**DEPARTMENT OF INFORMATION TECHNOLOGY FACULTY OF ENGINEERING & TECHNOLOGY**

###### **IoT PROJECT REPORT**

###### **SUBJECT TITLE : INTERNET OF THINGS**

**SUBJECT CODE: 15IT422E**

**SUBMITTED TO: Prof Kayalvizhi Jayavel**

**FALL DETECTION SYSTEM**

SARNAAM MUSTHAFA KARIPP

RA1611008010086

**Department of Information Technology**



**SRM University, SRM Nagar, Kattankulathur-603203 Kanchipuram District, Tamil Nadu**

LINKS TO GITHUB AND YOUTUBE:

YouTube:

<https://www.youtube.com/watch?v=-hb-8XGw5ZU&t=8s>

Github:

https://github.com/sarnaam/falldetection

**ACKNOWLEDGEMENT**

I would like to express my sincere gratitude to our IoT Professor Mrs Kayalvizhi Jayavel for being out there at every step of my course and guiding us all along the way to be capable of moulding our ideas into smart projects.

I would also thank Mr Hazim Hamza of Intel Corporation, Ireland, who helped me with hands-on , weeks of explanations and demos on IoT applications which helped us visualize how IoT is being used in the industry .

I would also like to thank my parents, without whom I wouldn’t be able to anything in any regard.

**ABSTRACT**

Fall detection is a major challenge in the public healthcare domain, especially for the elderly as the decline of their physical fitness, and timely and reliable surveillance is necessary to mitigate the negative effects of fall. This paper develops a novel fall detection system based on a wearable device. The system monitors the movements of human body, recognises a fall from normal daily activities by an effective quaternion algorithm, and automatically sends the request for help to the caregivers with the patient’s location.

Falling is an external aspect that can lead to death for the elderly. With so many activities they can do will increase the likelihood of falling. A fall detection device is designed to minimize post-fall risk. An MPU6050 sensor with 3 axis accelerometer and 3 gyroscope axis is used to detect the activities of the elderly. This research is expected to recognize the falling forward movement, falling aside, falling backward, sitting, sleeping, squatting, upstairs, down stairs and praying. This project module is a combined integration of NodeMCU ESP8266 and MPU6050 for gyroscope and accelerometer for retrieving the fall detection details.

**HARDWARE REQUIRED:**

* Nodemcu ESP8266 module.
* MPU6050
* Jumper Wires
* A Smartphone
* A Computer.
* Micro USB cable

**SOFTWARE REQUIRED :**

* Arduino IDE.
* ESP8266 library.
* MPU6050 Library.
* Cayenne MQTT Library.

**MISCELLANEOUS :**

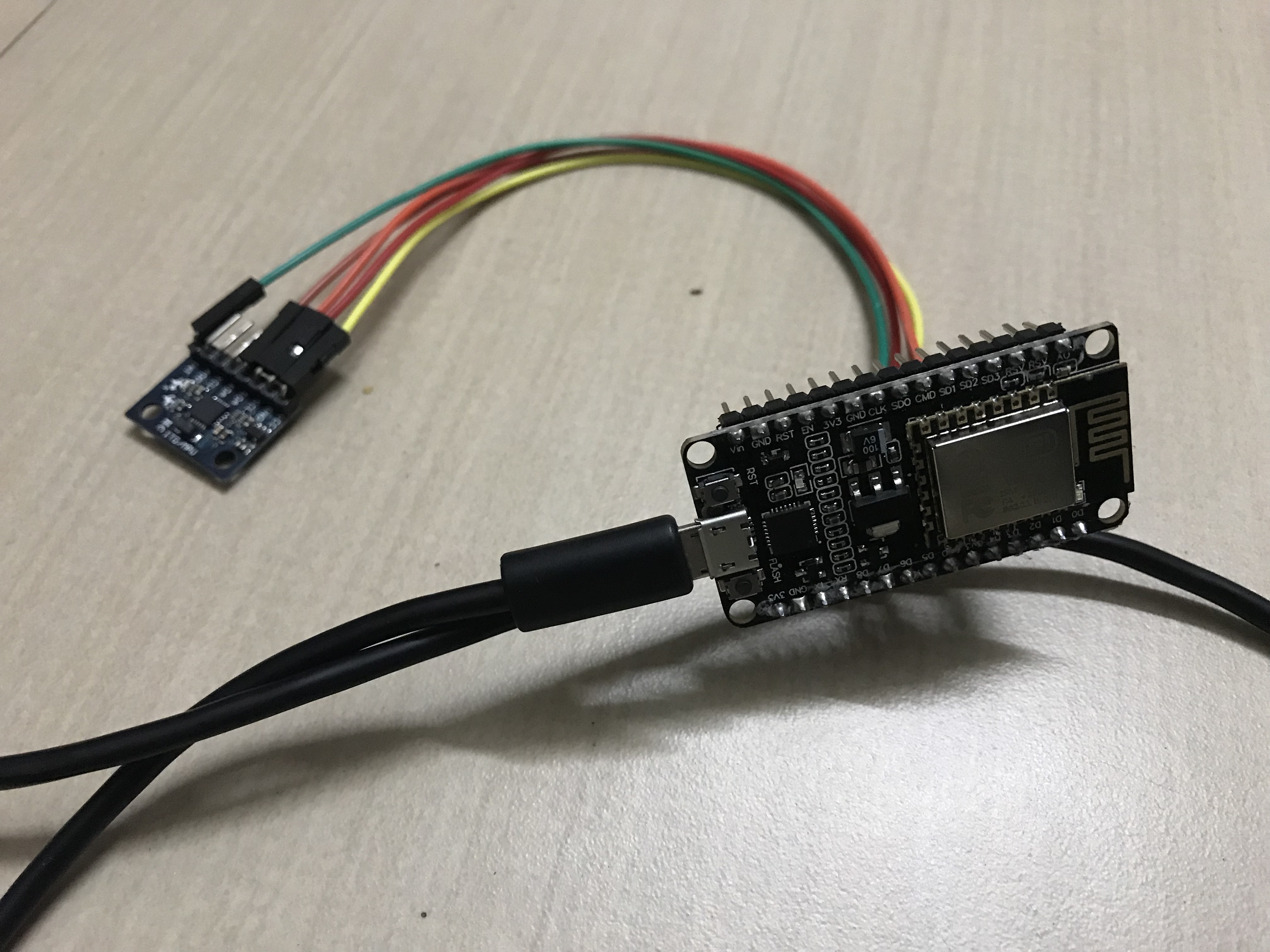
* Wifi Internet Connection
* Soldering tool
* Soldering wires

**TOTAL COST OF COMPONENTS :- RS.1000 - RS.1500.**

**SYSTEM OVERVIEW**

The main component of the setup is the Nodemcu ESP8266 module. All the other hardware components are connected to the Nodemcu. The board is programmed in Arduino IDE and uses the ESP8266,MPU6050, Cayenne MQTT libraries. These libraries have been added to the Arduino IDE.

The MPU6050 with gyroscope and accelerometer is directly connected to the NodeMCU using a micro USB cable. This MPU6050 is used to detect the fall achieved when hits the condition and is signalled by the programmed NodeMCU ESP8266 with a trigger as an Email using Cayenne Project Builder.

 Circuit for NodeMCU and MPU6050

**Programming NodeMCU and setting up Cayenne Dashboard :**

* Import MPU6050 Library to the Arduino IDE
* Fill your cayenne token, ssid, and wifi password to the code.
* Upload falldetection code to your NodeMCU
* Setup your cayenne dashboard, add a custom widget, choose two state widget and give it name, choose virtual pin V8 for connectivity, and press add widget.
* Now press the trigger, give trigger's name, on if add the widget added befor, on then fill email with your email, and press save.

**CODE**

#include <Wire.h>

#include <MPU6050.h>

#define CAYENNE\_PRINT Serial

#include <CayenneMQTTESP8266.h>

MPU6050 mpu;

boolean freefallDetected = false;

char ssid[] = "Sakht Launde-5G";

char wifiPassword[] = "hello3087";

char username[] = "f3ee8980-4e2a-11e9-9622-9b9aeccba453";

char password[] = "af47a27852e8b4a1c8081e8b9c37c1e080765ff8";

char clientID[] = "80481ae0-5529-11e9-8da4-359d3972629e";

void setup()

{

Serial.begin(115200);

Cayenne.begin(username, password, clientID, ssid, wifiPassword);

Serial.println("Initialize MPU6050");

while(!mpu.begin(MPU6050\_SCALE\_2000DPS, MPU6050\_RANGE\_16G))

{

Serial.println("Could not find a valid MPU6050 sensor, check wiring!");

delay(500);

}

mpu.setAccelPowerOnDelay(MPU6050\_DELAY\_3MS);

mpu.setIntFreeFallEnabled(true);

mpu.setIntZeroMotionEnabled(false);

mpu.setIntMotionEnabled(false);

mpu.setDHPFMode(MPU6050\_DHPF\_5HZ);

mpu.setFreeFallDetectionThreshold(17);

mpu.setFreeFallDetectionDuration(2);

checkSettings();

attachInterrupt(12, doInt, RISING);

}

void doInt()

{

freefallDetected = true;

}

void checkSettings()

{

Serial.println();

Serial.print(" \* Sleep Mode: ");

Serial.println(mpu.getSleepEnabled() ? "Enabled" : "Disabled");

Serial.print(" \* Motion Interrupt: ");

Serial.println(mpu.getIntMotionEnabled() ? "Enabled" : "Disabled");

Serial.print(" \* Zero Motion Interrupt: ");

Serial.println(mpu.getIntZeroMotionEnabled() ? "Enabled" : "Disabled");

Serial.print(" \* Free Fall Interrupt: ");

Serial.println(mpu.getIntFreeFallEnabled() ? "Enabled" : "Disabled");

Serial.print(" \* Free Fal Threshold: ");

Serial.println(mpu.getFreeFallDetectionThreshold());

Serial.print(" \* Free FallDuration: ");

Serial.println(mpu.getFreeFallDetectionDuration());

Serial.print(" \* Clock Source: ");

switch(mpu.getClockSource())

{

case MPU6050\_CLOCK\_KEEP\_RESET: Serial.println("Stops the clock and keeps the timing generator in reset"); break;

case MPU6050\_CLOCK\_EXTERNAL\_19MHZ: Serial.println("PLL with external 19.2MHz reference"); break;

case MPU6050\_CLOCK\_EXTERNAL\_32KHZ: Serial.println("PLL with external 32.768kHz reference"); break;

case MPU6050\_CLOCK\_PLL\_ZGYRO: Serial.println("PLL with Z axis gyroscope reference"); break;

case MPU6050\_CLOCK\_PLL\_YGYRO: Serial.println("PLL with Y axis gyroscope reference"); break;

case MPU6050\_CLOCK\_PLL\_XGYRO: Serial.println("PLL with X axis gyroscope reference"); break;

case MPU6050\_CLOCK\_INTERNAL\_8MHZ: Serial.println("Internal 8MHz oscillator"); break;

}

Serial.print(" \* Accelerometer: ");

switch(mpu.getRange())

{

case MPU6050\_RANGE\_16G: Serial.println("+/- 16 g"); break;

case MPU6050\_RANGE\_8G: Serial.println("+/- 8 g"); break;

case MPU6050\_RANGE\_4G: Serial.println("+/- 4 g"); break;

case MPU6050\_RANGE\_2G: Serial.println("+/- 2 g"); break;

}

Serial.print(" \* Accelerometer offsets: ");

Serial.print(mpu.getAccelOffsetX());

Serial.print(" / ");

Serial.print(mpu.getAccelOffsetY());

Serial.print(" / ");

Serial.println(mpu.getAccelOffsetZ());

Serial.print(" \* Accelerometer power delay: ");

switch(mpu.getAccelPowerOnDelay())

{

case MPU6050\_DELAY\_3MS: Serial.println("3ms"); break;

case MPU6050\_DELAY\_2MS: Serial.println("2ms"); break;

case MPU6050\_DELAY\_1MS: Serial.println("1ms"); break;

case MPU6050\_NO\_DELAY: Serial.println("0ms"); break;

}

Serial.println();

}

void loop()

{

Cayenne.loop();

}

CAYENNE\_OUT(V8)

{

Vector rawAccel = mpu.readRawAccel();

Activites act = mpu.readActivites();

if (freefallDetected)

{

Cayenne.virtualWrite(V8, act.isFreeFall); //virtual pin

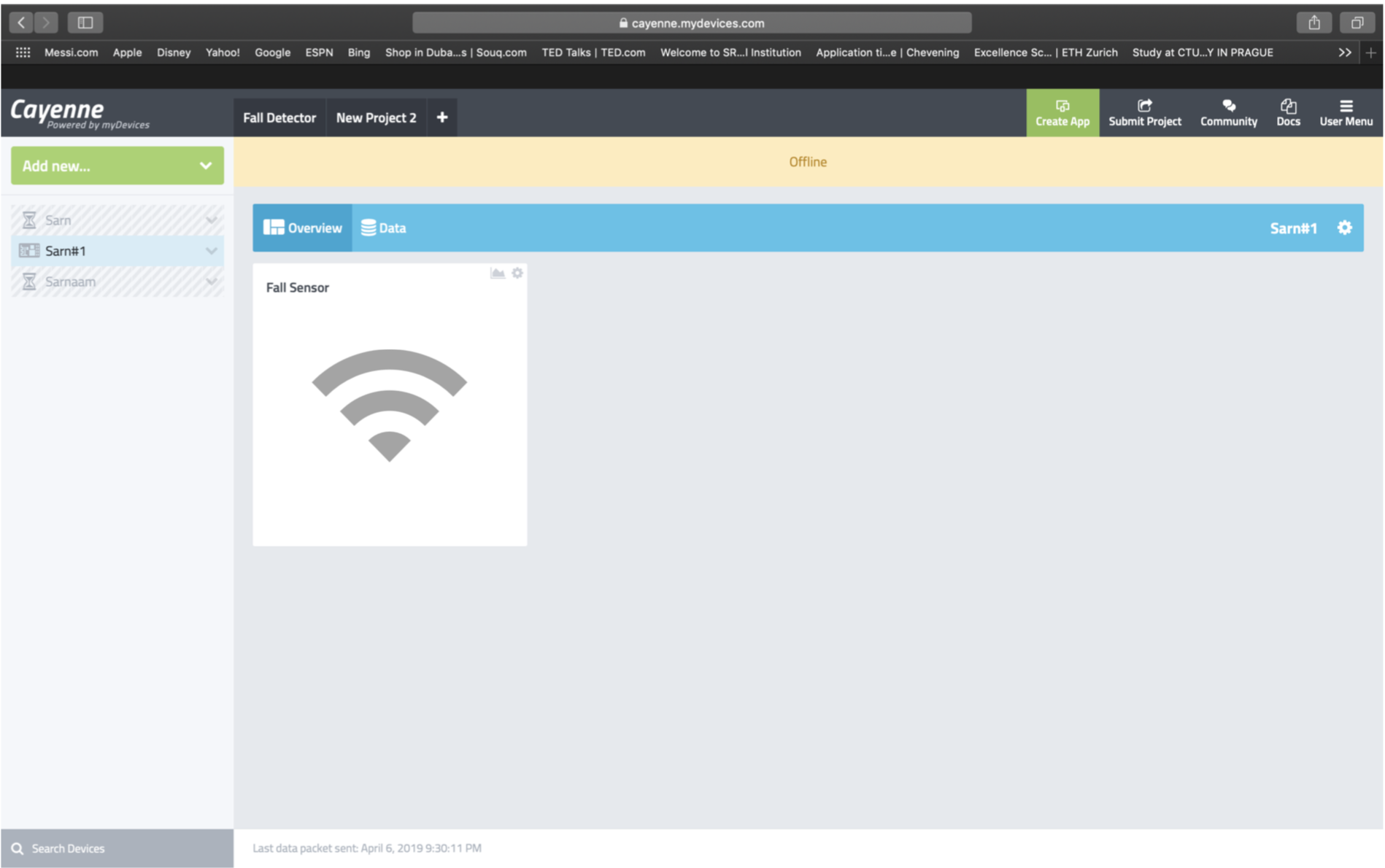
delay(100);

}

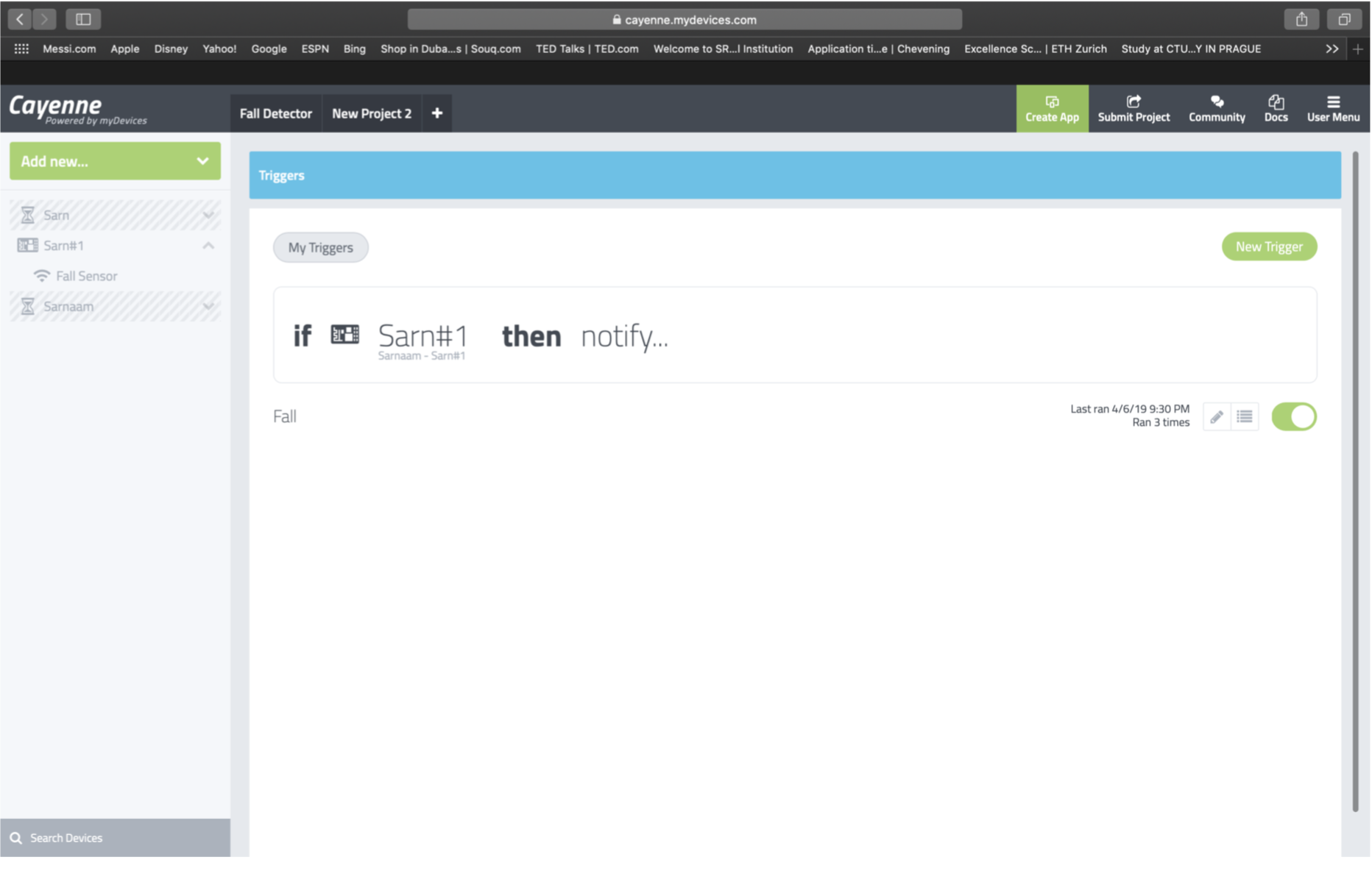
}

**Cayenne Dashboard, Widgets and Triggers :**

We give our WiFi SSID and password inside the code before compiling and uploading it to the Nodemcu. Also in order to connect to the Cayenne database, we give the Cayenne username, Cayenne password and Cayenne client ID so that it can connect via MQTT.

We then create the two state widget to make triggers in it. The trigger states that when a fall is signalled by the gyro and accelerometer module, it send a detected information to NodeMCU board connected to the Cayenne network which then provides a Triggered notification to the mail given on the trigger.

**Result :**

Fall detection system using ESP8266 NodeMCU, MPU6050 for gyroscope and accelerometer module has been successfully developed and implemented.

