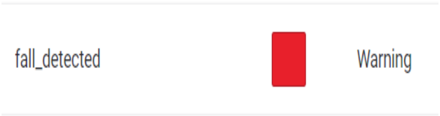
FIRMWARE CONFIGURATION

```
#define BLYNK_TEMPLATE_ID "TMPLSrVQ2FEw"
#define BLYNK_DEVICE_NAME "Fall Detection Temp and Humidity
sensors"
```

Template ID and Device Name should be included at the top of your main firmware



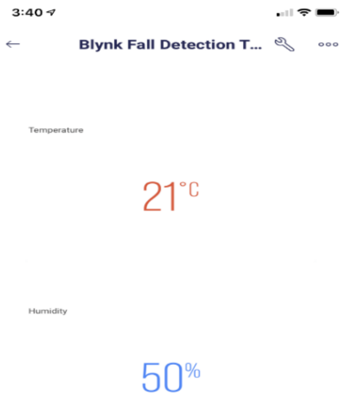
Alias	Color	Pin
Temperature	🟥	V1
Humidity	🟦	V2



fall_detected	🟥	Warning

Functionality

NodeMCU	Connects to WiFi and acts as a hotspot to connect with Blynk cloud. Sends data of humidity and temperature to Blynk app. Detects fall and sends notification to Blynk app.
DHT22 sensor	Measures humidity and temperature values
MCU-6050 sensor	Measures acceleration of motion and orientation values.
Jumper wires	Connect sensors with ESP8266 NodeMCU
Micro-USB cable	Connects NodeMCU to laptop for power supply



3:40

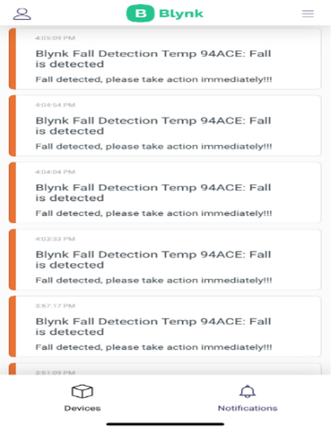
Blynk Fall Detection T...

Temperature

21°C

Humidity

50%



4:05:09 PM
Blynk Fall Detection Temp 94ACE: Fall is detected
Fall detected, please take action immediately!!!

4:04:54 PM
Blynk Fall Detection Temp 94ACE: Fall is detected
Fall detected, please take action immediately!!!

4:04:04 PM
Blynk Fall Detection Temp 94ACE: Fall is detected
Fall detected, please take action immediately!!!

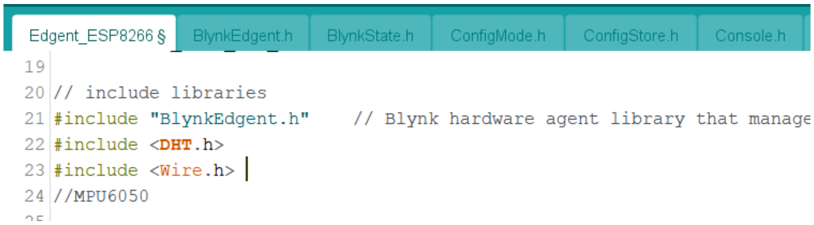
4:03:33 PM
Blynk Fall Detection Temp 94ACE: Fall is detected
Fall detected, please take action immediately!!!

3:57:17 PM
Blynk Fall Detection Temp 94ACE: Fall is detected
Fall detected, please take action immediately!!!

3:51:09 PM

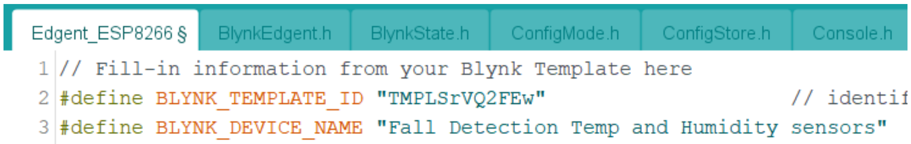
<u>Code</u>

1. Install libraries: In this code, other than ESP8266wifi library, we have included Blynk and Blynkedgent libraries which helps in communication with Blynk cloud. DHT.h library for the humidity and temperature sensor, and Wire.h which helps in communication with the SCL and SDA pins of MPU6050 with Node MCU.



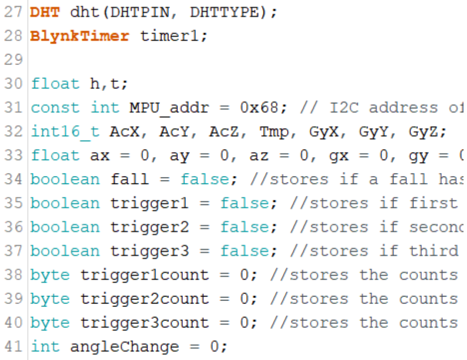
```

// include libraries
#include "BlynkEdgent.h"    // Blynk hardware agent library that manage
#include <DHT.h>
#include <Wire.h>
//MPU6050
```

2. Add Blynk template ID and name at the top of the code.

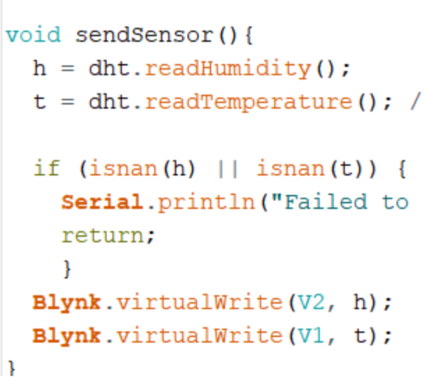


```
// Fill-in information from your Blynk Template here
#define BLYNK_TEMPLATE_ID "TMPLSrVQ2FEw"                  // identi
#define BLYNK_DEVICE_NAME "Fall Detection Temp and Humidity sensors"
```

3. BlynkTimer allows us to send data periodically with given intervals. Assign variables for temperature (t), humidity (h), accelerometer (ax, ay, az) and gyroscope (gx, gy, gz). The void sendSensor() function will read h and t values and send it to Blynk cloud by Blynk.virtualWrite function by reading the pins V1 and V2 that we have set on the Blynk dashboard.

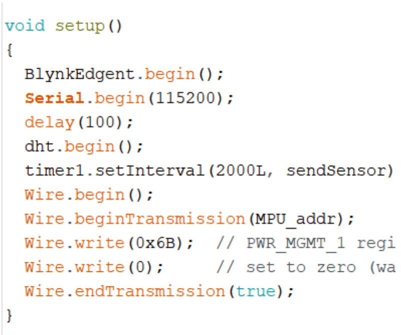


```
DHT dht(DHTPIN, DHTTYPE);
BlynkTimer timer1;

float h,t;
const int MPU_addr = 0x68; // I2C address o
int16_t AcX, AcY, AcZ, Tmp, GyX, GyY, GyZ;
float ax = 0, ay = 0, az = 0, gx = 0, gy =
boolean fall = false; //stores if a fall ha
boolean trigger1 = false; //stores if first
boolean trigger2 = false; //stores if secon
boolean trigger3 = false; //stores if third
byte trigger1count = 0; //stores the counts
byte trigger2count = 0; //stores the counts
byte trigger3count = 0; //stores the counts
int angleChange = 0;
```

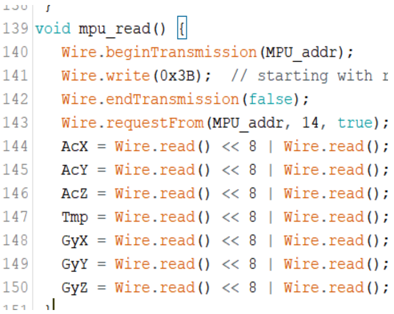


```
void sendSensor(){
  h = dht.readHumidity();
  t = dht.readTemperature(); /

  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to
    return;
    }
  Blynk.virtualWrite(V2, h);
  Blynk.virtualWrite(V1, t);
}
```

4. Begin libraries and set timer1 to 2 seconds(s) which will update t and h values after every 2s on Blynk app. Read the accelerometer and gyroscope values by mpu_read function.

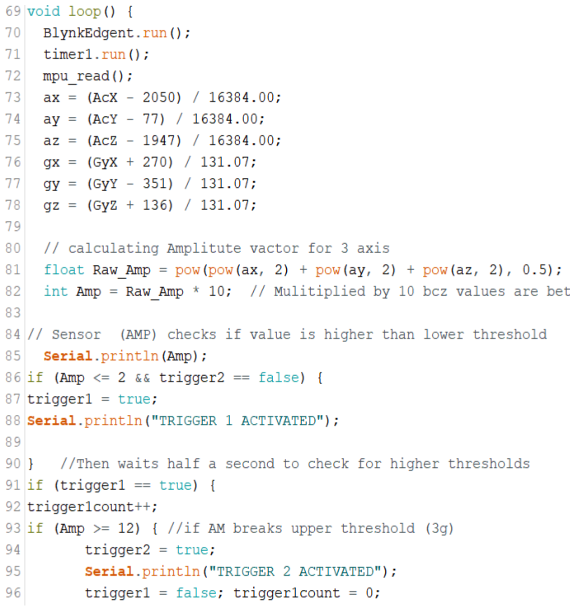


```
void setup()
{
  BlynkEdgent.begin();
  Serial.begin(115200);
  delay(100);
  dht.begin();
  timer1.setInterval(2000L, sendSensor)
  Wire.begin();
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x6B);  // PWR_MGMT_1 regi
  Wire.write(0);     // set to zero (wa
  Wire.endTransmission(true);
}
```

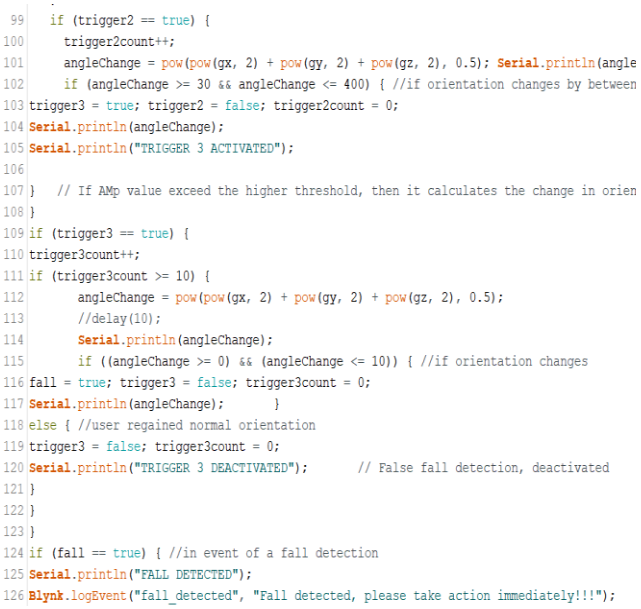


```
void mpu_read() {
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x3B);  // starting with r
  Wire.endTransmission(false);
  Wire.requestFrom(MPU_addr, 14, true);
  AcX = Wire.read() << 8 | Wire.read();
  AcY = Wire.read() << 8 | Wire.read();
  AcZ = Wire.read() << 8 | Wire.read();
  Tmp = Wire.read() << 8 | Wire.read();
  GyX = Wire.read() << 8 | Wire.read();
  GyY = Wire.read() << 8 | Wire.read();
  GyZ = Wire.read() << 8 | Wire.read();
}
```

5. Run libraries. Calculate amplitude vector (Amp) for accelerometer values. Triggers 1, 2, and 3 are activated according to certain values of Amp. When Trigger 3 is activated, change in angle is calculated for gyroscope values. If angleChange is between 0 and 10, fall is detected and sends a notification through Blynk.LogEvent function to Blynk app. Otherwise, Trigger 3 is deactivated.



```
void loop() {
  BlynkEdgent.run();
  timer1.run();
  mpu_read();
  ax = (AcX - 2050) / 16384.00;
  ay = (AcY - 77) / 16384.00;
  az = (AcZ - 1947) / 16384.00;
  gx = (GyX + 270) / 131.07;
  gy = (GyY - 351) / 131.07;
  gz = (GyZ + 136) / 131.07;

  // calculating Amplitute vactor for 3 axis
  float Raw_Amp = pow(pow(ax, 2) + pow(ay, 2) + pow(az, 2), 0.5);
  int Amp = Raw_Amp * 10;  // Mulitiplied by 10 bcz values are bet

// Sensor  (AMP) checks if value is higher than lower threshold
  Serial.println(Amp);
if (Amp <= 2 && trigger2 == false) {
trigger1 = true;
Serial.println("TRIGGER 1 ACTIVATED");

}   //Then waits half a second to check for higher thresholds
if (trigger1 == true) {
trigger1count++;
if (Amp >= 12) { //if AM breaks upper threshold (3g)
       trigger2 = true;
       Serial.println("TRIGGER 2 ACTIVATED");
       trigger1 = false; trigger1count = 0;
```



```
  if (trigger2 == true) {
    trigger2count++;
    angleChange = pow(pow(gx, 2) + pow(gy, 2) + pow(gz, 2), 0.5); Serial.println(angle
    if (angleChange >= 30 && angleChange <= 400) { //if orientation changes by between
trigger3 = true; trigger2 = false; trigger2count = 0;
Serial.println(angleChange);
Serial.println("TRIGGER 3 ACTIVATED");

}   // If AMp value exceed the higher threshold, then it calculates the change in orien
}
if (trigger3 == true) {
trigger3count++;
if (trigger3count >= 10) {
      angleChange = pow(pow(gx, 2) + pow(gy, 2) + pow(gz, 2), 0.5);
      //delay(10);
      Serial.println(angleChange);
      if ((angleChange >= 0) && (angleChange <= 10)) { //if orientation changes
fall = true; trigger3 = false; trigger3count = 0;
Serial.println(angleChange);      }
else { //user regained normal orientation
trigger3 = false; trigger3count = 0;
Serial.println("TRIGGER 3 DEACTIVATED");      // False fall detection, deactivated
}
}
}
if (fall == true) { //in event of a fall detection
Serial.println("FALL DETECTED");
Blynk.logEvent("fall_detected", "Fall detected, please take action immediately!!!");
```

Discussion

- Learned how to make an IoT system which sends data from sensors to cloud through ESP8266 NodeMCU in an Arduino IDE environment. Had a clear understanding of the network system, communication and programming.
- There were problems with the performance of the code as some of the functions were showing problems. For e.g Blynk.notify had changed to Blynk.LogEvent(). Therefore, updates in Blynk documentation https://docs.blynk.cc/ was used for help
- Initially, DHT22 sensor and MPU-6050 sensor codes were run separately which made it challenging to merge the two codes. Help from https://community.blynk.cc/ and https://stackoverflow.com/ was taken.

Cost of the project

ESP8266 NodeMCU CP2102	$17
DHT22 sensor	$13
MPU-6050 sensor	$15
Breadboard	$6
Jumper wires	$5
Micro-USB cable	$10
Total	$56

References

1. Al-Dahan, Z. T., Bachache, N. K., & Bachache, L. N. (2016). Design and Implementation of Fall Detection System Using MPU6050 Arduino. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9677*, 180–187. https://doi.org/10.1007/978-3-319-39601-9_16
2. BharathiC, S., professor, A., & Students, U. (2020). *A wearable diagnostic and therapeutic device for hyperthermic and hypothermic patients*. www.ijsdr.org
3. Brody, G. M. (1994). Hyperthermia and Hypothermia in the Elderly. *Clinics in Geriatric Medicine*, *10*(1), 213–229. https://doi.org/10.1016/S0749-0690(18)30368-9
4. *Introduction - Blynk Documentation*. (n.d.). Retrieved December 4, 2021, from https://docs.blynk.io/en/
5. *IoT Fall Detector Using MPU6050 & ESP8266 | The IOT Projects*. (n.d.). Retrieved December 4, 2021, from https://theiotprojects.com/iot-fall-detector-using-mpu6050-esp8266/
6. *Monitor Temperature & Humidity using Blynk 2.0 - circuiTician*. (n.d.). Retrieved December 4, 2021, from https://circuitician.com/monitor-temperature-humidity-using-blynk-2-0/