```
#define BLYNK_PRINT Serial
#include<ESP8266WiFi.h>
#include<BlynkSimpleEsp8266.h>
#include <Wire.h>

char auth[] = "Código do autor do projeto";

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "Nome da rede WIFI";
char pass[] = "SSID rede WIFi";

// MPU6050 Slave Device Address
const uint8_t MPU6050SlaveAddress = 0x68;

// Select SDA and SCL pins for I2C communication
const uint8_t scl = D1;
const uint8_t sda = D2;

// sensitivity scale factor respective to full scale setting provided in
datasheet
const uint16_t AccelScaleFactor = 16384;
const uint16_t GyroScaleFactor = 131;

// MPU6050 few configuration register addresses
const uint8_t MPU6050_REGISTER_SMPLRT_DIV   =  0x19;
const uint8_t MPU6050_REGISTER_USER_CTRL    =  0x6A;
const uint8_t MPU6050_REGISTER_PWR_MGMT_1   =  0x6B;
const uint8_t MPU6050_REGISTER_PWR_MGMT_2   =  0x6C;
const uint8_t MPU6050_REGISTER_CONFIG       =  0x1A;
const uint8_t MPU6050_REGISTER_GYRO_CONFIG  =  0x1B;
const uint8_t MPU6050_REGISTER_ACCEL_CONFIG =  0x1C;
const uint8_t MPU6050_REGISTER_FIFO_EN      =  0x23;
const uint8_t MPU6050_REGISTER_INT_ENABLE   =  0x38;
const uint8_t MPU6050_REGISTER_ACCEL_XOUT_H =  0x3B;
const uint8_t MPU6050_REGISTER_SIGNAL_PATH_RESET  = 0x68;

int16_t AccelX, AccelY, AccelZ, Temperature, GyroX, GyroY, GyroZ;

void setup() {
  Serial.begin(9600);
  Wire.begin(sda, scl);
  MPU6050_Init();
  Blynk.begin(auth, ssid, pass);
}

void loop() {
   double Ax, Ay, Az, T, Gx, Gy, Gz;
```

```
 Read_RawValue(MPU6050SlaveAddress, MPU6050_REGISTER_ACCEL_XOUT_H);

  //divide each with their sensitivity scale factor
  Ax = (double)AccelX/AccelScaleFactor;
  Ay = (double)AccelY/AccelScaleFactor;
  Az = (double)AccelZ/AccelScaleFactor;
  T = (double)Temperature/340+36.53; //temperature formula
  Gx = (double)GyroX/GyroScaleFactor;
  Gy = (double)GyroY/GyroScaleFactor;
  Gz = (double)GyroZ/GyroScaleFactor;

  Serial.print("Ax: "); Serial.print(Ax);
  Serial.print(" Ay: "); Serial.print(Ay);
  Serial.print(" Az: "); Serial.print(Az);
  Serial.print(" T: "); Serial.println(T);

  delay(1000);
  Blynk.run();
  Blynk.virtualWrite(V1, Ax);
  Blynk.virtualWrite(V2, Ay);
  Blynk.virtualWrite(V3, Az);
  Blynk.virtualWrite(V4, T);
}


void I2C_Write(uint8_t deviceAddress, uint8_t regAddress, uint8_t data){
 Wire.beginTransmission(deviceAddress);
  Wire.write(regAddress);
  Wire.write(data);
  Wire.endTransmission();
}

// read all 14 register
void Read_RawValue(uint8_t deviceAddress, uint8_t regAddress){
 Wire.beginTransmission(deviceAddress);
  Wire.write(regAddress);
  Wire.endTransmission();
  Wire.requestFrom(deviceAddress, (uint8_t)14);
  AccelX = (((int16_t)Wire.read()<<8) | Wire.read());
  AccelY = (((int16_t)Wire.read()<<8) | Wire.read());
  AccelZ = (((int16_t)Wire.read()<<8) | Wire.read());
  Temperature = (((int16_t)Wire.read()<<8) | Wire.read());
  GyroX = (((int16_t)Wire.read()<<8) | Wire.read());
  GyroY = (((int16_t)Wire.read()<<8) | Wire.read());
  GyroZ = (((int16_t)Wire.read()<<8) | Wire.read());
}

//configure MPU6050
void MPU6050_Init(){
```

```
 delay(150);
 I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_SMPLRT_DIV, 0x07);
 I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_PWR_MGMT_1, 0x01);
 I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_PWR_MGMT_2, 0x00);
 I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_CONFIG, 0x00);
 I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_GYRO_CONFIG, 0x00);//set +/-250
degree/second full scale
 I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_ACCEL_CONFIG, 0x00);// set +/-
2g full scale
 I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_FIFO_EN, 0x00);
 I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_INT_ENABLE, 0x01);
 I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_SIGNAL_PATH_RESET, 0x00);
 I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_USER_CTRL, 0x00);
}
```