Internet de las Cosas

Taller 2
“Proyecto ESP”

Estudiantes:

- Matias Arias.
- Roberto Illesca.
- Javier Ortega.
- Daniel Ruiz.
- Claudio Sáez.

Índice

Idea de proyecto

El proyecto consiste en desarrollar un sistema de monitoreo para personas de tercera edad o movilidad reducida, con el objetivo de reducir significativamente los tiempos de respuesta en situaciones de emergencia que pongan en riesgo su integridad física.

Este sistema estará basado en un dispositivo portátil que contará con un sensor especializado capaz de detectar caídas u otros eventos que puedan poner en riesgo la integridad de la persona. Al ocurrir una caída, el dispositivo enviará una alerta inmediata.

Además, a futuro, el sistema podrá integrarse con otros dispositivos, como buzzer para avisar a través de sonido al momento de la caída.

Detalle de sensores y actuadores utilizados

Sensores

Acelerómetro de 3 ejes (SEN0178 Triple Axis Accelerometer)

Precio	3,99 USD
Interfaz de datos	Salida analógica.
Interfaz de energía	Alimentación a partir del rango de 3.3V a 8V.
Sensores alternativos	• ADXL345 • SKU:DFR0134

Pulsador

Precio	$750 (10 x $7.500)
Interfaz de datos	Entrada digital y GND (a tierra), conectado al Arduino.
Interfaz de energía	Alimentación a partir de bajo voltaje (5V), proveniente del circuito al que está integrado.
Sensores alternativos	• Cruceta

Datos Enviados

El sistema de monitoreo utiliza un sensor de aceleración registra datos en los ejes X, Y, y Z, proporcionando información sobre los movimientos. Estos datos permiten detectar caídas. A continuación, se describen los ejes:

Dato	Descripción	Unidad	Ejemplo
X	Aceleración horizontal	g / m/s²	Detección de inclinación lateral, giros horizontales
Y	Aceleración vertical	g / m/s²	Detección de caídas, altura de saltos
Z	Aceleración en profundidad	g / m/s²	Detección de movimiento hacia adelante o atrás
Botón	Estado del botón (presionado o no presionado)	0/1	Detección de la pulsación del botón de confirmación de caída.

Protocolo utilizado

TLS, Smart healthcare

Ejemplo de Api utilizada:

```
#define BLYNK_TEMPLATE_ID "TMPL2Dpc1MpJb"
#define BLYNK_TEMPLATE_NAME "Quickstart Template"
#define BLYNK_AUTH_TOKEN "74ay2OfFVVwx4l8ZEym6yyPVut0Qeo6H"

#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <math.h>
#include <stdio.h>

// WiFi credentials
char ssid[] = "SSID RED WIFI";
char pass[] = "CONTRASEÑA";

BlynkTimer timer;

int val_x = 0, val_y = 0, val_z = 0;
double b;
const int portX = 34;
const int portZ = 35;
const int portY = 32;

const int portButton = 23;
int buttonState = 0;

// Esta función se ejecuta periódicamente para enviar datos a Blynk
void sendAccelDataToBlynk() {
  // Inicializar valores a cero antes de la suma
  val_x = 0;
  val_y = 0;
  val_z = 0;

  // Acumular valores analógicos
  for (int i = 0; i < 10; i++) {
    val_x += analogRead(portX);
    val_y += analogRead(portY);
    val_z += analogRead(portZ);
  }

  // Calcular el promedio de las lecturas
  val_x = val_x / 10;
  val_y = val_y / 10;
  val_z = val_z / 10;

  // Cálculo del ángulo
  b = (double)(abs(val_x - 320)) / (abs(val_z - 320));
  float a = atan(b) / 3.14 * 180;

  Serial.print("X: ");
  Serial.print(val_x);
  Serial.print(" Y: ");
  Serial.print(val_y);
  Serial.print(" Z: ");
  Serial.println(val_z);
  Serial.print(" Botón: ");
  Serial.println(buttonState);

  // Enviar los datos a Blynk en pines virtuales
  Blynk.virtualWrite(V0, val_x);
  Blynk.virtualWrite(V1, val_y);
  Blynk.virtualWrite(V2, val_z);
  if (val_y > 2000) {
    Blynk.virtualWrite(V3, buttonState == LOW ? 1 : 0);
  }
}

void setup() {
  Serial.begin(115200);
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
  pinMode(portButton, INPUT_PULLUP);

  // Llamar a la función sendAccelDataToBlynk cada segundo
  timer.setInterval(1000L, sendAccelDataToBlynk);
}

void loop() {
  Blynk.run();
  timer.run();
  buttonState = digitalRead(portButton);
}
```

Plataforma Utilizada

Blynk ofrece una plataforma de internet de las cosas (IoT) de marca blanca que ofrece software, firmware, soluciones web y aplicaciones móviles a miles de pequeñas, medianas y grandes empresas de todo el mundo.

Esta es la aplicación más popular para conectar dispositivos a la nube, diseñar aplicaciones para controlarlos y supervisarlos de forma remota, y administrar miles de productos implementados. El software Blynk ayuda a individuos y organizaciones a avanzar fluidamente desde un prototipo de un producto conectado hasta su lanzamiento comercial.

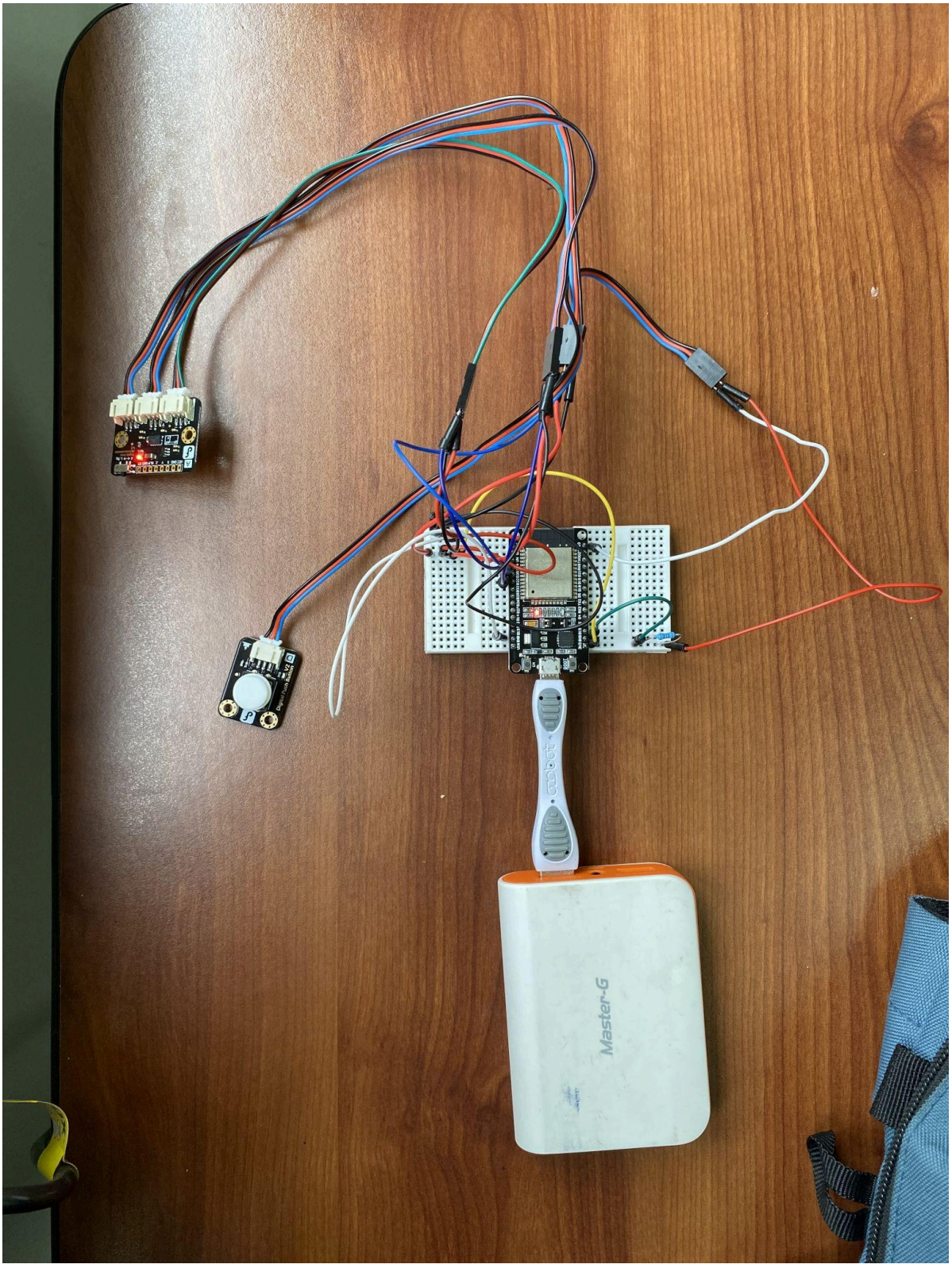
Tipo de licencias disponibles

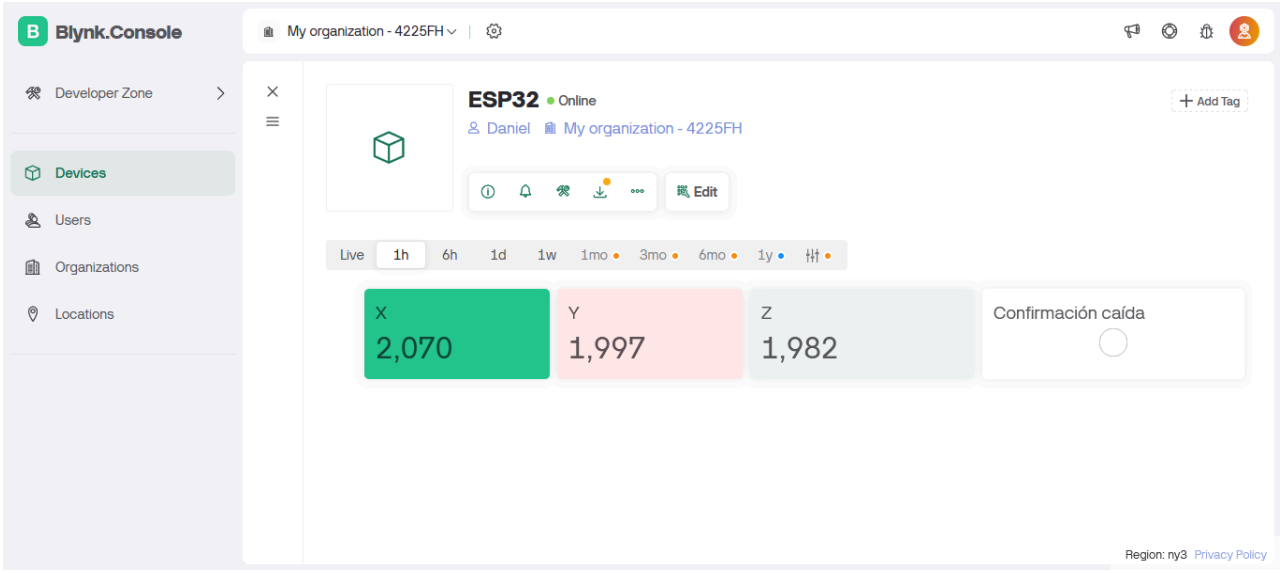
- Licencia gratuita.
- Licencia Maker.
- Licencia Pro.
- Licencia Enterprise.

Protocolos utilizados

HTTP

Evidencia de funcionamiento del prototipo




Blynk.Console
My organization - 4225FH
Developer Zone
Devices
Users
Organizations
Locations
ESP32
Online
Daniel
My organization - 4225FH
Add Tag
Edit
Live
1h
6h
1d
1w
1mo
3mo
6mo
1y
X
2,070
Y
1,997
Z
1,982
Confirmación caída
Region: ny3
Privacy Policy

Secciones de código importantes

```
// Esta función se ejecuta periódicamente para enviar datos a Blynk
void sendAccelDataToBlynk() {
  // Inicializar valores a cero antes de la suma
  val_x = 0;
  val_y = 0;
  val_z = 0;

  // Acumular valores analógicos
  for (int i = 0; i < 10; i++) {
    val_x += analogRead(portX);
    val_y += analogRead(portY);
    val_z += analogRead(portZ);
  }

  // Calcular el promedio de las lecturas
  val_x = val_x / 10;
  val_y = val_y / 10;
  val_z = val_z / 10;

  // Cálculo del ángulo
  b = (double)(abs(val_x - 320)) / (abs(val_z - 320));
  float a = atan(b) / 3.14 * 180;

  Serial.print("X: ");
  Serial.print(val_x);
  Serial.print(" Y: ");
  Serial.print(val_y);
  Serial.print(" Z: ");
  Serial.println(val_z);
  Serial.print(" Botón: ");
  Serial.println(buttonState);

  // Enviar los datos a Blynk en pines virtuales
  Blynk.virtualWrite(V0, val_x);
  Blynk.virtualWrite(V1, val_y);
  Blynk.virtualWrite(V2, val_z);
  if (val_y > 2000) {
    Blynk.virtualWrite(V3, buttonState == LOW ? 1 : 0);
  }
}
```

Anexo

Código completo del Arduino:
https://github.com/DanRuizK57/Proyecto-ESP

Referencias

- https://docs.blynk.io/en