Final Project: STM32 Lifesaver

Group Member:

Poomchai Taechaprapasrat 5831055821

Burin Amornpaisannon 5831035221

Krittakan Tisantiea 5830009821

Supawit Kunopagarnwong 5831073021

This project is falling detector. When user has fallen down, it will alarm to other people by sound and sending an alarm signal to the caretaker.

1. Wear it.
2. It will detect acceleration of itself to know whether the use, who is wearing it, is falling down or not.
3. If the user fell down, it will make sound by a speaker and send alarm signal to the server.
4. Caretaker will know that user fell down by watching status on the server and people near fallen user will know by the alarm from the speaker.

Sensors: Accelerometer and Speaker

Github: https://github.com/Poomchaio/STM32-LIFESAVER

Embedded System Development

Burin Amornpaisannon 5831035221

STM32 Lifesaver is a falling detector for detecting whether user, an elderly person, has fallen down or not. This is one of many IOT devices for preparing us to cope with Aging Society, which is the problem that Thailand will certainly face in the future.

For my position, embedded system development, I have to make an embedded system from microcontrollers and sensors that meets our specification, to detect whether user has fallen on a floor or not.

For this project, STM32 Lifesaver or our falling detector, when the user using it is falling down, what we can detect is his/her acceleration. The acceleration will be increasing nearly the gravitational acceleration and will be decreasing immediately when the user has fallen to a floor.

Hence, I chose, of course, the accelerometer inside STM32 to detect user's acceleration. Not only the accelerometer but also a speaker to alarm to other people near user when user has fallen down and also send NodeMCU ESP8266 to alarm caregiver too (but another person in my group implemented the program inside ESP8266, my position covers only STM32 board).

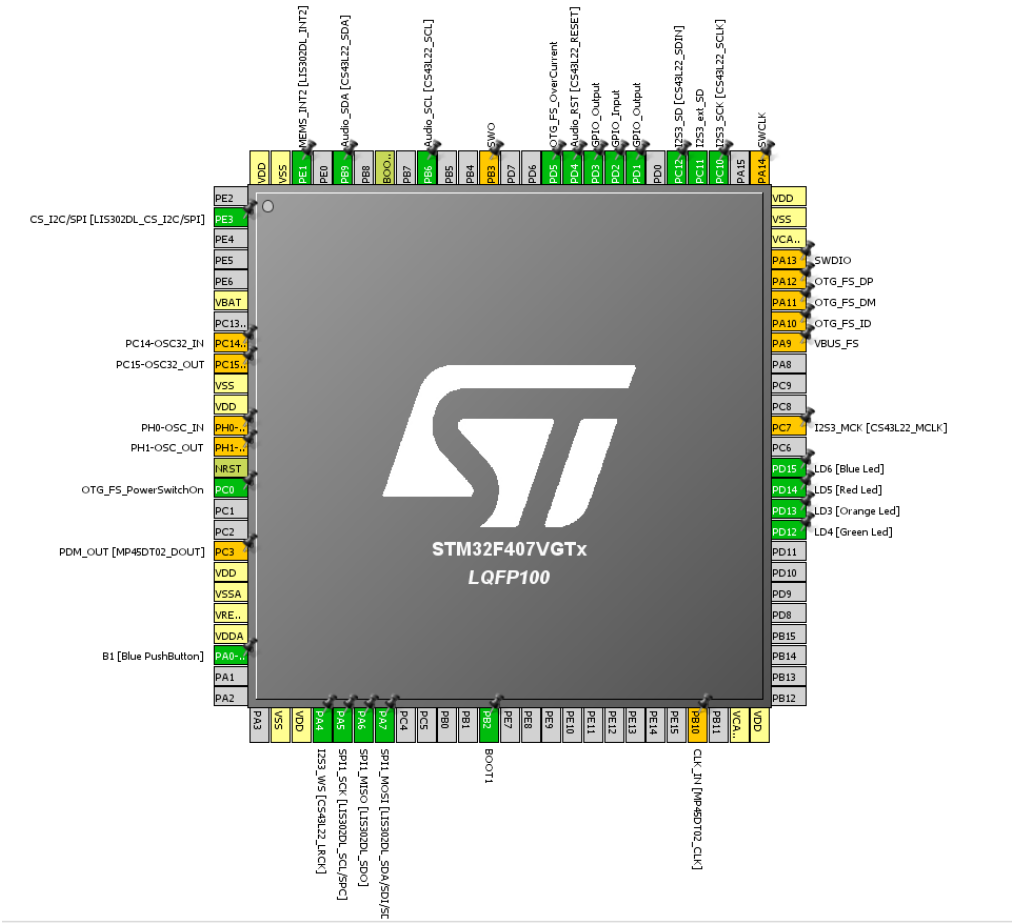
Embedded System Development Implementation

We used 2 sensors which are an accelerometer inside our microcontroller board, STM32F407, to detect whether user is falling or not and a speaker to alarm to other people near user when user has fallen down.

In addition, we used STM32F407 connected with NodeMCU ESP8266 because ESP8266 can connect to the server and send alarm to the caregiver.

In conclusion, the STM32 Lifesaver has a speaker to alarm to other people near the user and ESP8266 to alarm the caregiver wherever the caregiver is by sending a notification on the caregiver's phone.

The picture below is the pins we used.



I used D1 pin connected with the speaker for making a sound, D14 pin for the red LED on the board, D2 and D3 pin to send ok signal and fallen signal respectively to NodeMCU and, SPI1 and I2S3 for using accelerometer.

What I implemented

First I initialized all variables, inside while loop every loop it will call functions, callvalue(), callaxis() and callangle() (callangle() is not be used, it is for angle calculation)

```
void calvalue(){
    BSP_ACCELERO_GetXYZ(xyz);

    gravity[0] = alpha * gravity[0] + (1 - alpha) * xyz[0];
    gravity[1] = alpha * gravity[1] + (1 - alpha) * xyz[1];
    gravity[2] = alpha * gravity[2] + (1 - alpha) * xyz[2];

    output[0] = xyz[0] - gravity[0];
    output[1] = xyz[1] - gravity[1];
    output[2] = xyz[2] - gravity[2];

    if(output[0] < 0) x = -1*output[0];
    else x = output[0];
    if(output[1] < 0) y = -1*output[1];
    else y = output[1];
    if(output[2] < 0) z = -1*output[2];
    else z = output[2];

}
```

Firstly, callvalue() call BSP_ACCELERO_GetXYZ(xyz[3]) for getting raw X, Y and Z accelerations from accelerometer. Unfortunately, these accelerations contain gravitational acceleration, so it is not accurate to get how much acceleration user is really getting. I solved this problem by using high-pass filter which is a method to extract gravitational acceleration from raw data. Therefore I can get X, Y and Z gravitational accelerations by the formula above and also can get accelerations, without gravitational acceleration, from the formula above.

```
void calaxis(){
    int max = -9999;
    for(int i = 0 ; i < 3 ; i++){
        if(gravity[i] > max){
            max = gravity[i];
            axis = i;
        }
    }
}
```

Another function, callaxis(), is for calculating which axis, X, Y or Z, is the vertical axis. This function will check whether an acceleration it detects is in vertical axis or not (if not then that acceleration will not be created by falling).

```
if(axis == 2 && z > lthreshold && z < rthreshold){
    while(1){
        calvalue();
        calaxis();
        if(axis == 2 && z < lthreshold) {
            fall();
            break;
        }
        else if(z > rthreshold) break;
    }
}
```

After the microcontroller has already got X, Y and Z accelerations and the vertical axis, the last step is to analyze whether the accelerations are created by falling or not. The microcontroller analyzes by taking care only the axis that is the vertical axis and check the acceleration on that axis should be more that a minimum value (lthreshold) and should be less than a maximum value (rthreshold) and this peak of this acceleration should not be too high.

Web Developer and NodeMCU

Supawit Kunopagarnwong 5831073021

ทำตัว nodeMCU โดยรับสัญญาณมาจากตัว STM32 แล้วต่อ WiFi โดย WiFiClient ของ nodeMCU เสร็จแล้วเมื่อสัญญาณที่ส่งมาจาก STM32 ส่งมา

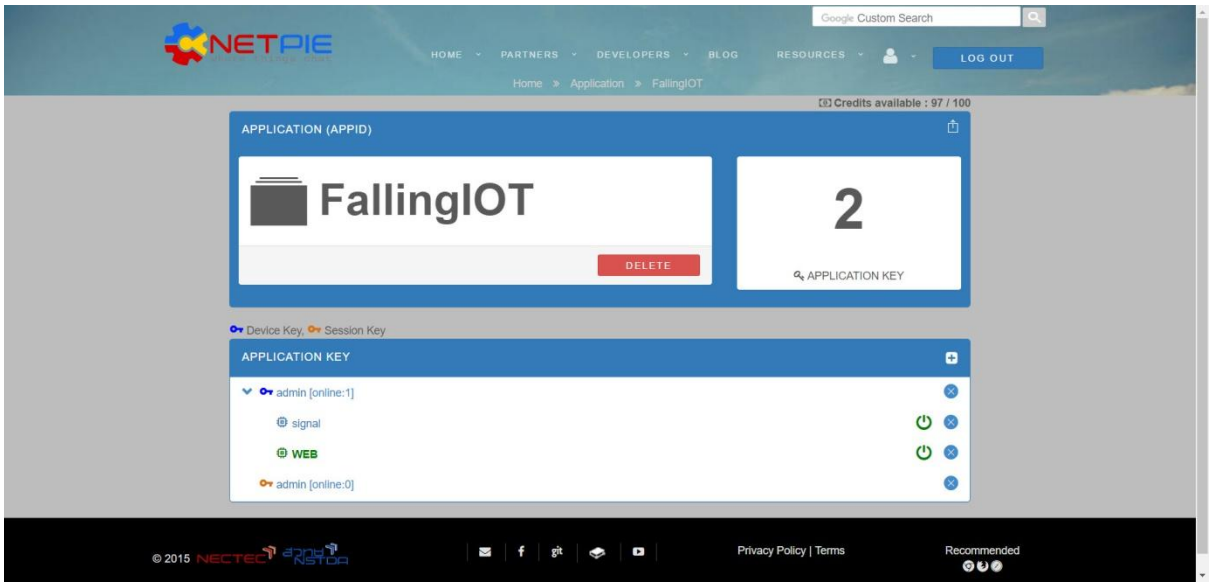
บอกว่าล้ม แล้วตัว nodeMCU จะส่งสัญญาณและค่าไปทั้ง 2 ที่ คือ

nodeJS

server ของตัว Application โดยการใช้ HTTPClient หรือที่เรียกว่าการ POST request ไปที่ server

NETPIE

server ไว้ส่งต่อไปที่ Websiteโดยใช้ Microgear chat ระหว่างตัว nodeMCU กับ Website ผ่านทาง Netpie โดยเราต่อกับ Netpie โดยใช้ APPID , KEY , SECRET ของทาง Netpie ทั้งสิ้น



โดยเราจะเข้าเงื่อนไขนี้ก็ต่อเมื่อมีสัญญาณจาก STM32 ว่ามีการล้ม หรือ ล้มแต่ปลอดภัย ประกอบกับการใช้ตัว state ในการให้มัน check เพิ่งรอบเดียวคล้าย single pulser ป้องกันการกดซ้ำๆ โดย

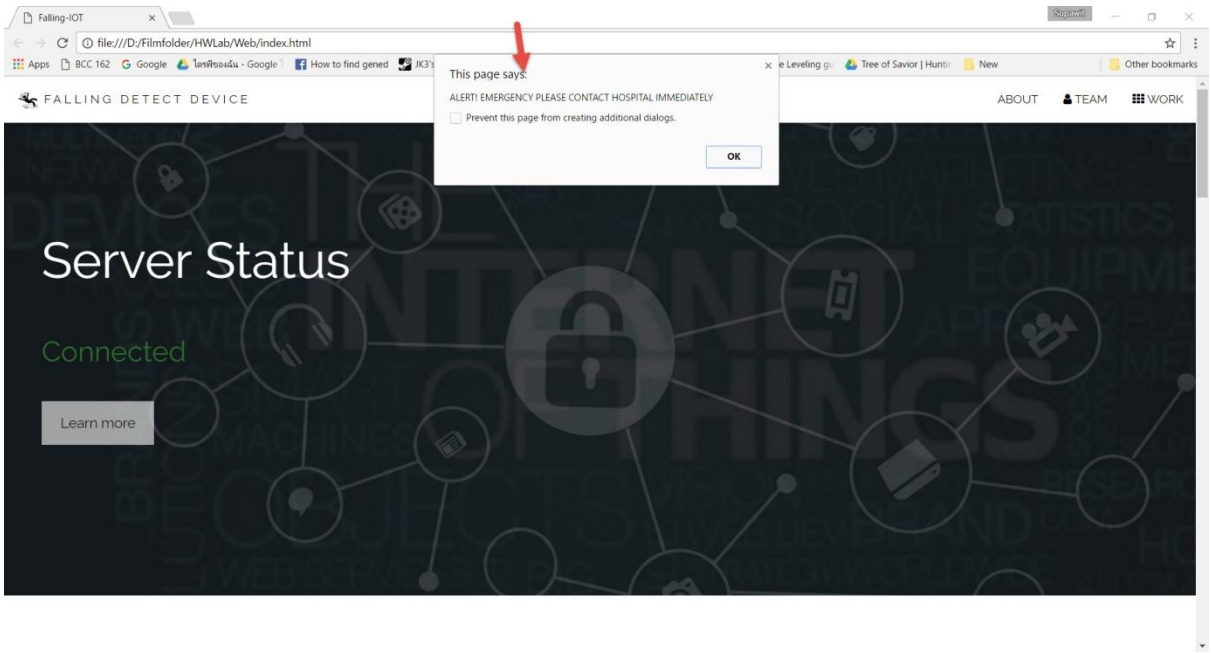
state 0 คือ ปกติไม่มีอะไรเกิดขึ้นทั้งสิ้น

state 1 คือ เกิดอาการล้มเกิดขึ้น

state 2 คือ ล้มแล้วแต่ปลอดภัยหายห่วง

ทำ Website โดยใช้ Boostrap กับ JavaScript เป็นหลักใช้การสื่อสารกับ MCU ผ่าน Netpie โดย Microgear chat

เว็บไซต์ main page มี status บอกว่า connect กับ netpie อยู่หรือไม่



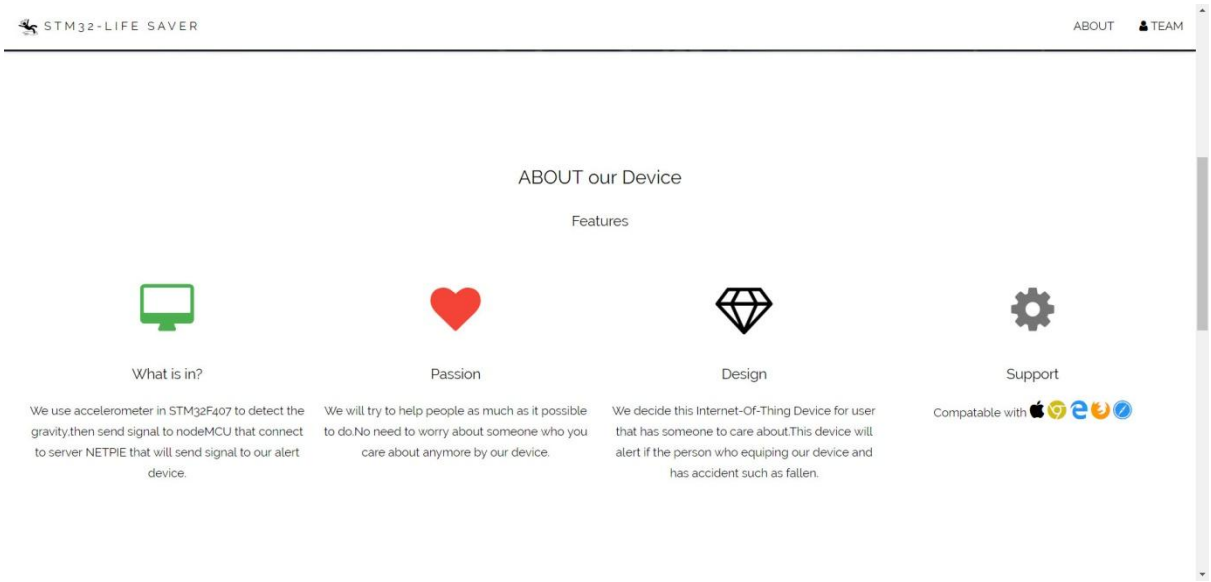
main page เมื่อมีสัญญาณการล้มถูกส่งมาจะมี alert และ เสียงแจ้งเตือน ประกอบกับ server status เปลี่ยนเป็น human fallen ดังภาพข้างล่าง

main page แสดงการแจ้งเตือนเมื่อผู้ใช้กด OK แล้ว ก็ควรรีบติดต่อคนที่บ้านหรือรถพยาบาลเป็นต้น

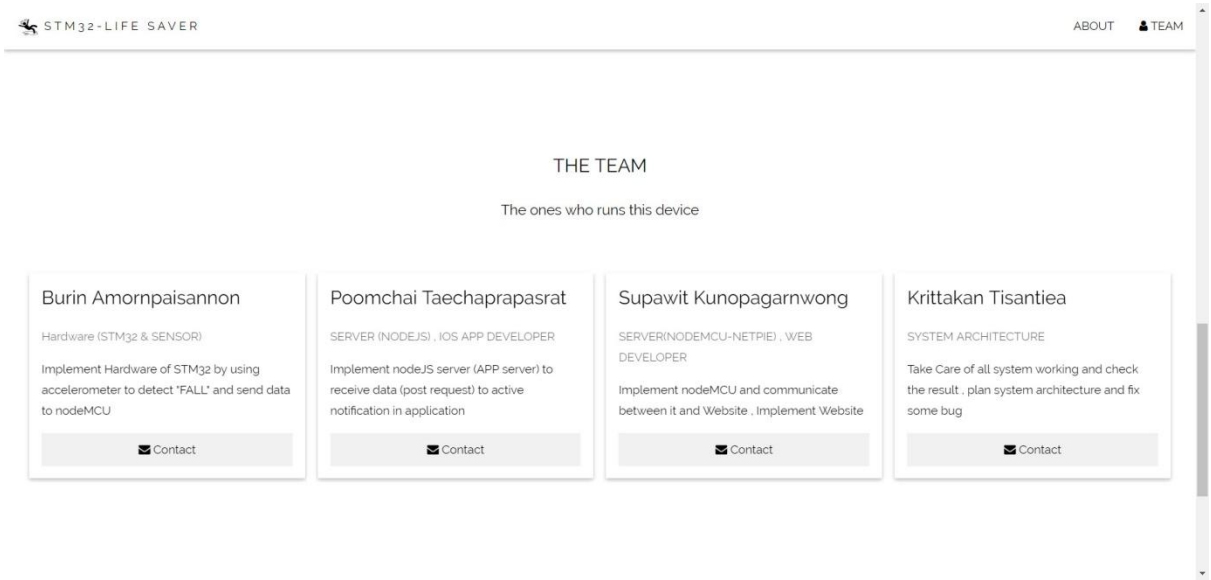
เมื่อคนที่ล้มลง กดปุ่มเพื่อส่งสัญญาณว่าไม่เป็นไรจะขึ้นข้อความ I’m Fine

Website Interfaces

About US



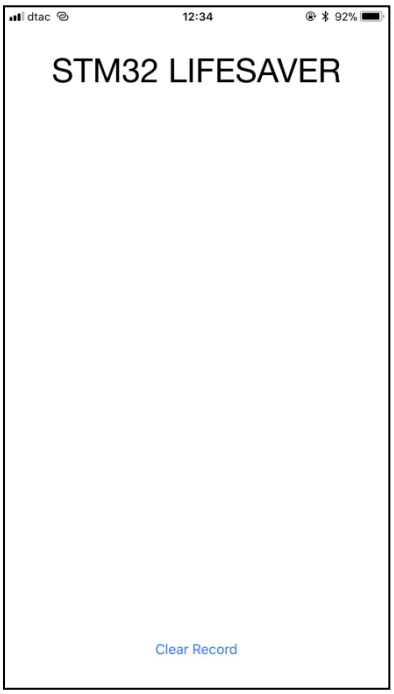
OUR TEAM



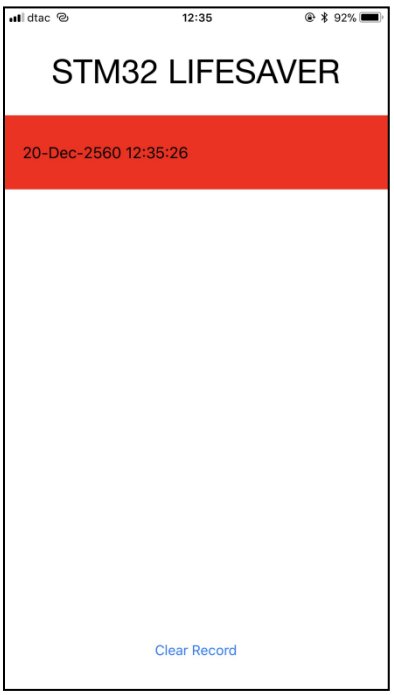
Application Developer

Poomchai Taechapraparat 5831055821

1. หน้านี้จะแสดงการล้มทั้งหมดที่เกิดขึ้น พร้อมทั้งมีปุ่มที่ใช้เคลียร์การล้มทั้งหมดที่เกิดขึ้นได้



2. เมื่อบอร์ดมีการส่งสัญญาณว่ามีการล้มเกิดขึ้น จะมีเวลาที่บอร์ดส่งสัญญาณปรากฏพร้อมทั้งมีพื้นหลังเป็นสีแดง



3. เมื่อการล้มที่เกิดขึ้นนั้นผู้สวมใส่ไม่ได้รับบาดเจ็บจากการล้มครั้งนั้นสามารถกดปุ่มที่บอร์ดเพื่อส่งสัญญาณกลับมาและเปลี่ยนพื้นหลังเป็นสีเขียวเพื่อบอกว่าผู้สวมใส่ปลอดภัยดี

ในส่วนของการรับสัญญาณจากบอร์ดไปสู่ผู้ใช้มีการใช้งาน

Node.js

ใช้ในการสร้างเซิฟเวิอร์จำลองขึ้นมาเพื่อใช้ในการรับ - ส่งrequest ไปยัง Pusher

express

ใช้ในการสร้าง post request

Pusher

ใช้รอ request จาก Node.js จากนั้นจึงส่ง request ไป application

NodeMCU and STM32 Debugger

Krittakan Tisantiea 5830009821

ช่วยตรวจสอบผลการทำงานของการรับค่าของ Accelerometer และออกความเห็นเผื่อช่วยแก้ไขให้ Accelerometer ตรวจจับการล้มได้ถูกต้องมากขึ้น

ช่วยตรวจสอบผลการส่งรับค่าจาก board stm32 ไปยัง nodeMCU และช่วยแก้ไข bug การรับค่าที่ผิดพลาดจาก board stm32 ไปยัง nodeMCU