upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

TENSIONAPP:
SISTEMA DE DETECCIÓN DE LA FRECUENCIA CARDIACA MEDIANTE EL SENSOR FOTOGRÁFICO DE UN SMARTPHONE Y MEDIANTE UN MICROCONTROLADOR.

Autor: Miguel Fuertes Fernández
Tutor: Javier Rodríguez Falces (Universidad Pública de Navarra)
Director: Javier Rodríguez Falces (Universidad Pública de Navarra)

CONTENIDOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

1. Introducción y contexto
2. Objetivos
3. Requisitos
4. Implementación
    1. Arquitectura y modelo de datos
    2. Cliente y servidor
5. Mediciones
    1. Introducción
    2. Smartphone (Cámara + flash)
    3. Microcontrolador
6. Resultados
7. Conclusiones y líneas futuras

INTRODUCCIÓN Y CONTEXTO

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

1. Articulo y trabajo de Ainara Garde:
   - *"… desarrolló un instrumento que permite a los trabajadores de salud de primera línea detectar rápidamente la necesidad de los niños de ser hospitalizados…"*
   - *"Para medir este factor de riesgo, el proyecto utilizó un sensor de dedo, el oxímetro de teléfono. "*

2. Experiencia y necesidad personal:
   - Soy hipertenso y ninguna de las aplicaciones en el mercado reunía y controlaba de manera eficiente todo lo que necesitaba medir:
     - Control de Peso
     - Control de Presión Sanguínea

https://www.diariodenavarra.es/noticias/navarra/2017/11/27/la-investigadora-navarra-ainaragarde-premio-internacional-medicina-infantil-564133-300.html

CONTENIDOS

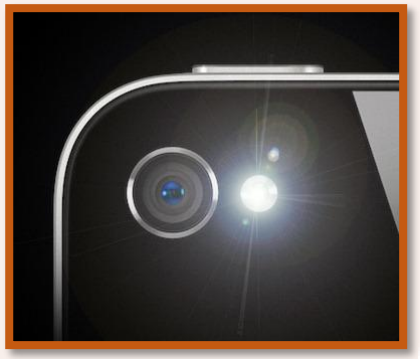
upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

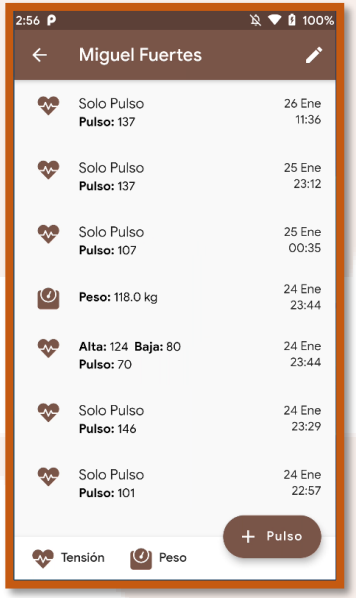
1. Introducción y contexto
2. Objetivos
3. Requisitos
4. Implementación
    1. Arquitectura y modelo de datos
    2. Cliente y servidor
5. Mediciones
    1. Introducción
    2. Smartphone (Cámara + flash)
    3. Microcontrolador
6. Resultados
7. Conclusiones y líneas futuras

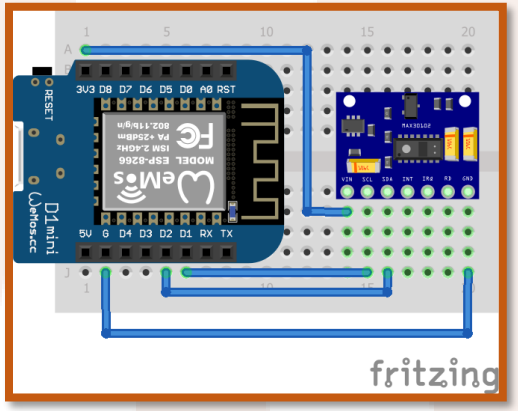
OBJETIVOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

- **Medición del pulso mediante la cámara y el flash del smartphone**
  - Ampliamente disponible hoy en día.
  - Replica el mecanismo del pulsioxímetro tradicional, al captar la variación de intensidad de color en un dedo, mediante la cámara y el flash del smartphone.
- **Medición del pulso mediante un microcontrolador y un sensor**
  - Así mismo, los smartphones, a la vez cuenta con tecnología USB-OTG, que permite conectar periféricos al aparato a través del puerto USB.
  - Se ha desarrollado un pequeño programa en un microcontrolador unido a un sensor más fiable, para realizar las mediciones en la aplicación.
- **Sistema (cliente-servidor) de gestión de medidas e historia médica**:
  - Por último y para unir todo esto, se ha desarrollado un sistema de gestión de medidas por médico y paciente para el control en el tiempo, a la vez de poder realizar un seguimiento.
  - Esta información siempre podrá ser exportada anónimamente para cualquier investigación en un paquete estadístico externo.







CONTENIDOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

1. Introducción y contexto
2. Objetivos
3. Requisitos
4. Implementación
    1. Arquitectura y modelo de datos
    2. Cliente y servidor
5. Mediciones
    1. Introducción
    2. Smartphone (Cámara + flash)
    3. Microcontrolador
6. Resultados
7. Conclusiones y líneas futuras

REQUISITOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Los requisitos (o casos de uso, en su forma más generalizada) es la forma que se emplea en el desarrollo de software para ver que necesidades sugiere el cliente y exponerlos de manera clara en una tabla. Los requisitos recogidos para este proyecto se exponen a continuación:

- Como **médico**, quiero poder registrar la edad y el género de un paciente
- Para un paciente, quiero poder introducir variables físicas observadas durante la exploración de dicho paciente, esto es además de la presión sanguínea, el pulso o el peso, los siguientes indicadores: "*Enfermedad Renal Crónica*", "*Asma*", "*Enfermedad Pulmonar Obstructiva Crónica*", "*Diabetes*", "*Dislipemia*", "*Cardiopatía Isquémica*", "*Insuficiencia cardiaca previa*".
- Para un **paciente**, quiero poder realizar una medida de pulso (frecuencia cardiaca) utilizando la cámara y el flash del smartphone.
- Para un **paciente**, quiero poder realizar una medida de pulso (frecuencia cardiaca) utilizando un pequeño sensor conectado por USB (Arduino).
- Como **médico**, quiero poder guardar los datos de esta exploración en un servidor para alimentar un modelo predictivo.
- Como **médico**, quiero poder hacer uso de ese modelo predictivo para mostrar una valoración automática por pantalla.
- Como **médico**, quiero poder exportar los datos de la base de datos, anonimizados para el posterior análisis en un ordenador.
- Como **médico**, quiero poder acceder al mismo sistema (base de datos) desde cualquier smartphone utilizando una cuenta personal (email y contraseña).
- Como **médico**, me gustaría poder utilizar la huella y demás sensores biométricos para utilizar la aplicación, sin depender de meter la contraseña cada vez.

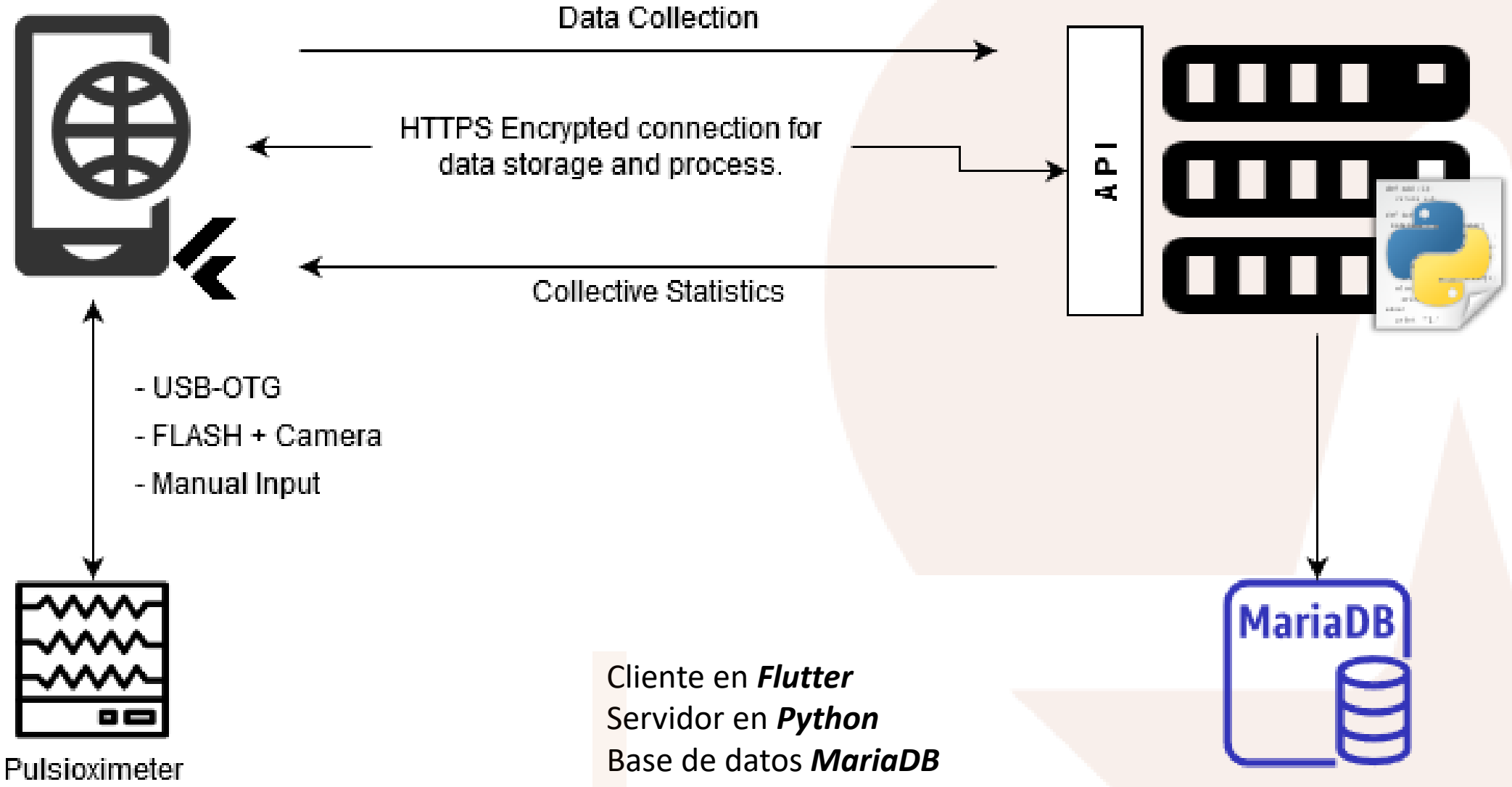
CONTENIDOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

1. Introducción y contexto
2. Objetivos
3. Requisitos
4. Implementación
    1. Arquitectura y modelo de datos
    2. Cliente y servidor
5. Mediciones
    1. Introducción
    2. Smartphone (Cámara + flash)
    3. Microcontrolador
6. Resultados
7. Conclusiones y líneas futuras

IMPLEMENTACIÓN | ARQUITECTURA

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa



Cliente en ***Flutter***
Servidor en ***Python***
Base de datos ***MariaDB***
Conexión mediante API

IMPLEMENTACIÓN | MODELO DE DATOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

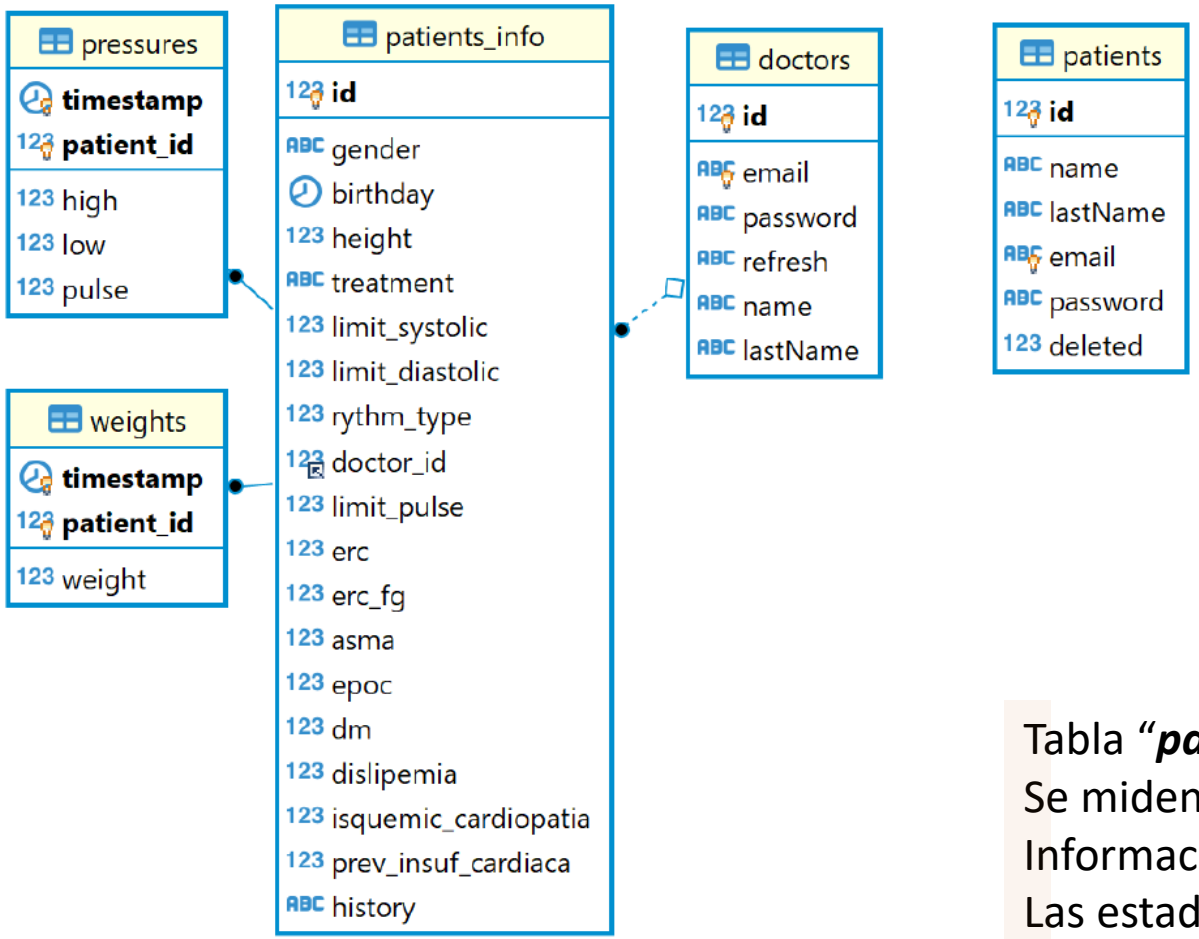


Tabla "***patients***" separada, permite *auth* externa (futuro).
Se miden igual las "***pressures***" que los "***pulses***".
Información estática en la tabla de paciente.
Las estadísticas se calculan "*on-the-fly*"

IMPLEMENTACIÓN | CLIENTE

- La tecnología utilizada para crear el cliente (la app) es ***Flutter***.

  Flutter *es una tecnología desarrollada por Google, programada en Dart, cuya peculiaridad es que compila el lenguaje Dart al lenguaje nativo de ambas plataformas, Java/Kotlin en Android y Objective C/Swift en iOS*.

  Con un solo código podemos generar ambas aplicaciones. Para la mayoría de las acciones es suficiente, pero si queremos utilizar la cámara para medir el pulso, o comunicarnos con un dispositivo mediante el puerto USB, tenemos que desarrollar esa característica especifica en el lenguaje nativo de cada plataforma (actualmente solo desarrollado en Android).

- Tecnologías descartadas:
  - Cordova/Phonegap, PWA, React Native, programacion directa en Swift/Objective C y Android Java/Kotlin

IMPLEMENTACIÓN | SERVIDOR

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

- La tecnología utilizada para crear el servidor donde se almacenaran los datos es ***Python***, por su gran cantidad de módulos que facilitan la interconexión del programa con casi todos los sistemas de procesado de estadísticas.

- Encima de Python, el framework utilizado es ***Flask***, ya que provee una manera rápida y potente de implementar una API.

- La comunicación se hace mediante ***HTTPS***, y la autenticación se hace mediante ***JWT***, que garantiza que los tokens sean auténticos para devolver la información requerida.

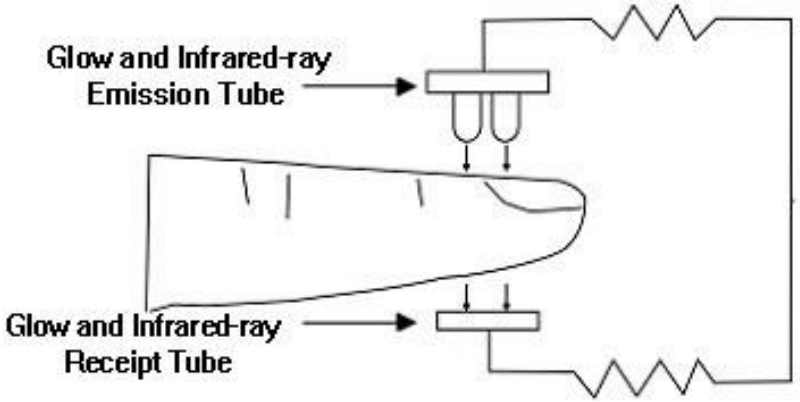
- La información devuelta por el servidor en lo referente a estadísticas se hace mediante ficheros ***CSV***.

CONTENIDOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

1. Introducción y contexto
2. Objetivos
3. Requisitos
4. Implementación
    1. Arquitectura y modelo de datos
    2. Cliente y servidor
5. Mediciones
    1. Introducción
    2. Smartphone (Cámara + flash)
    3. Microcontrolador
6. Resultados
7. Conclusiones y líneas futuras

MEDICIONES | INTRODUCCIÓN



¿Cómo funciona un Pulsioxímetro?

1. Entran en juego varias partes, pero en esencia se le hace pasar un haz de luz por el dedo, desde un emisor hasta un receptor.

2. Al oxigenarse con cada latido, hay mas cantidad de sangre en el dedo con lo que la luz que recibe el receptor es menos por un intervalo mínimo de tiempo.

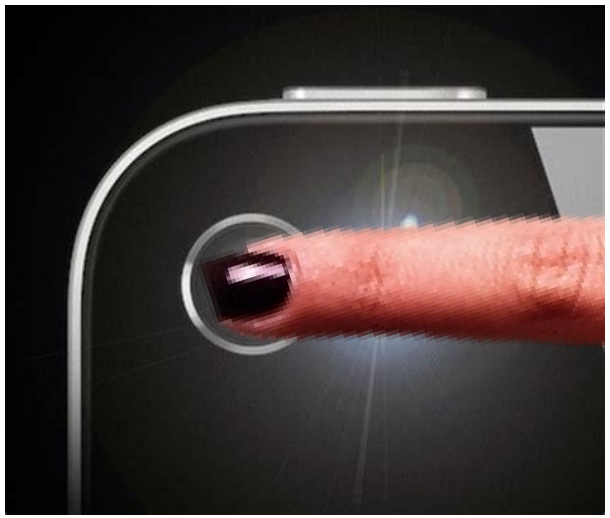
3. Al repetir esto por un intervalo de tiempo, se calcula viendo el numero de veces que la intensidad registrada ha sido menor.

CONTENIDOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

1. Introducción y contexto
2. Objetivos
3. Requisitos
4. Implementación
    1. Arquitectura y modelo de datos
    2. Cliente y servidor
5. Mediciones
    1. Introducción
    2. Smartphone (Cámara + flash)
    3. Microcontrolador
6. Resultados
7. Conclusiones y líneas futuras

MEDICIONES | SMARTPHONE

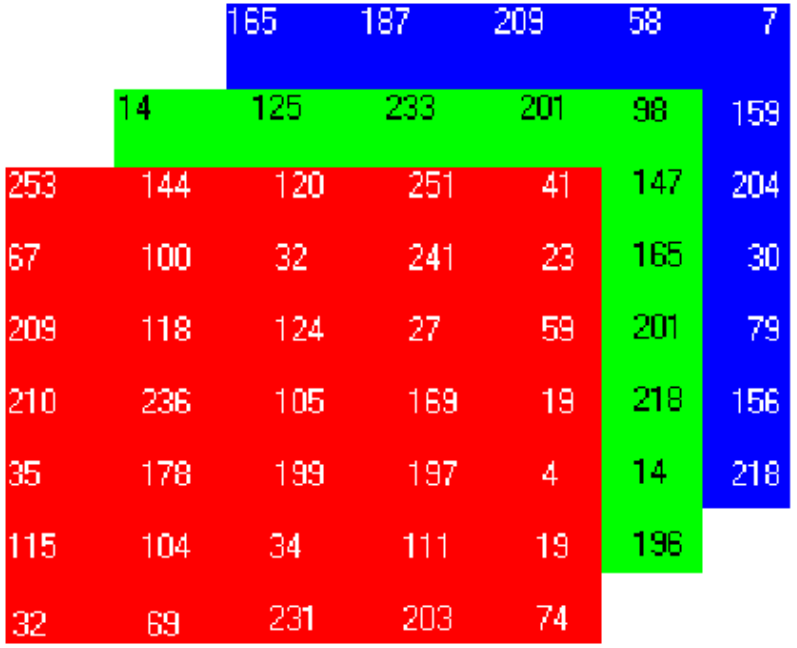


Podemos reproducir el proceso utilizando como fuente de luz el flash del teléfono y como sensor de luz la cámara del teléfono.

1. En este caso la idea es similar, solo que en vez de tener el emisor de luz y el receptor alineados en la misma vertical, usaremos el flash del smartphone como la fuente de luz, y la cámara como el sensor de luz.
2. Mediremos diferencia de color (en este caso el rojo) para ver cuando hay un flujo de sangre (debido al latido) y cuando no.
3. Repitiendo esto varias veces podemos obtener las pulsaciones por minuto.

MEDICIONES | SMARTPHONE

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa



Representación de imágenes en un sistema informático.
La imagen se representa como una colección de tres matices, mapeando los tres colores principales.

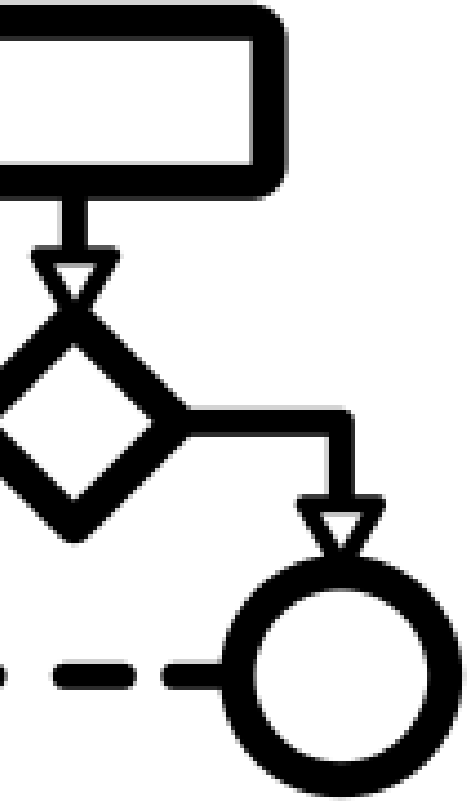
En cada matriz (filas y columnas, como anchura y altura de la imagen) se almacena un valor de intensidad entre 0 y 255 para cada uno de los colores:

- El 255,0,0 es el Rojo
- El 0,255,0 es el Verde
- El 0,0,255 es el Azul
- El 255,255,0 es el Amarillo
- … *(y así todas las posibles combinaciones.)*

Como el dedo y la sangre son de color rojo, computar las otras dos aportaciones seria calculo innecesario, ya que la variación va a ser despreciable, con lo que solo trabajaremos con una matriz, la roja.

MEDICIONES | SMARTPHONE

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa



ALGORITMO

1. Para cada momento calcularemos la media del color rojo de la imagen obtenida para ese momento.
    1. Si justo ha habido un latido, habrá mas sangre, con lo que veremos menos luz, con lo que el color será mas oscuro **(mas cerca de 0).**
    2. Si no ha habido latido, hay menos sangre, veremos más luz, y el color será mas claro **(mas cerca de 255).**

2. Lo que interesa es saber cuando hay una diferencia de color. Eso significara que ha habido latido. Un simple recuento de latidos por unidad de tiempo nos dará las pulsaciones por minuto.

MEDICIONES | SMARTPHONE


```
// Obtenemos la media de los pixeles rojos de de imagen (0 -255)
int imgAvg = ImageProcessing.decodeYUV420SPtoRedAvg(data.clone(), size.height, size.width);

if (imgAvg == 0 || imgAvg == 255) {
    processing.set(false);
    return;
}

// Calculamos la media
int rollingAverage = imgArray.calculaMedia();

TYPE newType = currentType;

// Si la media obtenida es menor que la media que teniamos guardada, cambiamos hacia abajo.
if (imgAvg < rollingAverage) {
    newType = TYPE.SUBIENDO;
    // Sumamos uno al numero de cambio/beats ocurridos
    if (newType != currentType){
        beats++;
    }

    // Si la media obtenida es mayor que la media que teniamos guardada, cambiamos hacia arriba.
} else if (imgAvg > rollingAverage) {
    newType = TYPE.BAJANDO;
}

// Metemos la media de la imagen en el array de medias.
imgArray.add(imgAvg);

if (newType != currentType) {
    currentType = newType;
}
```

Calculamos la media de intensidades para el canal rojo.

Calculamos la media de los 4 ultimos valores de media de intensidad de rojo.

Comprobamos la ultima media de rojo con la media de medias de rojo que teníamos almacenada. De esta manera detectamos si cambia. Si cambiamos a “SUBIENDO” contabilizamos un latido mas.

Metemos el valor de la media de rojos actual en la lista de valores medios del rojo. Es una lista circular con los 4 últimos valores. Como solo la usamos para calcular la media, no nos importa el orden.

MEDICIONES | SMARTPHONE

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

```
// Controlamos el tiempo anterior
long endTime = System.currentTimeMillis();

// Controlamos el tiempo inicial anterior y compromabos que han pasado 10 segundos antes de hacer nada.
double totalTimeInSecs = (endTime - startTime) / 1000d;
if (totalTimeInSecs >= 10) {

    // Dividimos el numero de beats entre el entre los segundos.
    double bps = (beats / totalTimeInSecs);
    int dpm = (int) (bps * 60d);

    // Si no esta en un rango entre 30 y 180 lo deshechamos
    if (dpm < 30 || dpm > 180) {
        startTime = System.currentTimeMillis();
        beats = 0;
        processing.set(false);
        return;
    }


    // Guardamos en un array los ultimos "beatsArraySize" valores.
    beatsArray.add(dpm);

    // Calculamos la media de los ultimos "beatsArraySize" valores.
    int beatsAvg = beatsArray.calculaMedia();
    onFrameChanged.onChanged(beatsAvg);
    startTime = System.currentTimeMillis();
    beats = 0;
}
```

Guardamos el inicio y final en milisegundos, para hacer el delta.

Calculamos el delta en segundos y esperamos a que el mismo sea al menos de 10 segundos para empezar a tener datos concluyentes.

Hacemos la conversión a ***bps*** y luego a ***bpm*** usando el delta anterior y el numero de latidos en este preciso instante.

Tenemos un array circular de tamaño 4 guardando los últimos 4 bpms registrados. Añadimos el nuevo, calculamos la media y ese será el valor que pasaremos al programa principal-

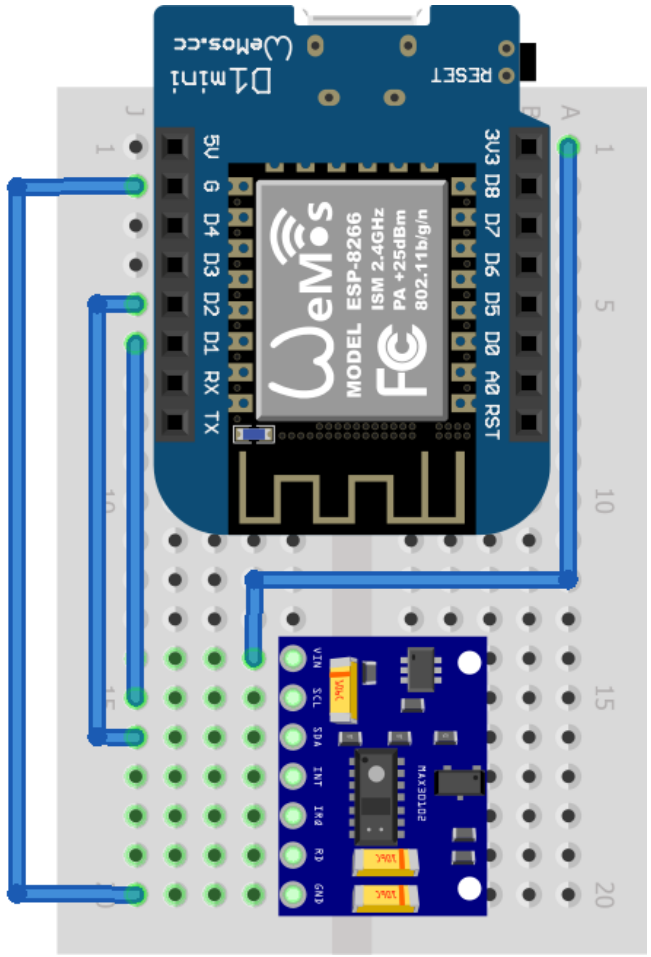
CONTENIDOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

1. Introducción y contexto
2. Objetivos
3. Requisitos
4. Implementación
    1. Arquitectura y modelo de datos
    2. Cliente y servidor
5. Mediciones
    1. Introducción
    2. Smartphone (Cámara + flash)
    3. Microcontrolador
6. Resultados
7. Conclusiones y líneas futuras

MEDICIONES | MICROCONTROLADOR

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa



El Arduino (en nuestro caso el ***WemosD1***) se conecta con el sensor (***MAX30100***) mediante el protocolo I2C, que es un protocolo serie para interconexión de controladores.

Al ser *Serial*, solo hacen falta dos hilos para realizar la conexión (D1,D2). Los otros dos son 3.3v y tierra.

La alimentación le llega al Arduino mediante su conexión (micro) USB con el smartphone.

Hace falta un adaptador *OTG* para poder conectarlo al smartphone.

MEDICIONES | MICROCONTROLADOR

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

```
void InitServer()
{

    SPIFFS.begin(); // Start the SPI Flash Files System

    server.serveStatic("/", SPIFFS, "/").setDefaultFile("index.html");

    server.begin();
    Serial.println("HTTP server started");
}

void setup()
{
    Serial.begin(115200);
    // Creamos una instancia de la clase WiFiManager
    AsyncWiFiManager wifiManager(&server, &dns);

    // Descomentar para resetear configuración
    //wifiManager.resetSettings();

    // Cremos AP y portal cautivo
    wifiManager.autoConnect("ESP8266Temp");
    //Serial.println("Ya estás conectado: "+ WiFi.localIP());
    webSocket.begin();
    //webSocket.onEvent(webSocketEvent);

    // Initialize sensor
    particleSensor.begin(Wire, I2C_SPEED_FAST); //Use default I2C port, 400kHz speed
    particleSensor.setup();                     //Configure sensor with default settings
    particleSensor.setPulseAmplitudeRed(0x0A);  //Turn Red LED to low to indicate sensor is running

    InitServer();
}
```

Iniciamos la comunicación con el USB mediante una conexión serial.

Inicializamos la conexión wifi y nos conectamos a la red previamente guardara.

Inicializamos el sensor de pulso MAX3010

MEDICIONES | MICROCONTROLADOR

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

```
void loop()
{
    webSocket.loop();

    long irValue = particleSensor.getIR();
    if (irValue > 7000)
    {
        if (checkForBeat(irValue) == true)
        {

            long delta = millis() - lastBeat;
            lastBeat = millis();

            beatsPerMinute = 60 / (delta / 1000.0);

            if (beatsPerMinute < 255 && beatsPerMinute > 20)
            {
                rates[rateSpot++] = (byte)beatsPerMinute;
                rateSpot %= RATE_SIZE;


                beatAvg = 0;
                for (byte x = 0; x < RATE_SIZE; x++)
                    beatAvg += rates[x];
                beatAvg /= RATE_SIZE;
            }

            sprintf(datagram, "{ \"pulse\": \"%lu\" }", beatAvg);
            webSocket.broadcastTXT(datagram);
            Serial.println(datagram);
        }
    }
}
```

Iniciamos la conexión con la pagina web.

Comprobamos la cantidad de luz infrarroja que detecta el sensor y utilizando una función interna de la librería del controlador determinamos si ha habido un latido.

Calculamos el numero de milisegundos que han pasado entre latido y latido, y tras repetir esto en el tiempo, obtenemos el numero de latidos por minuto.

Enviamos la información en formato JSON tanto por la conexión serial al USB del teléfono, como por el websocket a la pagina web.

MEDICIONES | MICROCONTROLADOR

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

{ pulse: “120” }

```
@Override
public void onNewData(byte[] data) {
    try {
        JSONObject pulse = new JSONObject(new String(data));

        _beat = pulse.getInt("pulse");
        runOnUiThread(() -> {
            //_heart.setColorFilter(_red ? Color.RED : Color.BLACK);
            _text.setText(_beat + " bpm");
        });
    } catch (JSONException e) {
        e.printStackTrace();
    }

}
```

1. La cadena de texto con la información en formato JSON es luego capturada por la clase Java nativa de Android y procesada.

2. Cuando todo esta correcto y el usuario pulsa el botón “Guardar”, se le devuelve el valor “_beat” al widget de ***Flutter*** para su posterior envío al servidor.

3. Como dije al principio, no todo se puede hacer con ***Flutter***, y aunque esto si, era mas claro desarrollarlo nativamente en Android y conectarlo a ***Flutter***. En cualquier caso, iOS es un sistema muy cerrado y no permite el acceso al USB a cualquier aplicación.

CONTENIDOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

1. Introducción y contexto
2. Objetivos
3. Requisitos
4. Implementación
    1. Arquitectura y modelo de datos
    2. Cliente y servidor
5. Mediciones
    1. Introducción
    2. Smartphone (Cámara + flash)
    3. Microcontrolador
6. Resultados
7. Conclusiones y líneas futuras

RESULTADOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

1. ¿Funcionan cada una de las técnicas correctamente, se han comparado los valores de frecuencia cardiaca?

TENSIÓMETRO (BPM)	SMARTPHONE (BPM)	MICROCONTROLADOR (BPM)
63	75	70
60	67	64
62	62	60
74	83	80
69	75	67

2. Comparación de ambas técnicas desde el punto de vista computacional y desde el punto de vista del programador.
3. Comparar las dos técnicas desde un punto de vista de operatividad para el médico o del usuario
4. Comparar las dos técnicas desde un punto de vista económico
5. ¿En qué escenarios/entornos recomiendas una técnica y la otra?

RESULTADOS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

1. Arduino (WeMosD1) + Sensor MAX3010

Fortalezas	Debilidades
Más sensible y preciso	Aparato externo
No le afecta la luz ambiental.	Compatibilidad con smartphone (OTG)
Económico	MAX3010 difícil de encontrar
Actualizable (por ejemplo, con sensores nuevos)	

2. Smartphone (Cámara + Flash)

Fortalezas	Debilidades
Accesibilidad (siempre contigo)	Menos preciso
Facilidad de uso (dedo sobre la cámara)	Requiere que la cámara y el flash estén cerca entre sí

CONCLUSIONES

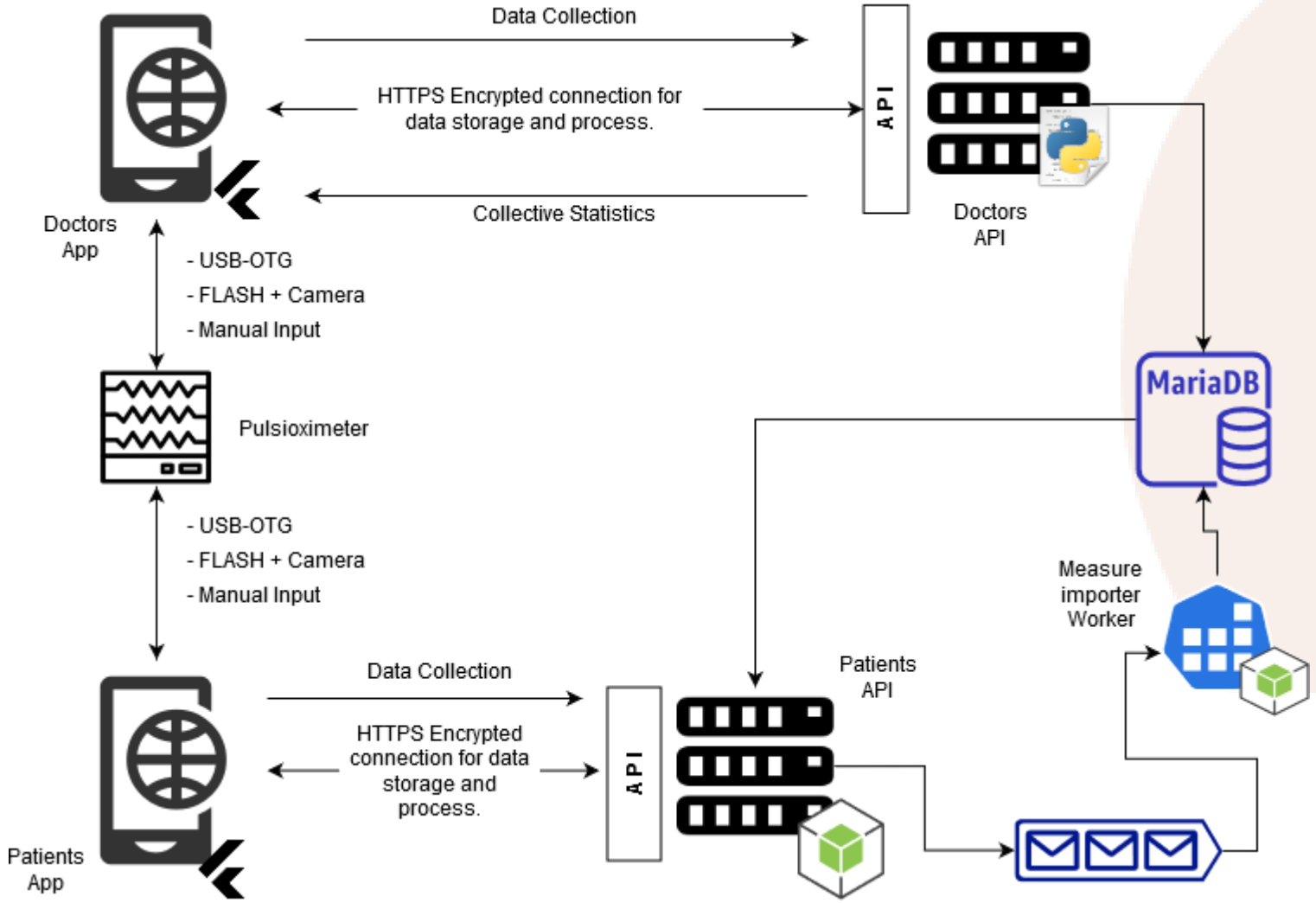
upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

- Se ha logrado implementar con éxito dos técnicas de detección de la frecuencia cardiaca utilizando la tecnología disponible en los teléfonos móviles actuales:
  - La técnica basada en la operatividad del flash y la cámara del teléfono
  - La técnica utilizando conjuntamente un microcontrolador y un sensor de pulso externos junto con el teléfono móvil.

- Basándonos en resultados obtenidos, se tiene confianza en que esta aplicación es de gran utilidad para los médicos: tanto para gestionar a sus pacientes y ver rápidamente una posible patología, como para almacenar una gran cantidad de datos que poder utilizar luego anónimamente para realizar estudios estadísticos.
  - La confianza en las técnicas implementadas en este proyecto se ha reforzado tras recibir la valoración positiva de un médico cardiólogo interesado en el proyecto, el cual valoró el trabajo del siguiente modo: "***le veo futuro a la aplicación en el campo de seguimiento de pacientes, y en el campo de la investigación***".

LÍNEAS FUTURAS

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa



Funcionalidad Paciente-Doctor

- Recogida de datos en la app de paciente
- Introducción de diagnostico en la app de doctor.
- Mensajería de consulta entre paciente y doctor.

Actualización de la arquitectura

Se plantea introducir un sistema de colas así como un servidor nuevo, para gestionar la app de cliente y la importación de datos.

DEMO | DESCARGAR

upna

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

GRACIAS POR ESCUCHARME…

… QUEDO A SU DISPOSICIÓN PARA CUALQUIER TIPO DE PREGUNTAS

TENSIONAPP:
SISTEMA DE DETECCIÓN DE LA FRECUENCIA CARDIACA MEDIANTE EL SENSOR FOTOGRÁFICO DE UN SMARTPHONE Y MEDIANTE UN MICROCONTROLADOR.

https://tensionapp.herokuapp.com/

Autor: Miguel Fuertes Fernández
Tutor: Javier Rodríguez Falces (Universidad Pública de Navarra)
Director: Javier Rodríguez Falces (Universidad Pública de Navarra)

Miguel Fuertes Fernandez
Ingeniero Informático
https://mfuertes.net