UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCOLA POLITÈCNICA SUPERIOR DE GANDIA

GRADO EN ING. SIST. DE TELECOM., SONIDO E IMAGEN

“Desarrollo de un sistema de control de tanques de almacenamiento de agua”

TRABAJO FINAL DE GRADO

Autor/a:

Jose Agustín Cabrerizo Martínez

Tutor/a:

Jordi Bataller Mascarell

GANDIA, 2021

Desarrollo de un sistema de control de tanques de almacenamiento de agua.

Autor: Cabrerizo Martínez, Jose Agustín

Director: Bataller Mascarell, Jordi

RESUMEN

En este Trabajo Final de Grado hemos diseñado e implementado un sistema para el control del nivel de llenado de almacenamiento de agua compuesto por un sensor de ultrasonidos conectado a un microcontrolador que envía información a una base de datos y una aplicación para teléfono móvil que permite a un usuario realizar el seguimiento del tanque remotamente.

PALABRAS CLAVE

Control y gestión tanque de agua, sensor de ultrasonidos, microcontrolador Arduino, aplicación para teléfono móvil, servidor web, base de datos.

ABSTRACT

In this Final Degree Project we have designed and implemented a system to control the filling level of water storage composed of an ultrasound sensor connected to a microcontroller that sends information to a database and a mobile phone application that allows a user to track the tank remotely.

KEYWORDS

Water tank control and management, ultrasound sensor, Arduino microcontroller, mobile phone application, web server, database.

ÍNDICE

1 - INTRODUCCIÓN

Hoy en día es habitual el uso de tanques de almacenamiento de agua con diferentes fines: abastecimiento de viviendas aisladas o no, depuración mediante ósmosis, para riego agrícola, almacenamiento de excedentes, reaprovechamiento de aguas, etc.

Algo tan sencillo como controlar el nivel de llenado de cada tanque puede ser complicado si no se tiene un acceso fácil o hay un número importante de tanques que vigilar.

Es evidente que la tecnología actual nos ofrece sensores y medios para monitorizar dichos tanques y empezar a hacerlos "inteligentes", ya que todo aquello que facilite la vigilancia y control de tanques de agua de forma remota mejorará y promoverá su uso.

Las prácticas en empresa que se realizan al final de la titulación, en nuestro caso, tuvieron lugar en una empresa dedicada al desarrollo de aplicaciones informáticas para la resolución de problemas de ingeniería, donde acordamos que éste podría ser un buen tema para realizar tanto las prácticas como el presente trabajo final de grado.

Así pues, los objetivos de este trabajo son los siguientes:

- Estudio y valoración de distintos tipos de sensores que puedan ser útiles en la monitorización de tanques de almacenamiento de líquidos.
- Estudio y aprendizaje del uso y programación de dichos sensores a través de un microcontrolador, y la forma en que éste puede transmitir los datos a un servicio remoto.
- Montaje de un prototipo que integre un sensor con su microcontrolador.
- Diseño e implementación de los programas para el microcontrolador.
- Diseño e implementación de una "APP " para teléfono móvil que permita la visualización del estado de un tanque a partir de la información que el microcontrolador va dejando en una base de datos al efecto.

Esta memoria está organizada de la siguiente forma:

- Elección de herramientas: En este capítulo veremos las herramientas hardware y software que hemos necesitado para llevar a cabo nuestro proyecto.
- Diseño de la solución: En este capítulo explicaremos como hemos llevado a cabo la puesta en marcha de nuestro dispositivo.
- Manuales: En este capítulo veremos los pasos que debemos seguir para la correcta instalación del dispositivo.
- Conclusión: En este capítulo plantearemos posibles mejoras para el proyecto.

2 - ELECCIÓN DE HERRAMIENTAS

2.1 - HARDWARE

MICROCONTROLADOR

Nuestra decisión ha sido trabajar con el microcontrolador NodeMCU V2 Wifi - ESP8266. La principal ventaja de este microcontrolador es que incorpora un chip con capacidades WiFi lo que nos permitirá enviar datos a través de internet a nuestro servidor.

Otro aspecto muy positivo es que se trata de un microcontrolador 'Open Hardware' y esto nos brinda posibilidades muy amplias a la hora de encontrar mucha información de manera amplia y gratuita en internet.

En la ilustración 1 podemos ver distintas imágenes del circuito con el chip ESP8266 que integra una conexión USB de alimentación y para poder programarlo así como un conjunto de pines donde poder conectar sensores.

Este chip lo hemos programado en C usando el entorno de desarrollo de Arduino. A través de él se puede gestionar cualquier sensor usando las entradas y salidas digitales.

Ilustración 1. NodeMCU V2 WiFi

SENSOR DE ULTRASONIDOS

Nuestra decisión ha sido trabajar con este sensor ya que la empresa donde he estado realizando las prácticas conocía su funcionamiento y sus principales características encajaban perfectamente con el proyecto. Otra de las principales ventajas que ha hecho que nos decantemos por él es que este sensor es barato y sencillos de usar.

Un sensor de ultrasonidos es un dispositivo que se utiliza para medir distancias. Su funcionamiento se base en el envío de un pulso de alta frecuencia no audible por el ser humano. Este pulso rebota en los objetos cercanos y es reflejado hacia el sensor, que dispone de un micrófono adecuado para la recepción de esa frecuencia.

Midiendo el tiempo entre pulsos y conociendo la velocidad del sonido podemos estimar la distancia del objeto contra cuya superficie impacta el impulso de ultrasonidos.

$$v = \frac{s}{t} \Rightarrow s = v \times t$$

Ilustración 2. Sensor de ultrasonidos HC-SR04

PLACA PROTOBOARD

Para realizar toda la interconexión de nuestros componentes hemos decidido hacerlo a través de una placa protoboard ya que nos facilita mucho el trabajo en cuánto a que podemos hacer pruebas durante el montaje sin necesidad de soldar ninguna conexión.

Esta placa es un tablero con orificios que se encuentran conectados eléctricamente entre sí de manera interna, en el cual se pueden insertar componentes electrónicos y cables para el armado y prototipado de circuitos electrónicos y sistemas similares.

Está hecha de dos materiales, un aislante, generalmente un plástico, y un conductor que conecta los diversos orificios entre sí. Uno de sus usos principales es la creación y comprobación de prototipos de circuitos electrónicos antes de llegar a la impresión mecánica del circuito en sistemas de producción comercial.

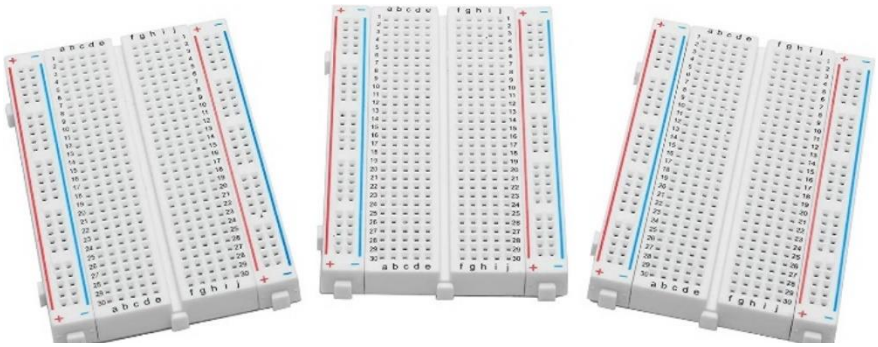

Ilustración 3. Placa protoboard

COMPONENTES ELECTRÓNICOS

Para realizar todo nuestro circuito eléctrico hemos necesitado un transistor BC547B, 6 resistencias de 10KΩ y varios cables jumper, el motivo es que la salida del microcontrolador funciona a una tensión de 3,3V y la entrada de nuestro sensor funcina a 5V y viceversa, por tanto para la regulación de tensión hemos tenido que usar un circuito con 3 resistencias de 10k y un transistor y para el circuito de la salida utilizaremos tres resistencias para bajar la tensión nominal a 3.3V.

Ilustración 5. Resistencias 10KΩ

Ilustración 6. Transistor BC547B

Ilustración 4. Cables jumper

FUENTE DE ALIMENTACIÓN

Para poder alimentar todo nuestro circuito necesitamos una fuente de alimentación de 5V de tensión. Dicha fuente alimentará la placa protoboard para dat tensión a todo el circuito.

Ilustración 7. Fuente de alimentación

RECIPIENTE GRADUADO

Para realizar nuestras pruebas hemos decidido utilizar un recipiente graduado ya que así sabremos con mayor exactitud el porcentaje de llenado que tenemos en nuestro tanque.

Debemos de tener en cuenta que el recipiente debe ser cilíndrico, es decir la base debe ser del mismo diámetro que la parte de arriba. Si intentamos medir en un recipiente que no es cilíndrico, por ejemplo, en forma de cono, las medidas no serán proporcionales.

Ilustración 8. RECIPIENTE GRADUADO

2.2 - SOFTWARE

BASE DE DATOS

Para realizar nuestra base de datos en la que almacenaremos toda la información que intercambiamos entre el ESP8266 y nuestro dispositivo móvil hemos utilizado una base de datos llamada MariaDB.

Tiene una alta compatibilidad con MySQL ya que posee las mismas órdenes, interfaces, API y bibliotecas. Toda la programación de dicha base de datos la hemos realizado en MySQL y JSON.

Se ha contratado un HOSTING o alojamiento web con tecnología PLESK en el cual tendremos acceso a las base de datos MySQL y podremos alojar nuestros archivos PHP para el manejo de la información.

Una Base de Datos es una herramienta que funciona como “almacén”, es decir, guarda grandes cantidades de información de forma organizada para poder encontrarla y utilizarla de manera fácil y ordenada.

Las bases de datos relacionales son las más usadas actualmente para administrar datos de forma dinámica. Permite crear todo tipo de datos y relacionarlos entre sí.

Los datos son almacenados en registros que son organizados en tablas, de esta forma pueden asociarse los elementos entre sí muy fácilmente, además se pueden cruzar sin ninguna dificultad.

SERVIDOR WEB

Para la programación del servidor web hemos utilizado el lenguaje PHP. Dicho servidor usa una tecnología de gestión de peticiones en el cual es capaz de recibir peticiones POST/GET/PUT o DELETE y devolver información a través del protocolo HTTP/HTTPS.

En este servidor se alojan todos los archivos necesarios PHP para el manejo de la información de servidor (con PHP podremos tener acceso a MySQL, servidor de base de datos alojado en el mismo servidor en la nube).

Un servidor web es un software que forma parte del servidor y tiene como misión principal devolver la información (páginas) cuando recibe peticiones por parte de los usuarios.

Para el funcionamiento correcto de un servidor web necesitamos un cliente web que realice una petición http o https a través de un navegador como Chrome, Firefox o Safari y un servidor donde esté almacenada la información.

IDE DE DESARROLLO

Para el desarrollo de la aplicación hemos usado el IDE 'Brackets', es un IDE de desarrollo totalmente de código libre escrito en tecnología WEB y compilado usando tecnología 'ELECTRON', la cual transforma dicho código en código nativo para ordenadores, ya sean Windows o Linux/Mac.

Hemos decidido trabajar con este IDE ya que es un muy versátil y nos permite crear proyectos de una forma muy rápida solo indicando un pequeño espacio de trabajo (carpeta). A través de sus plugins en su tienda (oficiales y de terceros) nos permite dar compatibilidad a cientos de diferentes tipos de lenguajes de programación, gracias a esto nos permite programar al mismo tiempo, HTML, CSS, JavaScript o PHP, haciendo mucho mas fácil este trabajo.

Para nuestro proyecto hemos usado un plugin llamado 'Synapse' el cual nos permite el acceso nativo a nuestro alojamiento WEB mediante la tecnología FTP (File transfer protocol) pudiendo trabajar directamente con nuestro servidor web haciendo cambios en tiempo real.

APACHE CORDOVA

Para realizar la programación de la APP hemos utilizado 'Apache Cordova' ya que nos permite realizar una 'CROSS-PLATAFORM' entre unas entradas (código HTML), una API de Cordova y unos plugins, pudiendo así transformar el código HTML/JavaScript a código nativo Android (JAVA o KOTLIN). Es un framework totalmente 'open source' que está mantenido por la fundación Apache. Para poder utilizar cordova tenemos que instalar previamente NODE JS. En Cordova hemos usado diferentes plugins que explicaremos más adelante.

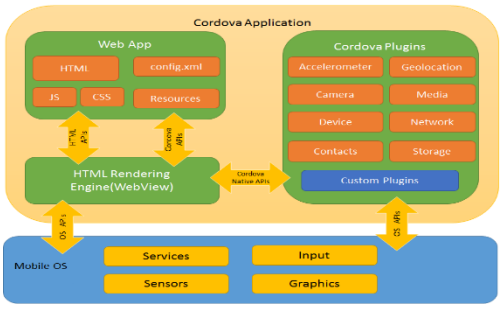

Ilustración 9. CROSS-PLATAFORM

PLUGINS

Un plugin es una aplicación (o programa informático) que se relaciona con otra para agregarle una función nueva y generalmente muy específica.

Los plugins proporcionan acceso a la funcionalidad de plataforma y dispositivo que está normalmente disponible para aplicaciones basadas en web.

Las principales características de Cordova API se implementan como plugins y permiten funciones tales como escáneres de código de barras, comunicación NFC, o adaptar el calendario, interfaces, etc.

Para nuestro proyecto hemos usado los siguientes plugins:

- White List: Viene con cordova por defecto, es un plugin que sirve para poder hacer “intents” y saltarnos el cross origin.
- Googe services: Depende de One Signal para poder usar notificaciones.
- One Signal: Habilita la posibilidad de realizar y recibir notificaciones en nuestros dispositivos.
- Bar code scanner: Plugin que nos permite abrir la cámara del dispositivo y leer el código qr para obtener así la información que hay en él.

FRAMEWORK7

Framework7 es un framework de código abierto y gratuito para desarrollar aplicaciones móviles y de escritorio con apariencia nativa.

En nuestro caso concreto Framework7 Core que usa HTML5 para la parte visual y CSS3 para darle estilo. Hemos usado JavaScript para la lógica de control.

Su librería contiene la programación de todos los componentes nativos de Android y iOs, se pueden usar de forma sencilla a través de HTML5, además en nuestro caso hemos usado la versión CORE que nos permite usar JavaScript de forma nativa para realizar la programación de comportamiento del componente.

También hemos incluido la librería DOM7 que sirve para usar ‘templates’ como ‘Activities’ y asi incrustar dicho template en el index de la aplicación a través de peticiones XHR request.

ANDROID STUDIO

Hemos decidido trabajar con Android Studio porque en nuestro caso al usar CORDOVA transformamos nuestro código HTML en código JAVA que es capaz de ser leído por Android Studio, y con este, realizamos la compilación de la aplicación para poder generar el archivo APK que se quedará instalado en el dispositivo.

Android Studio es un IDE basado en IntelliJidea en el cual contiene todas las herramientas necesarias para realizar la programación nativa de Android usando JAVA/Kotlin y XML, además de contener la herramienta GRADLE que es necesaria para compilar las aplicaciones y transformar su código en código leíble por la máquina virtual de JAVA del dispositivo.

NOTIFICACIONES

En nuestro caso concreto hemos elegido trabajar con la API de One Signal escrita en PHP ya que el encargado de emitir las notificaciones será nuestro servidor web.

OneSignal es una plataforma que permite el envío de Notificaciones Push a través de diferentes plataformas que estén registradas a nuestra aplicación. Una vez el dispositivo ha sido registrado adecuadamente a nuestra APP , queda listo para el envío de 'push notifications' desde nuestro 'Dashboard'.

En nuestro caso concreto, necesitaremos enviar mensajes específicamente a uno o varios usuarios. Para ello necesitaremos el APP Id y la REST API key de nuestra aplicación, que se encuentran en la sección de Keys & ID's en APP Settings en la plataforma de One Signal. Debemos enlazar el Player ID con el usuario que está usando la aplicación y almacenarlo en nuestra base de datos, de esta manera lo podemos usar luego con nuestro propio request a la REST API de OneSignal.

3 - DISEÑO DE LA SOLUCIÓN

3.1 - HARDWARE: MONTAJE Y ALIMENTACIÓN

De nuestra placa necesitaremos 2 entradas y salidas digitales, en nuestro caso concreto serían D1 para la entrada y D2 para la salida, también utilizaremos el pin de 3V y el GND (toma de tierra) para alimentar nuestra placa. Nuestra entrada D1 la conectaremos al 'Trigger' del sensor y la salida D2 la conectaremos al 'Echo', de esta manera conseguimos que nuestra información viaje del sensor hacía nuestra placa y viceversa.

Como ya hemos comentado antes, nuestra placa y sensor se alimentan a tensiones diferentes, por lo tanto, debemos de añadir unos circuitos intermedios para resolver dicho problema y hacer que nuestra placa pueda funcionar a 5V. Por ello introduciremos entre la entrada D1 y el 'Trigger' dicho circuito adicional, también haremos lo mismo en el caso de la salida D2 y el 'Echo'.

Utilizaremos fritzing para hacer la simulación virtual de lo que sería nuestro diseño. Antes de empezar a montar y programar el circuito, tenemos que asegurarnos que todos los componentes funcionan correctamente. Lo que haremos es probar por separado cada componente y una vez nos aseguremos de que todo funciona correctamente, procederemos al montaje. El diseño de nuestro circuito quedaría de la siguiente manera:

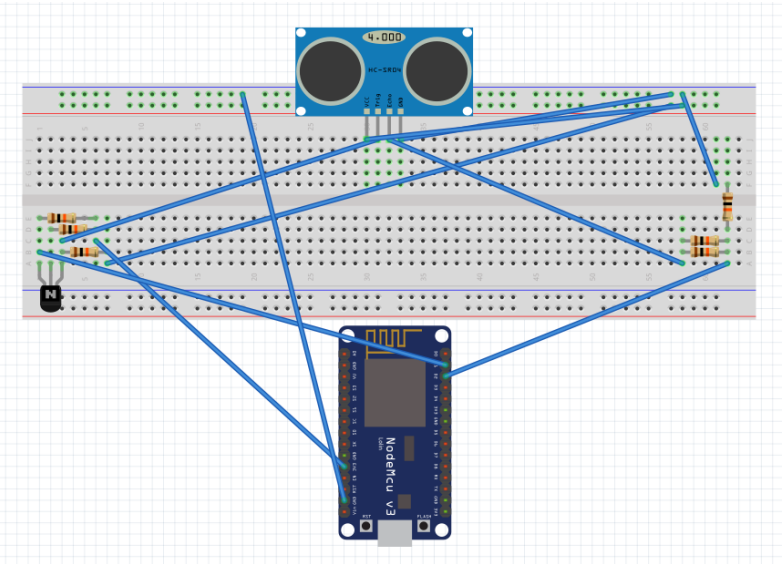

Ilustración 10. Montaje circuito

El sensor de ultrasonidos lo hemos situado en la parte superior del tanque, lo podemos hacer de muchas maneras, en nuestro caso hemos decidido aguantar el sensor con dos palos de madera cruzando toda la parte superior del tanque fijándola bien con adhesivo.

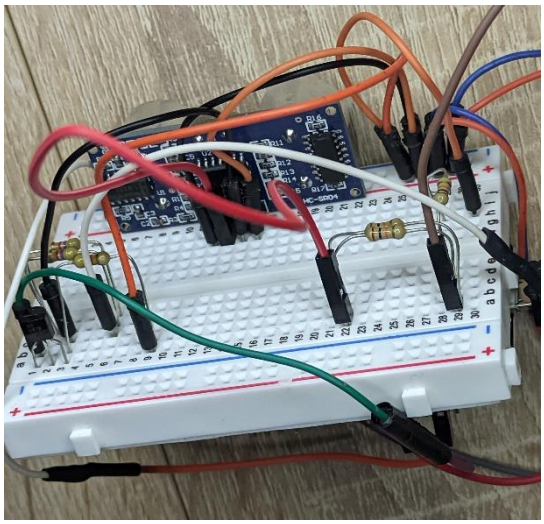

Ilustración 11. Divisores de tensión

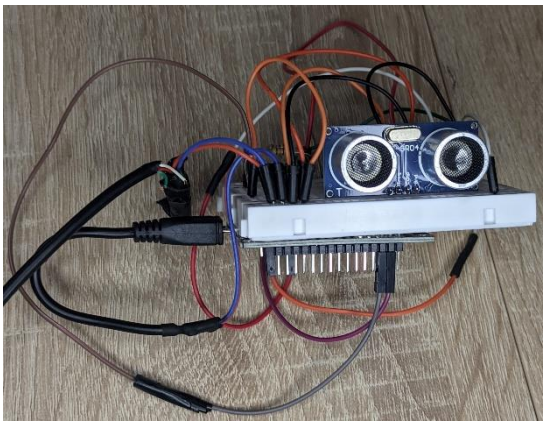

Ilustración 12. Montaje final

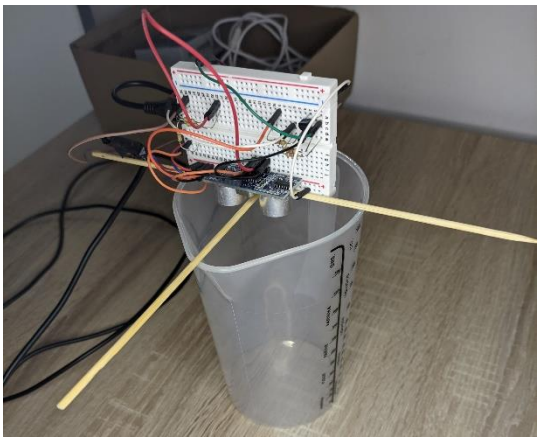

Ilustración 13. Recipiente con sensor

3.2 - PROGRAMACIÓN DEL MICROCONTROLADOR

A la hora de programar el sketch que instalaremos en la placa debemos de tener en cuenta el funcionamiento de nuestro sensor. Como bien sabemos nuestro sensor es capaz de medir distancias gracias a los pulsos de ultrasodinos.

Sabiendo la capacidad total del recipiente solo tenemos que restar la cantidad de espacio vacío para saber cuánto líquido hay como bien podemos ver en el siguiente esquema.

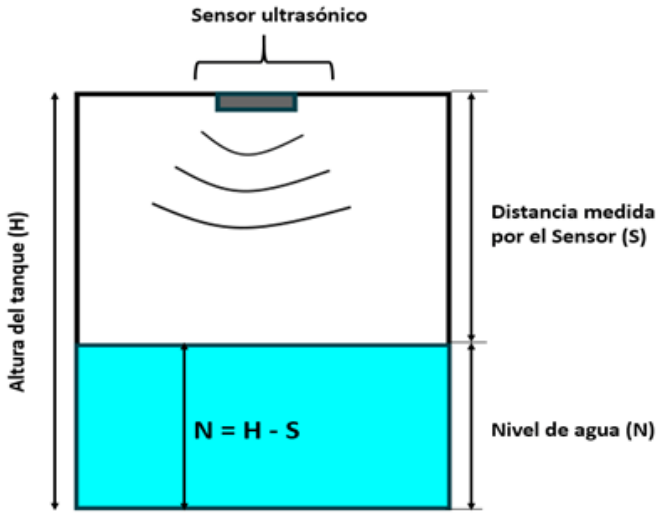

Ilustración 14. Esquema funcionamiento sensor

En todos los sistemas electrónicos se genera un ruido debido al movimiento de los electrones, al propio sensor, la placa de Arduino, etc. Esto se traduce en una variación o fluctuación en las medidas que vamos obteniendo a través del sensor. Para evitar en la medida de lo posible dicho error haremos una media de los valores obtenidos mediante programación. De esta forma también eliminaremos las fluctuaciones debidas al movimiento del recipiente. Dependiendo del número de muestras que utilicemos para obtener la media, podremos tener un valor más preciso o menos preciso. De la misma manera que también influirá en cuánto tarda en estabilizarse la medida cuando se produce un cambio. Por lo tanto, si cogemos muchas muestras, la media será más precisa, pero tardará más tiempo en estabilizarse. Por el contrario, si cogemos un número bajo de muestras, la media será menos precisa, pero veremos reflejados más rápidos los cambios.

En nuestro caso hemos elegido un número alto de muestras ya que conseideramos más importante la precisión del resultado que la rapidez, puesto que un tanque de agua no tiene mucha rapidez en su llenado, nuestro rango será de 100 muestras.

Una vez tenemos claro esto, ya podemos empezar con la programación de nuestro dispositivo.

A continuación podemos ver el sketch comentado:

```
//Incluimos las librerías que utilizaremos.
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>

// Sustituimos por los datos de nuestra red WiFi
const char* ssid = "POCO X3 NFC";//NOMBRE DE LA RED DE NUESTRO WiFi
const char* password = "1196edcf831a";//CONTRASEÑA DE NUESTRA RED WiFi
const String mac = WiFi.macAddress();//MAC DE NUESTRO NODE
String url = "http://esp.goodidea.com.es/agus/index.php";//PHP AL QUE ATACAMOS

//Configuramos los pines del sensor Trigger y Echo
const int PinTrig = 5;//D1
const int PinEcho = 4;//D2

//Constante velocidad sonido en cm/s
const float VelSon = 34000.0;

//Distancia del tanque lleno y vacío
float distancia;
const float distanciaLleno = 4;
const float distanciaVacio = 14.80;
float tiempoTotal;
float tiempo;
float Media;

//Creamos la conexión con nuestra red WiFi
void setup()
{
  Serial.begin(9600);
  WiFi.begin(ssid, password);
  Serial.println(mac);
  while (WiFi.status() != WL_CONNECTED){
    Serial.println("No conectado");
    delay(500);
  }
    delay(500);

  // Ponemos el pin Trig en modo salida
  pinMode(PinTrig, OUTPUT);
  // Ponemos el pin Echo en modo entrada
  pinMode(PinEcho, INPUT);
}
void loop()
{
  HTTPClient http;
  WiFiClient cliente;
//Podemos ver la mac de nuestro tanque
Serial.println("mac "+WiFi.macAddress());
```

```
// Obtenemos la distancia en cm, hay que convertir el tiempo en segudos ya que
está en microsegundos por eso se multiplica por 0.000001
  long medi2=calculaMedia();//Con este método recogemos las 100 muestras
  Serial.println("MEDIA "+medi2);
  float distanciaMedia = medi2* 0.000001 * VelSon / 2.0;//Usamos la fórmula de la
velocidad para calcular la distancia
  Serial.print(distanciaMedia);
  Serial.println("cm");

  //Aquí realizamos los cálculos para obtener el porcentaje de llenado del tanque
  float distanciaLleno = distanciaVacio - distanciaMedia;
  int porcentaje = distanciaLleno * 100 / distanciaVacio;
  Serial.print("porcentaje ");
  Serial.println(porcentaje);

  //Aquí enviamos la información a nuestra base de datos
   String serverPath = url + "?valorDeNivel="+porcentaje+"&mac="+mac;
   http.begin(serverPath.c_str());

   int httpResponseCode = http.GET();

   if (httpResponseCode > 0) {
      Serial.print("HTTP Response code: ");
      Serial.println(httpResponseCode);
      String payload = http.getString();
      Serial.println(payload);
    }
    else {
      Serial.print("Error code: ");
      Serial.println(httpResponseCode);
    }
    // Free resources
    http.end();

  delay(1500);
}
//Aquí haremos la media de las 100 lecturas
long calculaMedia()
{
 Serial.println("Dentro calculaMedia");
 tiempoTotal = 0;
 for(int i=0;i<100;i++){
  // Ponemos el Triiger en estado bajo y esperamos 2 ms
  digitalWrite(PinTrig, LOW);
  delayMicroseconds(2);

  // Ponemos el pin Trigger a estado alto y esperamos 10 ms
  digitalWrite(PinTrig, HIGH);
  delayMicroseconds(10);

  // Comenzamos poniendo el pin Trigger en estado bajo
  digitalWrite(PinTrig, LOW);
  unsigned long tiempo = pulseIn(PinEcho, HIGH);

  // Comenzamos poniendo el pin Trigger en estado bajo
  tiempoTotal=tiempoTotal + tiempo;
  Media=tiempoTotal/100;
  delay(50);
 }
  return Media;
}
```

3.3 - DIAGRAMA DE ENTIDAD DE RELACIÓN

Un diagrama entidad-relación es un tipo de diagrama de flujo que ilustra cómo las "entidades", como personas, objetos o conceptos, se relacionan entre sí dentro de un sistema. En nuestro caso particular lo usaremos para depurar nuestra base de datos.

También conocidos como los ERD o modelos ER, emplean un conjunto definido de símbolos, tales como rectángulos, diamantes, óvalos y líneas de conexión para representar la interconexión de entidades, relaciones y sus atributos. Nuestro diagrama se divide de tres bloques principalmente, son los siguientes:

ENTIDAD LOGIN

Representará en un futuro la tabla de login con sus atributos que serán los siguientes:

- Iduser: Clave fuerte primary key.
- User,pw,nombre,email y id_notify .

Todos los descritos anteriormente son campos (atributos) de entidad no fuerte, serán los encargados recoger información real del usuario.

ENTIDAD NIVELDEAGUA

Representará una futura tabla en la que crearemos la información de los tanques, esta tabla ontiene los siguientes atributos y claves:

- Id: Clave fuerte primary key.
- user_id_user: Clave fuerte foreign key, esta será una relación directa con la tabla login.
- error,valor,altura,nombre y mac.

Estos campos descritos anteriormente son los atributos de la entidad que recogerá la información del tanque, además hay que hacer incapié en el campo mac el cual al no ser una clave fuerte, será un campo único que relacionará la placa (NodeMCU) con la tabla para poder controlar dicha tabla.

ENTIDAD TOKEN

Esta entidad describirá una tabla de tokens la cual servirá para guardar unos tokens de seguridad de acceso hacia la API, tiene los atributos :

- Id: Clave fuerte de la entidad primary key

Hasta y desde, estas son los atributos que describirán un tiempo de duración. La relación del diagrama siempre se usa con un verbo que identifica la relación entre entidades.

En nuestro hemos puesto “crear” que sería que un “login”(usuario) puede crear muchos “nivelesDeAgua”(tanques) lo que significaría la N en la relación.

Por otro lado sería 1 “nivelDeAgua”(tanque) solo puede ser creado por un “login”(usuario) que significaría el 1 que hay descrito en el diagrama anexado a continuación.

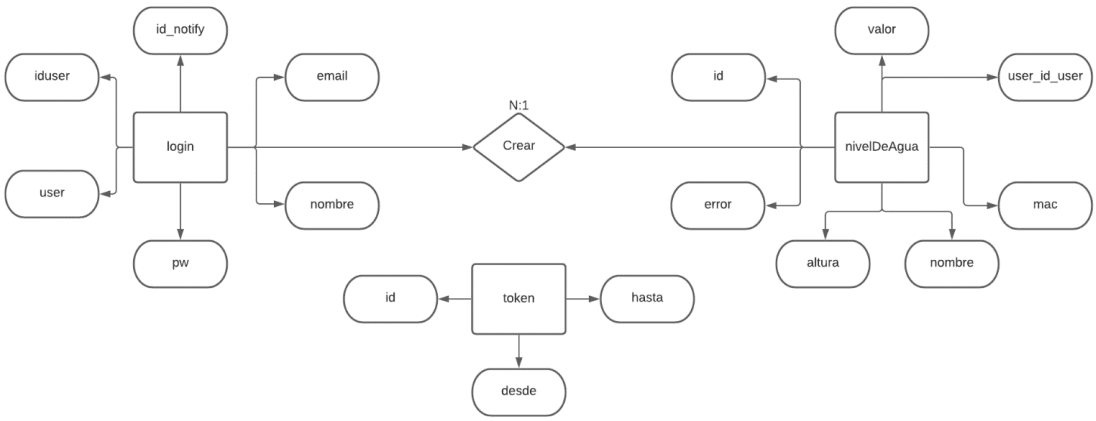

Ilustración 15. Diagrama de entidad de relación

3.4 - VISTA RELACIONAL

Una base de datos relacional es un tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí. Las bases de datos relacionales se basan en el modelo relacional, una forma intuitiva y directa de representar datos en tablas. En una base de datos relacional, cada fila en una tabla es un registro con una ID única, llamada clave. Las columnas de la tabla contienen los atributos de los datos y cada registro suele tener un valor para cada atributo, lo que simplifica la creación de relaciones entre los puntos de datos.

En nuestro caso concreto tenemos 3 tablas y una vista. La relación entre la tabla login y nivelDeAgua está hecha de la siguiente forma. En la tabla nivelDeAgua se ha generado un campo llamado user_id_user que contiene la 'foreign key' de la tabla de login, estos campos deben ser iguales, en nuestro caso son ambos un entero. Con la 'foreign key' conseguimos tener una relación directa (también llamada fuerte) en la cual, en la tabla nivelDeAgua asignamos el identificador único de un usuario, haciendo que estas filas pertenezcan solamente a dicho usuario.

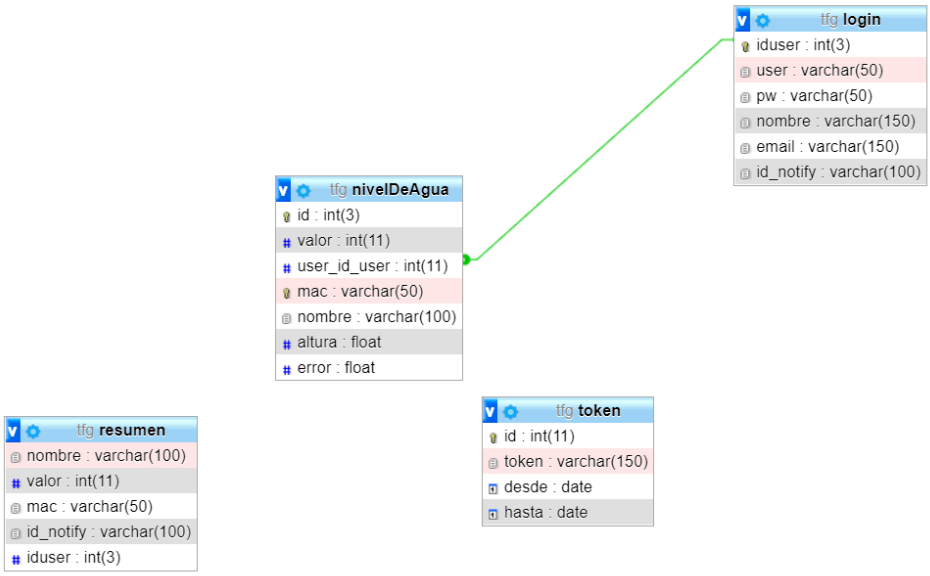

Ilustración 16. Vista relacional

3.5 - TABLAS DE LA BASE DE DATOS

La organización de nuestra base de datos con sus datos es la siguiente:

LOGIN

iduser	user	pw	nombre	email
2	aguss-14@hotmail.es	d4e716e45df06acaa05aa8370e166741	Agustin	aguss-14@hotmail.es
3	jazzwoman@periferianorte.com	39c3c6729651d990c4d72298019979bf	Jazz	jazzwoman@periferianorte.com
4	raulpc93@gmail.com	39c3c6729651d990c4d72298019979bf	Raúl	raulpc93@gmail.com
7	agusebrios@gmail.com	39c3c6729651d990c4d72298019979bf	Aguss	agusebrios@gmail.com

Ilustración 17. Tabla login

- Iduser: Identificador único de la fila, es un primary key autoincremental.
- User: Nombre de usuario de acceso, es un varchar.
- Pw: Contraseña de acceso, es un varchar. Contraseña cifrada con md5.
- Nombre: Nombre real del usuario, es un varchar.
- Email: Correo eléctronico de notificaciones, puede ser distinto al usuario, es un varchar.
- Id_notify: Identificador de One Signal para enviar notificaciones, es un varchar.

NIVELDEAGUA

id	valor	user_id_user	mac	nombre	altura	error
9	99	3	AA:BB:CC:DD:EE	Prueba tanque	15	2.25
10	99	2	40:F5:20:29:34:C4	Jardín	12	1.2
11	82	2	11:22:33:44	Patio	95	2

Ilustración 18. Tabla nivel de agua

- Id: Identificador único de la fila, es un primary key autoincremental.
- Valor: Es el porcentaje de llenado de cada tanque, es un int.
- User_id_user: Es la foreing key de login que hace referencia a iduser. Es un int.
- Mac: Representa al identificador único de cada sensor. Es un varchar.
- Nombre: Es el nombre de cada tanque, es un varchar.
- Altura: Es la altura en cm de cada tanque, es un float.
- Error: Es el margen de error en cm de cada tanque, es un float.

TOKEN

id	token	desde	hasta
1	63525ce9d52a709b241d62da9e47469a	2020-10-25	2050-10-26

Ilustración 19. Tabla token

- Id: Identificador único de la fila, es un primary key autoincremental.
- Token: Sirve para permitir el acceso a nuestra API a una aplicación externa.
- Desde: Fecha desde la que está disponible el token.
- Hasta: Fecha de caducidad del token.

RESUMEN

nombre	valor	mac	id_notify	iduser
Prueba tanque	99	AA:BB:CC:DD:EE	0	3
Jardín	99	40:F5:20:29:34:C4		2
Patio	82	11:22:33:44		2

Ilustración 20. Tabla resumen

A parte de estas tablas tenemos una vista resumen llamada resumen que nos permite relacionar diferentes tablas y sus campos correspondientes entre ellos.

En nuestro caso nos ha servido para relacionar los tanques, sus valores y macs entre la id_notify y el iduser, esto nos permite enviar notificaciones del estado de llenado al usuario a través de la APP.

3.6- SERVIDOR DE ACCESO A LA BASE DE DATOS

La interfaz de programación de aplicaciones, conocida también por la sigla API, en inglés, APP lication programming interface, es un conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Con esta API realizamos todos los 'endpoints' accesibles a través de peticiones POST o GET en el servidor WEB, esta API, será la encargada de recibir la información, realizar un control de dicha información, consultar a la base de datos y obtener un resultado, ya sea de inserción de actualización o de selección, y devolver dicha información a un cliente como respuesta http.

La organización de nuestra API se ha hecho de la siguiente forma:

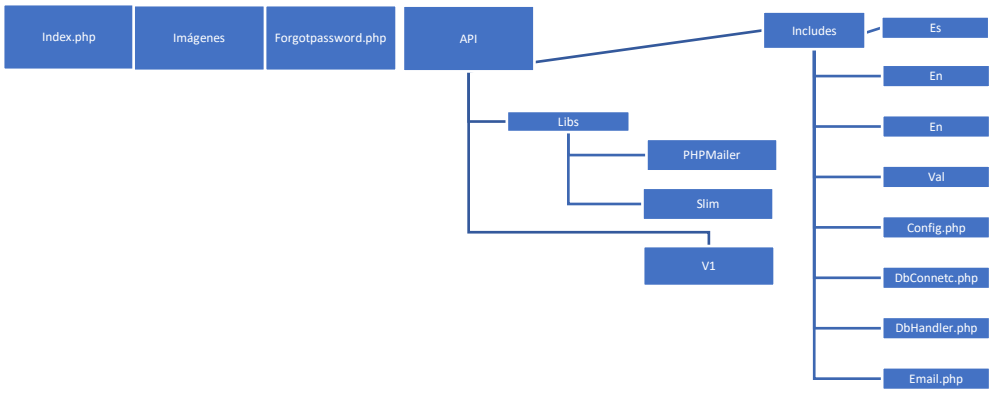

Ilustración 21. Estructura api

API: Carpeta raíz de la API, contiene todos los archivos para su funcionamiento correcto.

- Includes: Encontramos todas las clases con sus correspondientes métodos además de otros archivos helpers.
    - En: Encontramos la traducción del mail que se envía para recuperar la contraseña en inglés.
    - Es: Encontramos la traducción del mail que se envía para recuperar la contraseña en español.
    - Val : Encontramos la traducción del mail que se envía para recuperar la contraseña en valenciano.
    - Config.php: Variables de configuración de acceso a bases de datos.
    - DbConnect.php: Conexión a base de datos.
    - DbHandler.php: Contiene todas las funciones del manejo de la base de datos (insert, select, delete,update).
    - Email.php: Es una clase para poder enviar emails.
- Libs: Encontramos todas las librerías que usamos.
    - PHPMailer: Encargada de enviar los corros de recuperación de contraseña.
    - Slim: encargada de hacer accesibles los end points que están escritos dentro de la carpeta V1
- V1: Aquí encontramos todos los end points accesibles de la API.
    - Index.php: Contiene todos los 'endpoints' de la aplicación. Los 'endpoins' son un extremo de un canal de comunicación. Cuando una API interactúa con otro sistema, los puntos de contacto de esta comunicación se consideran puntos finales. Para las API, un punto final puede incluir una URL de un servidor o servicio. Cada punto final es la ubicación desde la que las API pueden acceder a los recursos que necesitan para llevar a cabo su función.

Imágenes: Encontramos la imagen que usaremos en el fichero (Forgotpassword.php) de a continuación.

Forgotpassword.php: Este archivo php contiene el formulario para cambiar la contraseña del usuario.

Index.php: Este es el archivo en el cual se conecta el sensor, le envía la información obtenida a la base de datos y además si dicho nivel se encuentra en un rango específico, nos enviará una notificación.

3.7- APLICACIÓN TELÉFONO MÓVIL

ESTRUCTURA DE LA APLICACIÓN

Como habíamos dicho previamente para la programación de la aplicación se ha usado una composición de Framework7 con su versión CORE y JavaScript.

Una aplicación Framework7 posee de una estructura cerrada para su funcionamiento que se compone de la siguiente forma:

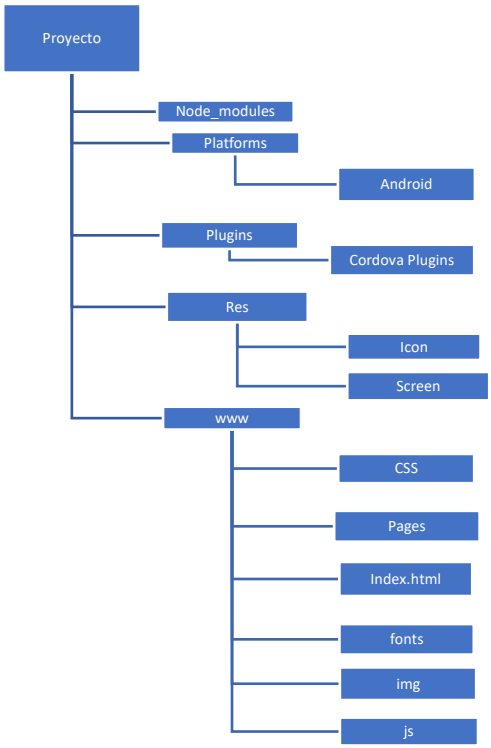

Ilustración 22. Estructura proyecto

PROYECTO: Carpeta raíz del proyecto contiene todos los archivos necesarios para su funcionamiento.

- Node_modules: Carpeta que contiene todos los archivos de NodeJs necesarios para que funcione perfectamente cordova y sus plugins, son dependencias.
- Plugins: Contiene todos los archivos de los plugins que se han instalado, llevan el código nativo del plugin (Puede ser editado por uno mismo si hace falta).
- Res: Contiene los resources que necesita la aplicación normalmente iconos y splashactivities.
- www: Es la estructura principal de la aplicación contiene todos los archivos de programación necesarios.
    - Css: Contiene los CSS de estilos necesarios de la aplicación.
    - Pages: Contiene todas las actividades de la aplicación escritos como templates en código HTML y JavaScript.
    - Index.HTML: Archivo raíz de la aplicación contiene el contendor principal del framework7, en este archivo, contiene todos los headers y scripts además del inicializador de cordova. En su body contiene el view-main de Framework7 donde el framework es capaz de cargar todos los templates usando peticiones xhr request internas en su localhost.
    - Fonts: Contiene las fuentes usadas en la aplicación en formato ttf.
    - Img: Contiene las imágenes que se usarán en la aplicación, pueden ser archivos JPG, PNG incluso SVG.
    - Js: Contiene todas las librerías y archivos necesarios de javascript, contiene los bundle de framework7, además tiene el APP .js archivo de inicialización de la clase de framework7 y generamos la variable global APP que contiene todo el control del framework dentro de la aplicación. También contiene la inicialización de DOM7 (manejador de templates).

FUNCIONAMIENTO DE LA APLICACIÓN

Nuestra APP se compone de diferentes vistas llamadas en Android Activity que han sido programadas de forma independiente unas de otras ya que todas las vistas se cargan sobre el mismo archivo (index.HTML). Para poder realizar esto hemos usado la opción que nos proporciona DOM7.

Cada vista tiene su propia estructura siendo estas programadas específicamente para su función, estas vistas contienen un elemento global APP que nos permite usar las funciones de Framework7 en cada una de ellas para así tener acceso a las ventajas que este nos proporciona como acceso a anclas, a espacios de memoría (JSON DATA) y a los propios métodos de Framework7.

Este framework nos proporciona una funcionalidad muy buena que son los eventos y triggers entre cargas de páginas (vistas) que nos permite de forma muy sencilla adaptar nuestra vista (carga de datos) cuando estos son necesarios, podemos hacer un PageInit y mostrar una información y obtener un pageBeforeInit para cargar dicha información, de esta manera, podemos realizar una carga optimizada de la información siendo esta cargada de forma backend sin que el usuario se entere y haciendo una aplicación mucho mas amigable y rápida.

Este Framework también nos permite de forma muy sencilla controlar los componentes nativos (buttons, badges, lists, inputs, cards …).

Gracias al CSS, podemos realizar vistas según las normas de Android, en nuestro caso, hemos optado por usar la vista Material Desing (Android) y seguir sus pautas. Google ha creado estas pautas, ya que con ellas, obtenemos la mejor usabilidad en nuestra aplicación y es lo que ellos nos recomiendan para que nuestra aplicación sea bien vista en todos los dispositivos disponibles en el mercado para así obtener la mejor compatibilidad posible.

4 - MANUALES

4.1 - INSTALACIÓN

En esta sección, vamos a explicar los pasos que hay que seguir para instalar todos los componentes del sistema. En resumen, los pasos serán estos:

INSTALACIÓN CORDOVA

Cordova funciona a través del command line de Windows/Linux o Mac. Para crear el proyecto, debemos realizar lo siguiente:

Ubicarnos en el directorio donde crearemos el proyecto usando el terminal, para ello debemos usar el comando 'cd'.

Una vez dentro de él, debemos usar este comando: "cordova create <titulo> <paquete>"

El titulo es como conoceremos la carpeta del proyecto. El paquete es el identificador único de la aplicación por convenio suele ser del estilo "com.desarrollador.titulo".

Una vez tenemos esto, se pone en marcha la creación del proyecto y es cuando entra en función NodeJS que habíamos instalado previamente, NodeJS en un gestor de paquetes y muchas otras cosas mas, que hará que descargue e instale todos los módulos necesarios para su funcionamiento, se crearán dentro de node_modules.

Una vez el proyecto esta creado y tenemos cordova habilitado, instalaremos los plugins, para ello usaremos dicho comando: "cordova plugin install <paquete plugin>".

Por ejemplo: "cordova plugin intall barcode-scanner"

Empezará a instalar el plugin, para ello se conectará al repositorio estipulado, normalmenete github, lo descargará y lo ubicará en la carpeta de plugins además de añadirlo en el archivo de configuración.

Una vez hecho todo esto, podemos testar nuestra aplicación, para ello deberemos compilar la aplicación generando así todos los archivos necesarios de Android, en nuestro caso para ello deberemos instalar la plataforma requerida, usaremos lo siguiente: "cordova platform add android".

Se instalarán los plugins en su versión Android, y se generarán las entradas y salidas disponibles para Android, además del archivo Cordova para Android. Una vez hecho esto, deberemos realizar lo siguiente: “cordova prepare android”.

Con este comando hacemos que las últimas actualizaciones realizadas en el archivo HTML se transformen a código Java android (Se copian los archivos HTML en su carpeta y se llamara file_assets) donde el webview de android podrá acceder a ellos y mostrarlos.

INSTALACIÓN ANDROID STUDIO

Con Android Studio realizaremos la compilación final de la aplicación para generar el archivo APK. Para ello, primero debemos tener instalado Android Studio IDE y tener descargado una versión de Android para realizar su compilación, para ello deberemos instalar desde el gestor de Android, en nuestro caso hemos compilado usando Android 9 API 28.

Una vez estos requerimientos hechos, podremos importar nuestro proyecto dentro de Android Studio, para ello, deberemos importar el proyecto que hay dentro de PROYECTO PRINCIPAL -> PLATFORMS -> ANDROID, normalmente Android Studio reconoce automáticamente que es un proyecto válido, ya que posee archivos Gardle ya generados por Cordova.

Una vez se ha realizado la importación del proyecto, podremos conectar nuestro dispositivo Android (usando su cable USB y habilitando la depuración USB en el mismo móvil, para ello, debemos habilitar también las opciones de desarrollador) y así podremos instalar directamente el APK en el móvil y realizar los test pertienentes.

Para poder depurar nuestra aplicación debemos usar la herramienta de desarrollador de Google depurando así los webviews disponibles.

Finalmente cuando hemos testado nuestra aplicación podemos ver si tenemos comunicación con nuestros ‘endpoints’ disponibles en la nube, para ello debemos tener habilitado la opción de WiFi o 4G en el dispositivo.

Si tenemos problemas de comunicación, debemos ayudarnos del depurador de Chrome, si seguimos teniendo problemas, puede que nuestro error se deba a un fallo en la API, para ello nos ayudaremos del software POSTMAN, que es un software con versión gratuita para testar APIs.

Después de realizar todo esto, podemos comprobar el funcionamiento y la programación que hemos desempeñado, siendo así posible encontrar errores de

funcionamiento y/o compatibilidad con algunos dispositivos, siendo necesaria su restructuración de código/diseño.

Si el funcionamiento es bueno en diferentes dispositivos (sobre todo diferentes versiones de sistema operativo o tamaño de pantalla) podemos dar por finalizada nuestra APP, ya que asi hemos comprobado que tenemos acceso a la información almacenada en la base de datos que ha generado nuestro sensor, además de otros métodos necesarios.

INSTALACIÓN TRADUCCIONES

Para realizar las traducciones de nuestra aplicación móvil hemos usado la librería i18next, esta librería tiene una integración muy sencilla con la librería jQuery que nos va a permitir añadir traducciones a nuestros textos de una forma muy sencilla. Cuenta con sus propios *hooks* algo que viene a ser un añadido ahora que los componentes funcionales son la tendencia en el desarrollo con esta librería.

A continuación será necesario configurar la librería para poder emplearla dentro de nuestros componentes. Esto es recomendable hacerlo en un archivo aparte que importaremos en el archivo index.js de nuestro proyecto.

Inicializamos i18next en el archivo APP .js dentro de la carpeta js: Con el método initializeI18n() conseguimos poder inicializar el uso uso del i18n back end para poder relizar una petición XHR request para cargar de forma dinámica las traducciones de nuestra aplicación que están escritas en varios archivo JSON (uno por idioma) alojados en la carpeta i18n.

```
function initializeI18n() {
    var language = app.utils.i18n.getLanguage();
    console.log(language);
    i18next
        .use(i18nextXHRBackend)
        .init({
        lng: language.lang,
        fallbackLng: 'en',
        whitelist: ['en', 'es','val'],
        nonExplicitWhitelist: true,
        preload: ['en', 'es','val'],
        backend: {
            loadPath: 'js/i18n/{{lng}}.json'
        }
    },function() {
        app.utils.i18n.setLanguage(language);
    });
}
```

Ilustración 23. Inicialización i18n

La configuración inicial de la librería la haremos en el archivo utils.js dentro de la carpeta js:

```
Framework7.utils.i18n = {

    getLanguage: function() {
        var language = localStorage.getItem('Nectar_Language') ? JSON.parse(localStorage.getItem('Nectar_Language')) : window.config.i18n;
        return language;
    },

    setLanguage: function(language) {
        var language = language ? language : window.config.i18n;

        i18next.changeLanguage(language.lang, function() {
            localStorage.setItem('Nectar_Language', JSON.stringify(language));

            console.log('lang utils '+language.lang);
            app.$('html').attr('lang', language.lang);
            app.$('html').attr('dir', language.dir);
            console.log(i18next);
            window.localize = locI18next.init(i18next);
            window.localize('body');

        });
    }

};
```

Ilustración 24. Configuración i18n

INSTALACIÓN DEL DRIVER CH340 EN WINDOWS:

El driver CH340 es el software requerido para poder usar dicho circuito integrado, nos ayudará para poder usar cualquier modelo de Arduino genérico.

Primero debemos descomprimir el archivo ZIP en una carpeta, encontraremos una carpeta llamada CH341SER, ejecutar el programa de instalación o “setup.exe”, saldrá una ventana a la que solamente debemos de darle a INSTALL, una vez terminada la instalación conectamos el Arduino al PC y esperamos a que Windows detecte e instale los drivers para el dispositivo conectado.Después de todo esto debemos seleccionar el puerto en el programa.

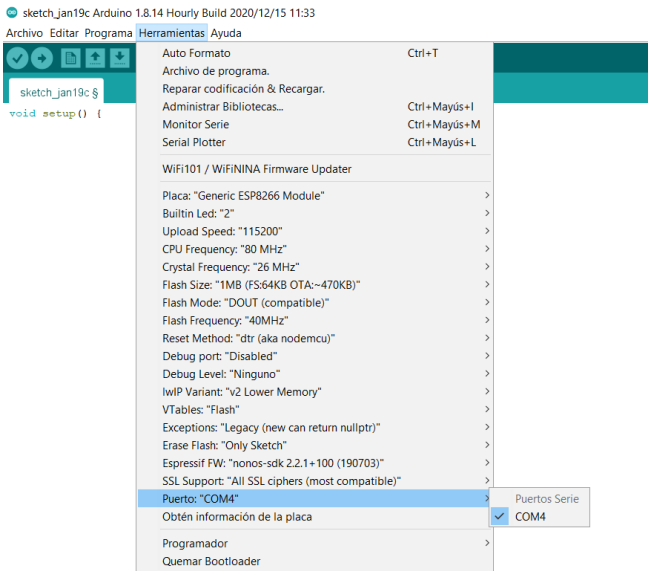

Ilustración 25. Selección de puerto

Debemos configurar el IDE estándar de Arduino para poder programar el ESP8266. Para poder programar las placas de desarrollo basadas en el ESP8266 simplemente tendremos que configurar la URL del paquete para que podamos agregarlas al gestor de placas del IDE de Arduino.

Para ello accedemos al menú de configuración y en "Gestor de URLs adicionales de tarjeta" hacemos click en el pequeño botón de la derecha.

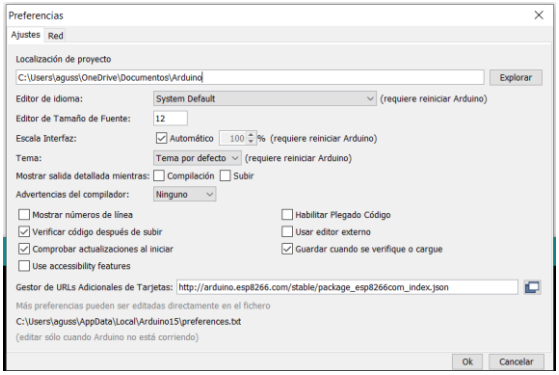

Ilustración 26. Gestor de url adicionales

En la ventana que aparece, añadimos esta la siguiente URL.

http://arduino.esp8266.com/stable/package_esp8266com_index.json

Ahora entramos en el gestor de tarjetas del IDE de Arduino y buscamos el paquete de placas de desarrollo basadas en el ESP8266 y lo instalamos.

Ilustración 27. Selección de paquete

Ya tenemos disponibles las placas de desarrollo basadas en el ESP8266 para programarlas con el IDE de Arduino.

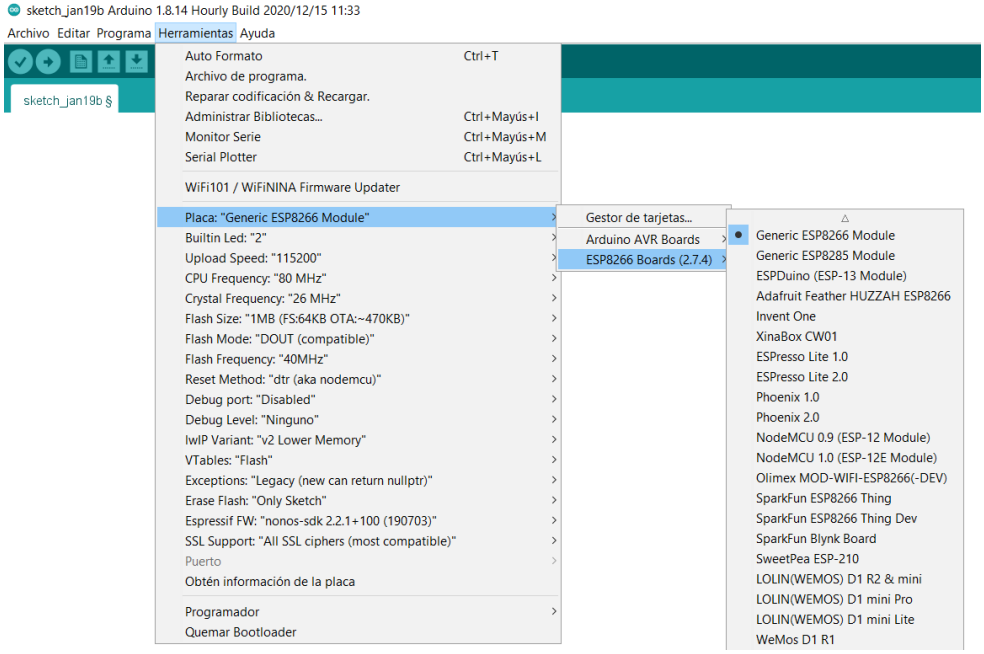

Ilustración 28. Selección de placa de desarrollo

CARGA DEL SKETCH EN LA PLACA

Lo primero que tenemos que hacer es ejecuta el software Arduino haciendo doble-click en el ícono de Arduino.

Luego tenemos que cargar el sketch, para ello hacemos click sobre Archivo->Abrir…->Sensor.ino

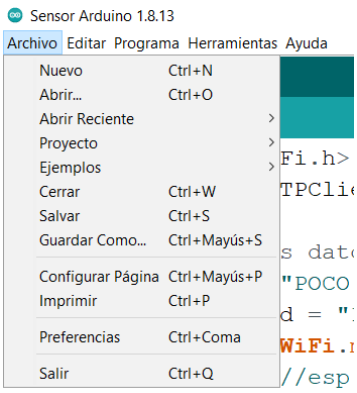

Ilustración 29. Selección archivo

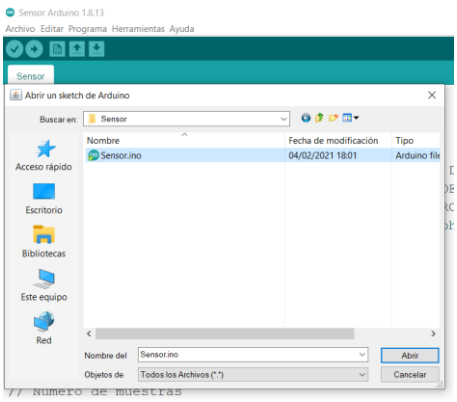

Ilustración 30. Selección sensor

Antes de ejecutar el sketch tenemos que poner nuestros datos de la red WiFi que utilizaremos para realizar la comunicación con la base de datos.

Ilustración 31. Datos red WiFi

Compilamos el sketch. Para hacerlo, presionamos el botón 'Play'.

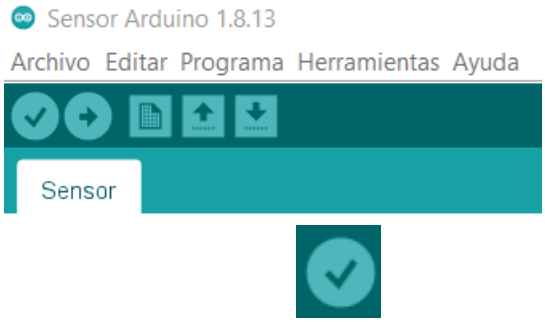

Ilustración 32. Botón play

Cargar el sketch. Presiona el botón 'Upload' para cargar el sketch a la placa Arduino.

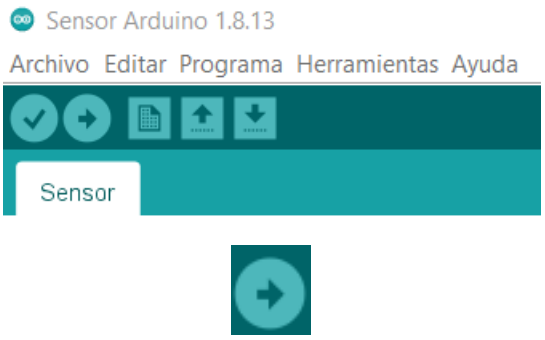

Ilustración 33. Botón upload

Una vez el sketch se haya cargado por completo en la placa, vemos un texto aparecer en la ventana inferior del software de Arduino. El texto dirá "Done Uploading.

Ahora que se ha cargado el sketch, la placa comenzará a ejecutar las instrucciones escritas en el sketch quedando así lista para su funcionamiento.

4.2 - USUARIO

INSTALACIÓN Y FUNCIONAMIENTO DE LA APP

Para poder instalar nuestra aplicación en el móvil únicamente tendremos que acceder a la 'Play Store' si tenemos un dispositivo Android y a la 'APP le Store' si tenemos un dispositivo APP le.

Ilustración 34. Play store y apple store

En nuestro caso no hemos realizado las peticiones necesarias para poder alojar nuestra aplicación en estas plataformas, por lo tanto lo que debemos de hacer es descargarnos el archivo APK de la aplicación, dicho archivo se instala desde el propio teléfono.

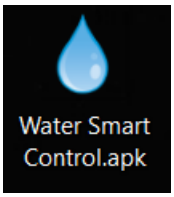

Ilustración 35. APK APP

Para poder cargar un tanque en nuestra APP debemos de escanear el código QR asociado al tanque, dicho código QR hace referencia a la mac de la placa instalada en cada tanque. Con esto conseguimos que cada código QR sea único.

Ilustración 36. Botón QR y QR

Cuando escaneasmos el código QR se nos abrirá una pantalla en la que deberemos seleccionar el nombre del tanque, la altura (en cm) y el margen de seguridad que queremos adptar en dicho tanque.

Ilustración 37. Crear tanque

Una vez realizado esto ya tendremos todos nuestros tanque en la pantalla principal de nuestra APP para poder tenerlos controlados.

Ilustración 38. Administración tanques

INTERFAZ DE USARIO DE LA APLICACIÓN

INICIAR SESIÓN

En esta página podemos iniciar sesión con una cuenta de usuario previamente creada. La página se compone de una 'card' con 2 'inputs' rellenables con usuario y contraseña. Más abajo tenemos un 'checkbox' que si marcamos la aplicación recordará nuestras credenciales para no tener que ponerlas nuevamente en el próximo inicio de sesión. Seguidamente tenemos 2 botones, en la parte izquiera el botón para acceder y en la derecha el botón para resetear nuestra contraseña en caso de no recordarla. Por último tenemos un 'link' que nos pregunta si no disponemos de una cuenta creada para que en caso de no tenerla podamos acceder a la página de 'crear usuario'. En la parte inferior de la página de 'inicio de sesión' encontramos un 'toast' que se activa cuando no introducimos las credenciales de inicio de sesión correctamente.

Ilustración 40. Inicio de sesión

Ilustración 39. Toast contraseña incorrecta

RESETEAR CONTRASEÑA

En esta página podremos resetear nuestra contraseña en caso de que la hayamos olvidado. En la parte central se encuentra una 'card' con un 'input' en el que deberemos de introducir el correo al cuál queremos que la aplicación nos envíe el email para poder cambiar nuestra contraseña actual. Solo podremos recuperar nuestra contraseña en caso de que ya seamos usuario y hayamos verificado nuestro correo como bien dice el propio texto.

Más abajo dentro de la 'card' encontramos el botón de 'reseteo de contraseña' y finalmente tenemos un texto para poder regresar al inicio de sesión. Por último cuando pulsamos el botón, si todo ha sucedidao correctamente, nos saltará un 'alert' para indicarnos que debemos de revisar nuestra bandeja de entrada para cambiar la contraseña.

Ilustración 42. Reseteo contraseña

Ilustración 41. Alert bandeja de entrada

CREAR USUARIO

En esta página podremos crear una cuenta de usuario nueva. La página se compone de un 'navbar' en la parte superior con el cuál podremos retroceder a la página de iniciar sesión. En la parte central tenemos una 'card' con 3 'inputs' que deberemos rellenar con nuestra información, nombre, email y contraseña. También tenemos un 'checkbox' que deberemos seleccionar para aceptar los términos y condiones y así poder completar el registro, si no lo seleccionamos nos saltará un 'alert' para indicarnos que es imprescindible seleccionarlo. Más abajo tenemos un botón que pulsaremos para confirmar el registro. Por último tenemos un 'text button' que nos pregunta si ya tenemos una cuenta creada para que en caso de ya tenerla podamos retroceder a la página de iniciar sesión. Para añadir mucha más seguridad a los usuarios hemos creado una condición a la hora de crear nuestra contraseña, dichas condiciones son. La contraseña debe incluir 8 caracteres como mínimo, debe contener letras mayúsculas y minúsculas y al menos un carácter especial como puede ser "*". Cuando intentamos

introducir una contraseña que no cumple dichas condiciones nos alertará con un ‘dialog’ para informarnos.

Ilustración 45. Condición contraseña

Ilustración 44. Crear usuario

Ilustración 43. Términos y condiciones

INICIO

Esta es la página principal de la aplicación, en ella podemos encontrar lo siguiente. En la parte superior izquierda en el ‘navbar’encontramos el botón de desconectarse y en la parte derecha encontramos el botón de ajustes. Más bajo encontramos un mensaje de bienvenida personalizado con nuestro nombre de usuario. Seguidamente encontramos un texto en el que explica como podemos añadir los tanques de agua, el texto dice “¡Desde aquí puede vincular su tanque! ¡Solo necesito escanear el Qr en el recipiente para configurarme!”. Más abajo encontramos el ‘body’ en los que a través de unas ‘cards’ encontramos los tanques que tenemos creados, cada ‘card’ se compone del nombre del tanque, el porcentaje de agua que tiene ocupado y una barra de progreso para ver el progreso de una forma más visual. En la parte inferior tenemos el botón ‘Qr’, aprentando el botón accedemos a la cámara de nuestro teléfono móvil para escanear los códigos puestos en los diferentes tanques. En la parte superior inzquiera en el ‘navbar’ de la APP podemos encontrar el botón de ‘desconcetarse’, con dicho botón podemos desconectarnos de nuestra sesión para iniciar sesión con otra cuenta diferente. Cuando pulsamos el botón nos aparece un ‘confirm dialog’ con el texto ‘¿Desea desconectarse?’, si pulsamos ‘OK’ volveremos a la página de ‘Inicio de sesión’.

Por último haciendo un ‘pull to refresh’ podemos refrescar la página para actualizar la información de cada tanque en caso de que hayan ocurrido cambios, y si no hay ningún tanque creado aparecerá un ‘alert’ para avisarnos que no hay ningún tanque creado.

Ilustración 46. Alert tanques

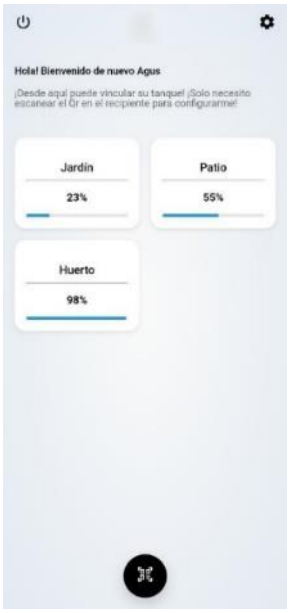

Ilustración 48. Inicio

Ilustración 47. Alert desconectarse

TANQUE

En esta página podemos ver la información principal que tiene cada tanque. En la parte superior izquierda en el ‘navbar’ de la página ‘tanque’ encontramos un ‘título’ con el nombre del tanque y si pulsamos en el ‘back button’ regresaremos a la página anterior. En la parte derecha tenemos un ‘action button’ con la opción de eliminar tanque y en el ‘body’ de la página encontramos una simulación del tanque llenándose hasta llegar a su porcentaje de ocupación, dicho porcentaje lo encontramos justo en el centro. En la parte inferior de esta página encontramos un ‘floating button’ de ‘vaciar’ si pulsamos en él accionaremos una simulación del tanque vaciándose hasta llegar al 0%, cuando esto ocurra nos saltará un ‘alert’ con el texto de “Tanque vaciado”. En la parte central de la página ‘tanque’ encontramos un ‘confirm dialog’ que se activa cuando queremos eliminar un tanque, dicho ‘confirm dialog’ es un método de seguridad añadido para no eliminar un tanque de manera accidental. En la parte superior derecha en el ‘navbar’ de la página ‘tanque’ encontramos un ‘action button’, si pulsamos en él nos aparecerá un ‘action sheet’ con las opciones de eliminar o cancelar.

Si pulsamos en ‘Eliminar’, eliminaremos el tanque seleccionado. Para revertir esta acción debemos pulsar en ‘Cancelar’.

Ilustración 52. Tanque

Ilustración 51. Confirm dialog vaciar

Ilustración 50. Alert vaciado

Ilustración 49. Action sheet cancelar

AJUSTES

En esta página podremos relizar los ajustes, en nuestro caso son 2 cambios en nuestra cuenta. La página se compone de un ‘navbar’ en la parte superior en el cuál podremos retroceder a la página de ‘inicio’. Más abajo tenemos un ‘list view’ con las 2 opciones disponibles. Abajo tenemos un ‘list item with action’ en el que podemos cambiar la contraseña si previamente hemos verificado la cuenta. Por último tenemos un ‘list item with action’ en el que podemos cambiar el idioma de nuestra aplicación, al pulsar en el ‘action button’ se abre una lista en la que podremos seleccionar entre 3 idiomas, español, inglés y valenciano.

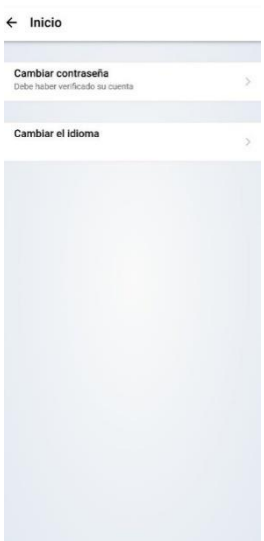

Ilustración 53. Ajustes

Ilustración 54. List item idiomas

CAMBIAR CONTRASEÑA

En esta página podremos cambiar nuestra contraseña. La página se compone de un ‘navbar’ en el cuál podremos retrodecer a nuestra página de inicio, después mediante una ‘card’ tenemos 2 campos que deberemos rellenar con nuestra nueva contraseña y por último un botón que pulsaremos para confirmar el cambio de contraseña. En caso de que ambas contraseñas no coincidan nos saltará un ‘alert’ con el siguiente mensaje “Las contraseñas no coinciden” para indicar que hemos escrito la nueva contraseña de manera errónea, esto lo hacemos para asegurarnos que el usuario no se equivoca al introducir la nueva contraseña. En caso de que las contraseñas coincidan nos saltará ‘alert’ con el mensaje “Guardado correctamente” para confirmar que el cambio se ha efectuado, acto seguido cuando le demos a ‘OK’ la aplicación nos redirigirá a la página de “Inicio de sesión” para que volvamos a acceder a nuetra cuenta con la nueva contraseña.

Ilustración 57. Alert guardado

Ilustración 56. Alert contraseña

Ilustración 55. Cambiar contraseña

NOTIFICACIONES POP UP

La aplicación cuenta con notificaciones del tipo ‘pop up’, estas notificaciones las usamos para informar a los usuarios del estado de llenado de cada tanque. Pulsando sobre la notificación nos remitirá directamente a la vista del tanque correspondiente a dicha notificación.
Tenemos 3 notificaciones diferentes, cuando el tanque llega a la mitad de su capacidad (50%), cuando el tanque está casi lleno rebasando el 80% de su capacidad y por último cuando el tanque está a punto de desbordarse a más del 90% de su capacidad.

Ilustración 60. Pop up mitad

Ilustración 59. Pop up casi lleno

Ilustración 58. Pop up desbordarse

RECUPERACIÓN DE CONTRASEÑA

Cuando ingresamos en el apartado de recuperación de contraseña, la aplicación nos pide que indiquemos un correo electrónico de recuperación. Dicho correo electrónico es dónde se enviará un email para poder modificar nuestra contraseña vía web.

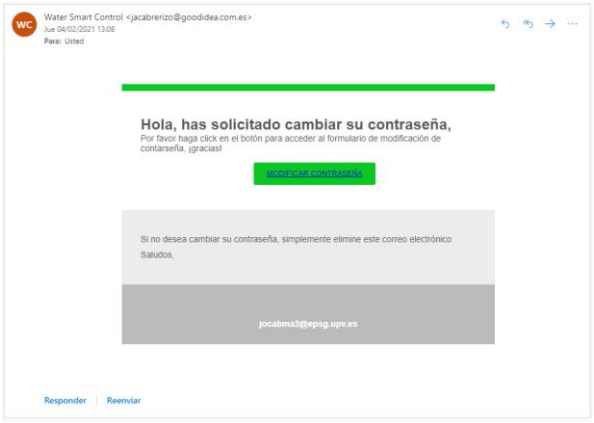

Ilustración 62. Correo cambio contraseña

Ilustración 61. Cambio contraseña web

En este apartado también debemos de cumplir con los requisitos mínimos para nuestra nueva contraseña, de no ser así nos saltará una alerta informándonos de ello.

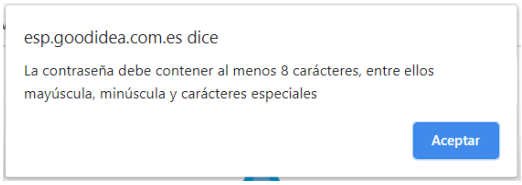

Ilustración 63. Alert condiciones contraseña

Tenemos 2 campos para reforzar la seguridad y asegurar que nuestra nueva contraseña ha sido introducida correctamente, si ambos campos no coinciden también saltará un alert notificándolo.

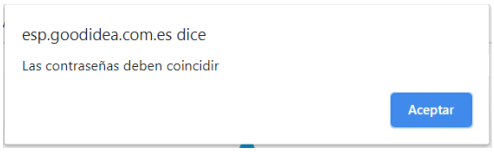

Ilustración 64. Alert contraseñas no coinciden

En caso de que todo haya sucedido correctamente nos saltará un alert don dicha información.

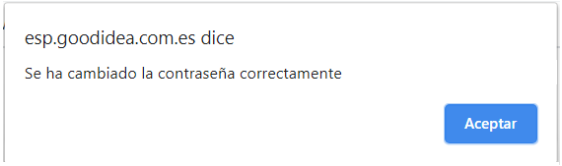

Ilustración 65. Alert contraseña correcta

5 - CONCLUSIÓN

En esta memoria hemos presentado el desarrollo de un sistema de vigilancia del nivel de llenado de tanques de agua. Este sistema está compuesto por un microcontrolador con un sensor de ultrasonidos que envía información a un servidor que la almacena en una base de datos. Por otro lado, hay una aplicación para teléfono móvil que, a partir de los datos del servidor, informa al usuario del estado del tanque. La intención de este proyecto ha sido facilitar dicha vigilancia de tanques de agua en un contexto como el actual donde este tipo de sistemas son más requeridos. En nuestra opinión este objetivo ha sido plenamente cumplido.

Igualmente, los objetivos de aprendizaje de nuevas herramientas y tecnologías, así como la aplicación de saberes adquiridos en nuestra titulación han sido también satisfechos; al tiempo que hemos podido también realizar las prácticas en la empresa.

Nos gustaría proponer las siguientes ampliaciones a este trabajo:

- La incorporación de una compuerta en cada tanque manejada a través de nuestra aplicación con la que podríamos decidir en todo momento cuando esta compuerta está cerrada o cuando se abre. Esta compuerta podría abrise para vaciar parte del agua que está en el tanque con la finalidad de que no rebose.

- Conectar la compuerta a un sistema de riego para que cuando nuestro tanque se encuentre al 90% de su capacidad se active dicha compuerta y riegue así nuestro huerto o jardín.

- Implementar sensores de humedad o sensores de luz, para regar dependiendo de la humedad del ambiente o de la cantidad de sol que haya.

- Incluir un apartado de temporización en nuestra aplicación, para que todas estas opciones que hemos comentado anteriormente pueden ser activadas en los intervalos de tiempo que se adapten a nuestras necesidades.

6 - BIBLIOGRAFÍA

[1] Página oficial de Arduino [https://arduino-esp8266.readthedocs.io/en/latest/] (Consulta: Octubre de 2020)

[2] Página oficial de Leantec [https://leantec.es/wp-content/uploads/2019/06/Leantec.ES-HC-SR04.pdf] (Consulta: Octubre de 2020)

[3] Página oficial de Farnell [https://www.farnell.com/datasheets/410427.pdf] (Consulta: Octubre de 2020)

[4] Página oficial de MariaDB [https://mariadb.com/kb/en/documentation/] (Consulta: Noviembre de 2020)

[5] Página oficial de Apache Cordova [https://cordova.apache.org/docs/en/10.x/] (Consulta: Noviembre de 2020)

[6] Página oficial de Git Hub [https://github.com/phonegap/phonegap-plugin-barcodescanner] [https://github.com/ionic-team/cordova-plugin-ionic-webview#installation-instructions] [https://github.com/sortdinc/cordova-plugin-wkwebview-ionic-xhr] (Consulta: Noviembre de 2020)

[7] Página oficial de Framework7 [https://framework7.io/docs/] (Consulta: Noviembre de 2020)

[8] Página oficial de Android [https://developer.android.com/docs] (Consulta: Diciembre de 2020)

[9] Página oficial de One Signal [https://documentation.onesignal.com/docs] (Consulta: Diciembre de 2020)

[10] Página oficial de i18next [https://www.i18next.com/] (Consulta: Diciembre de 2020)