



Sistema FALTAS

Faltas del Alumnado con la Tecnología
AS608

Documentación del proyecto

Diseño de Sistemas Interactivos

Desarrolladores:

- José Alberto Sánchez-Camacho
- María Blanco González-Mohino
- Pablo Velasco Crespo

Fecha:

14 de diciembre de 2021

ÍNDICE

Índice	1
Introducción.....	1
Motivación.....	1
Tecnologías Utilizadas.....	1
Hardware	1
Software	1
Objetivos	2
Objetivos Principales	2
Objetivos Secundarios	2
Desarrollo.....	3
Organización.....	3
Desarrollo de Objetivos Principales.....	3
Desarrollo de Objetivos Secundarios	5
Problemas y Soluciones	6
Bibliografía	7

INTRODUCCIÓN

MOTIVACIÓN

Dada la problemática de la abstinencia en clase por parte de los alumnos de todos los ámbitos académicos, desde secundaria y ciclos medios y superiores a bachillerato, universidad y estudios más elevados, se ha decidido hacer un sistema para combatir la falta de estos alumnos sin que se pierda un tiempo excesivo en la recolección de las faltas.

TECNOLOGÍAS UTILIZADAS

Para la resolución del proyecto se han utilizado para siguientes tecnologías:

Hardware

Lector de huellas AS608¹, con capacidad para leer, almacenar y comparar dos huellas dactilares, este lector puede almacenar internamente 137 huellas.

ESP32, placa que contiene Wi-Fi/Bluetooth todo en uno, con procesador integrado y capacidad para conectar y conectarse con varios periféricos.

Software

Arduino, es una plataforma flexible y fácil de utilizar para los creadores y desarrolladores de sistemas embebidos.

PHP, lenguaje de programación que permite el desarrollo de aplicaciones web dinámicas.

Python3, lenguaje de programación de alto nivel apto para cualquier tipo de programación.

MySQL, es un sistema de gestión de base de datos de código abierto.

JavaScript, HTML, CSS, JQuery & AJAX, utilizados para crear páginas web y hacer consultas a bases de datos.

¹ Documentación del lector AS608: <https://blog.uelectronics.com/tarjetas-desarrollo/arduino/como-utilizar-el-lector-de-huella-dactilar-as608/>

OBJETIVOS

Para la resolución de este proyecto se han marcado una serie de objetivos básicos o principales para que el sistema FALTAS funcionase correctamente y unos objetivos secundarios. A continuación, explicaremos estos dos tipos:

OBJETIVOS PRINCIPALES

Como objetivos básicos para resolver el problema nos marcamos 4:

- Una vez obtenido el lector *AS608* este deberá leer la huella, y comparar si esta huella es igual a otra ya almacenada.
- Se deberá construir una base de datos, ya que no tenemos acceso al sistema de almacenamiento de la Junta de Educación y no se tiene información sobre cómo se almacenan las faltas.
- Se utilizará la *ESP32* a modo de intermediario con el lector *AS608* para realizar el almacenaje y lectura de huellas de la base de datos y su escritura en el lector para realizar la comparación de huellas; también utilizamos la placa para comunicar los diferentes dispositivos.
- Asignar si el alumno ha asistido a clase en el periodo oportuno de tiempo, si no fuese así y la clase hubiese acabado o llegase en un periodo de tiempo considerado por el centro como falta, se deberá asignar la falta al alumno.

OBJETIVOS SECUNDARIOS

Como objetivos secundarios a modo de mejora del proyecto:

- Construir una página web para que el almacenaje de la huella del alumno sea más fácil.
- Indicar cuando el lector esté listo para leer y cuando no está disponible.

DESARROLLO

ORGANIZACIÓN

Todos los archivos del proyecto se pueden encontrar en este [enlace](#).

Una vez ingresamos en la carpeta de proyecto nos encontramos la estructura mostrada en la imagen derecha.

Los archivos incluidos en la carpeta *ARDUINO* son los dos programas creados para cargar en la placa *ESP32*.

El archivo *config.php* se encuentra explicado en el apartado de [manual](#). Por otro lado, el archivo *index.php* es la página principal de la web. El archivo *templateControler.py* se encuentra explicado en el apartado de [manual](#).

En la carpeta *addIndex* se encuentran los archivos encargados del registro.

La carpeta *bbdd* obtenemos el script para ejecutar y crear la base de datos.

En la carpeta *css* se encuentra la hoja de estilo de la página web.

La carpeta de *scripts* almacena los archivos que modifican la base de datos.

Los archivos *Templates* se encuentran explicados en el apartado de [desarrollo](#).

A continuación, se detalla el desarrollo de los diferentes objetivos marcados.

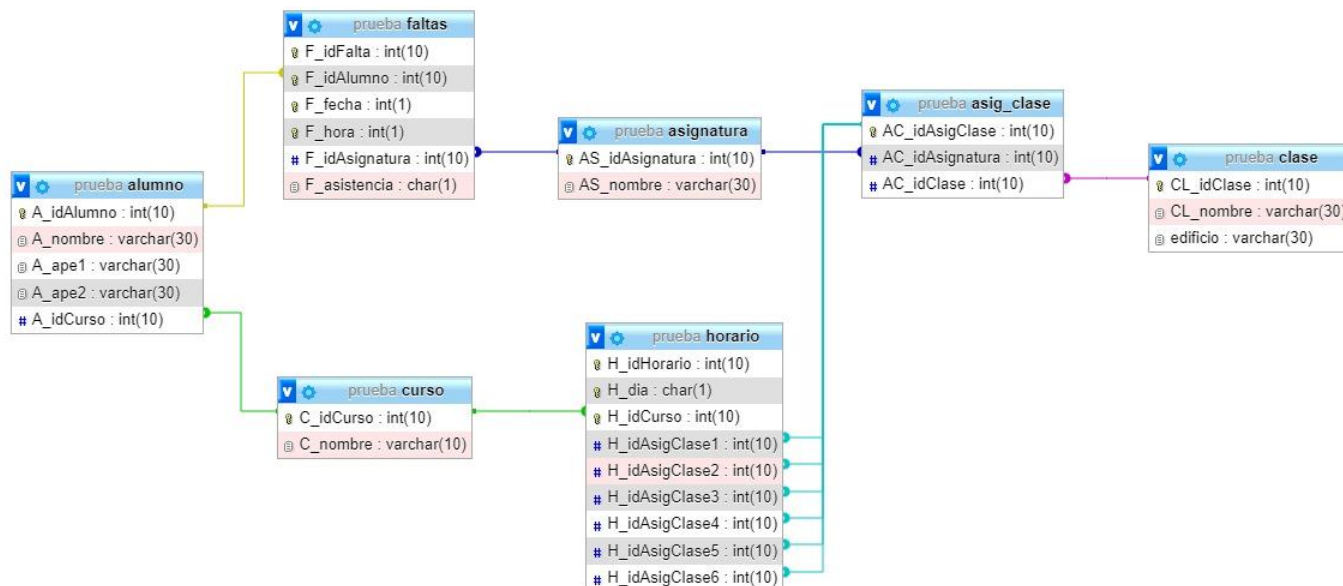
```
\---Sistema_FALTAS
|   config.php
|   index.php
|   templateControler.py
|
+---addIndex
|   control_data.php
|   control_huella.php
|   control_setup.php
|
+---ARDUINO
|   +---autenticar
|   |   autenticar.ino
|   |
|   \---registrar
|       registrar.ino
|
+---bbdd
|   bd_v01.sql
|
+---css
|   style.css
|
+---scripts
|   insertarAlumno.php
|   script.js
|   updateFalta.php
|
\---Templates
    1.json
    2.json
```

DESARROLLO DE OBJETIVOS PRINCIPALES.

Para realizar el programa correspondiente a la lectura y comparación de la huella se utilizó *Arduino*, en él, utilizamos una librería llamada *FPM*, librería de código abierto encontrada en *Github*, que nos permitía la escritura de huellas en buffers del lector y la comparación de estos. Téngase en cuenta que el lector lee la huella y la almacena en un array de longitud 738_{10} , los cuales son números en base 10.

Para el soporte web hemos utilizado el paquete *XAMPP* que integra el sistema de gestión de base de datos *MySQL* y el servidor web *Apache* e interpreta el lenguaje *PHP*.

El diagrama Entidad-Relación de la base de datos es el que se muestra a continuación, en él, cada tabla representa una entidad, y los atributos de estos son las columnas de la tabla:



Una vez realizadas las lecturas y la base de datos necesitamos incorporar las conexiones con los diferentes dispositivos, para esto utilizamos *Arduino* y cargamos el programa correspondiente en la placa ESP32, para las diferentes conexiones utilizamos la librería llamada *HardwareSerial*, la cual, se encuentra integrada en el paquete de *Arduino*, también se utiliza las librerías *WiFi*, *WiFiMulti* y *HTTPClient* para conectarnos a internet, útil para conectarse con la base de datos.

Para considerar que un alumno ha faltado a la clase se asigna como F (falta) en la base de datos a todos los alumnos, si estos llegan en un periodo intermedio de tiempo (normalmente establecido por el centro) se asignará una R (retraso) como símbolo de que el alumno ha llegado tarde, si el alumno llega en un plazo menor al de R será asignado como P (presente).

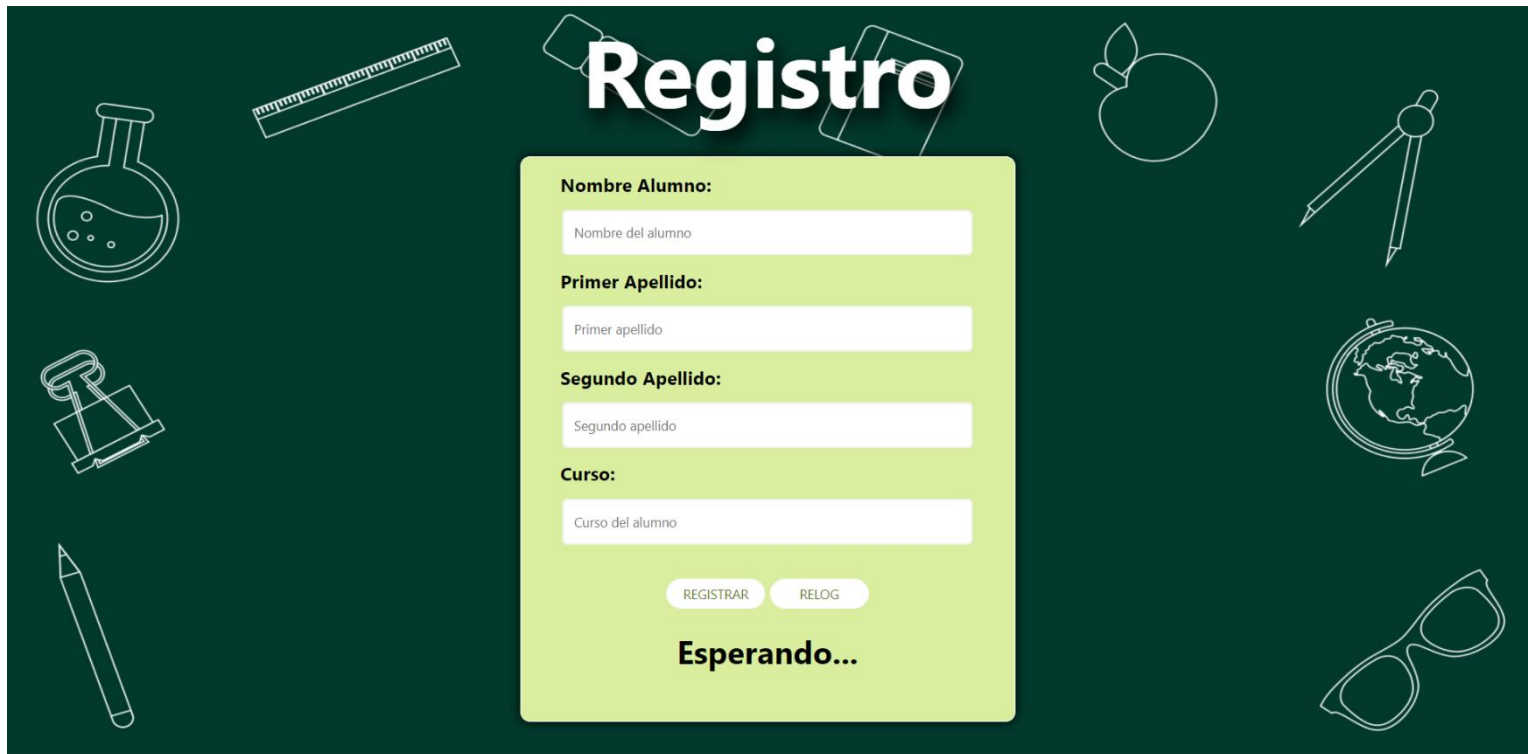
Hemos creado un controlador que genera archivos *.json* cada hora de clase (hemos supuesto que se imparten un total de 6 al día). Éstos son nombrados por el idClase que tiene, es decir, nuestro “identificador” del *ESP32*. Los *.json* contienen todas los *idAlumnos* y huellas correspondientes con cada estudiante. El controlador, que se ejecuta de manera constante (bucle), si está dentro de un intervalo de tiempo, significa que la hora va a terminar; por lo que tiene que cargar las nuevas huellas e *idAlumno* de la siguiente clase.

A su vez, el *autenticador.ino* tiene una función que comprueba la hora que es. Esto nos sirve para saber cuándo se han añadido las nuevas huellas. Cuando esto ocurre, se cargan los nuevos datos del *.json*, es decir, las huellas e *idAlumno*. Desarrollo de

DESARROLLO DE OBJETIVOS SECUNDARIOS

JavaScript, HTML, CSS, JQuery, AJAX y *PHP* han sido utilizados para crear una página web para el registro de la huella en la base de datos, esta página web también usa scripts asociados a consultas en la base de datos para introducir los datos.

Front-End de la página web:

The image shows a web registration form titled "Registro" in a large, bold, white font. The form is set against a dark green background decorated with white line-art icons of school supplies: a beaker, a ruler, a compass, a globe, a paperclip, a pencil, and a pair of glasses. The form itself is a light green rectangle with rounded corners. It contains four input fields with labels in bold: "Nombre Alumno:", "Primer Apellido:", "Segundo Apellido:", and "Curso:". Each label is followed by a white input box with a light green border. Below the input fields are two buttons: "REGISTRAR" and "RELOG", both in white text on light green rounded rectangles. At the bottom of the form, the text "Esperando..." is displayed in a bold, black font.

Para que el lector resulte más intuitivo a la hora de poder poner y quitar el dedo se han introducido dos luces leds asociadas a indicar si el dedo se ha leído y cambiado de estado en la base de datos (luz amarilla) y si se ha producido un error en cualquier punto y se debe volver a poner el dedo (luz roja).

PROBLEMAS Y SOLUCIONES

En este apartado se expondrán los problemas y soluciones que hemos encontrado a lo largo del proyecto:

- Problema: el primer problema encontrado nos llegó por parte del propio lector, la librería que nos proporcionaban los vendedores (*AdafruitFingerprint*), no implementaba ninguna opción para escribir las diferentes huellas en los buffers, es decir, nosotros como usuarios podríamos leer nuestra huella y el lector nos la compararía con una huella que tuviese internamente almacenada.

Solución: en este momento nos planteamos tres posibles soluciones, la primera sería implementar un algoritmo de comparación de huellas, el cuál resultaba excesivamente complicada porque no conocíamos la lectura interna que realizaba el lector. La segunda opción fue cambiar el firmware por *Microphyton*, fue la última opción a realizar ya que el proyecto se encontraba demasiado desarrollado. La tercera opción y la elegida fue encontrar una librería que nos permitiese enviar a uno de los dos buffers del lector la huella almacenada en la base de datos, para esto encontramos la librería *FPM* que nos permitía mover la huella a un buffer con el método *move_template()* para que el lector lo comparase de manera interna con la huella que lee en el momento.

- Problema: al utilizar la primitiva *GET* del lenguaje *HTTP* la comunicación fallaba ya que no recibía siempre respuesta de la web

Solución: se cambió de la librería *WiFi* a *WiFiMulti*, aunque seguimos utilizando la librería *WiFi* para la parte de horario.

- Problema: al enviar las huellas al servidor utilizamos de nuevo la primitiva *GET*, esto daba problemas debido a los espacios después de las comas.

Solución: se quitaron los espacios a la hora de almacenar en la base de datos.

- Problema: a la hora de leer las huellas el *GET* devolvía dicha huella como un *String*, mientras que el lector usa un array de *Integer* para realizar el método de comparación de huellas.

Solución: se creó un método *parser* el cual lee el *String* y detecta la coma como delimitador creando, así, un array de *Integer*.

- Problema: la complicación en la comparación de horas, ya que las clases que ocupan dos tramos horarios podrían dar problemas, por ejemplo, si una clase empieza a las 10:00 y otra a las 10:50.

Solución: la solución encontrada fue bastante simple, dado que un día tiene 24 horas y cada hora 60 minutos, definimos los tramos horarios en minutos en vez de horas, es decir, para la hora 10:50 se multiplicaría $10 * 60$ lo que nos daría un total de 600 más 50 minutos, lo que nos daría un total de 650 minutos.

ESCALABILIDAD

NÚMERO DE HUELLAS

El sistema está preparado para almacenar una huella por alumno, lo cual puede llegar a ser un problema porque si le pasa algo en ese dedo habría que registrarlo de nuevo. Pero tampoco podemos almacenar varias huellas por alumno por la limitación de almacenamiento de la AS608 (137 huellas). Si decidiéramos almacenar 5 huellas por alumno, al encontrarnos con una clase de 30 ya se superaría ese límite, por lo que para escalar el sistema y poder almacenar más de una huella por alumno, necesitaríamos otro lector de huella con más capacidad.

ADAPTABILIDAD

Esta tecnología también puede usarse para ámbitos fuera del académico. En cualquier empresa, establecimiento, gimnasio... que necesite controlar las personas que entran o salen puede utilizarse.

AMPLIACIÓN

El sistema está hecho en local con una base de datos nuestra, habría que conectar los resultados obtenidos por la base de datos con los servicios que se utilizan en cada comunidad para controlar la presencia de los alumnos. Por ejemplo, en Castilla-La Mancha habría que utilizar Educamos.

BIBLIOGRAFÍA

(n.d.). Retrieved from NTP Server With ESP32:
<https://lastminuteengineers.com/esp32-ntp-server-date-time-tutorial/>

(n.d.). Retrieved from Arduino Foro: <https://forum.arduino.cc/t/converting-string-to-int-array/628157/5>

GitHub. (n.d.). Retrieved from <https://github.com/brianrho/FPM>

GitHub. (n.d.). Retrieved from https://github.com/iotics/esp32_esp8266_get_request

MySQL and ESP32. (n.d.). Retrieved from <https://randomnerdtutorials.com/esp32-esp8266-mysql-database-php/>