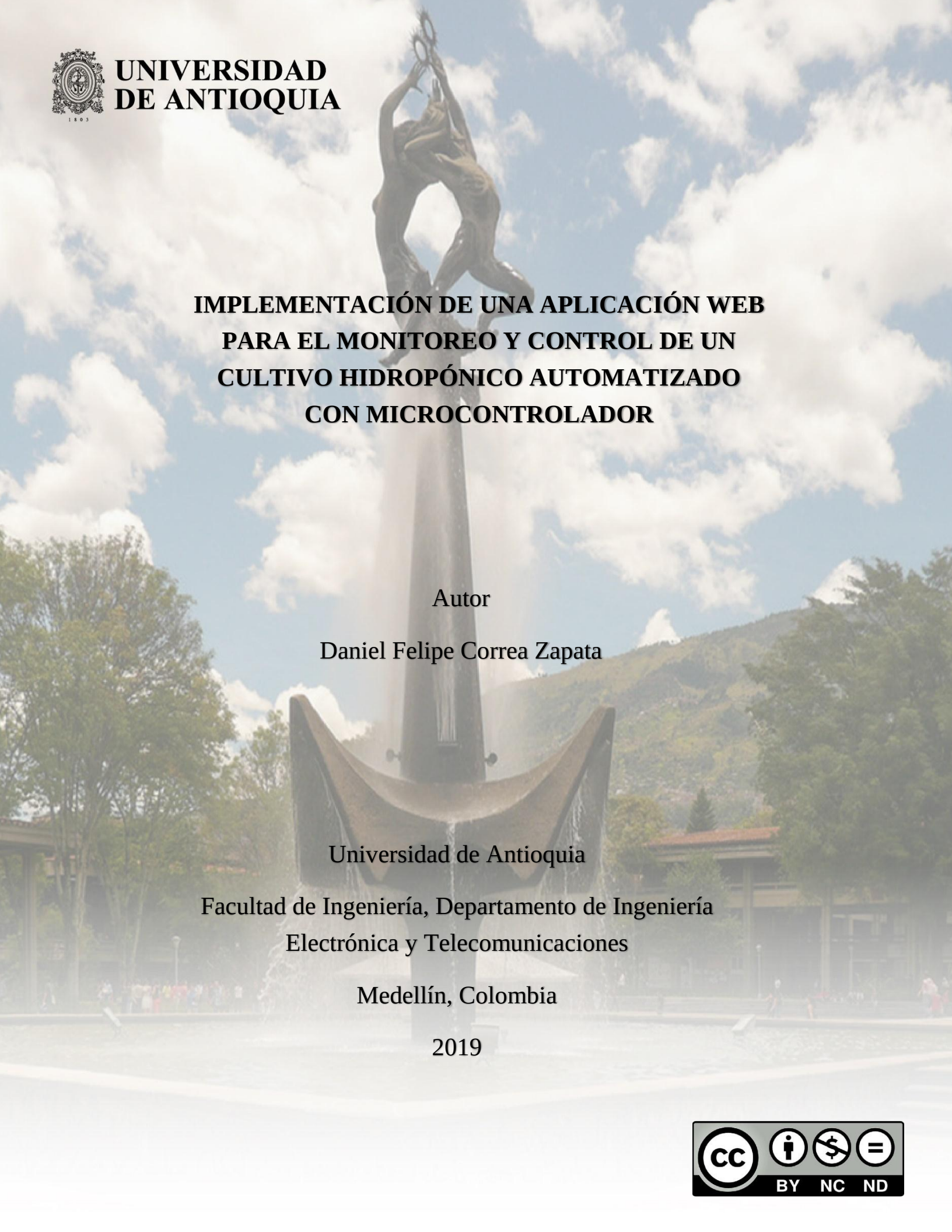

UNIVERSIDAD DE ANTIOQUIA

IMPLEMENTACIÓN DE UNA APLICACIÓN WEB PARA EL MONITOREO Y CONTROL DE UN CULTIVO HIDROPÓNICO AUTOMATIZADO CON MICROCONTROLADOR

Autor

Daniel Felipe Correa Zapata

Universidad de Antioquia

Facultad de Ingeniería, Departamento de Ingeniería Electrónica y Telecomunicaciones

Medellín, Colombia

2019

Resumen

Este proyecto consistió en el desarrollo de una aplicación web para monitorear un cultivo hidropónico a través de una tarjeta de control Raspberry Pi 3. En el diseño e implementación se utilizaron herramientas de uso libre que facilitaron el completo desarrollo de la aplicación. Para el desarrollo del proyecto se inició por una revisión y estudio de los manuales de las herramientas disponibles en la plataforma Firebase de Google con el objetivo de conocer su funcionamiento y configuración, también se estudiaron los lenguajes de programación JavaScript, Python y HTML; así como el lenguaje de diseño CSS, ya que estos constituyen las herramientas de software necesarias para la realización del proyecto. Una vez se tuvo manejo de las herramientas necesarias se procedió a realizar una aplicación web amigable e intuitiva para el usuario. Posteriormente se estudió los conceptos básicos de los cultivos hidropónicos así como su fabricación, requerimientos y automatización. Luego de escoger el mejor diseño de cultivo se construyó con materiales de bajo costo y buena calidad. El siguiente paso consistió en el proceso de automatización del sistema y gracias a las bibliotecas de funciones de Firebase creadas para Linux fue sencilla la comunicación entre la tarjeta de control y la aplicación. Posteriormente se realizó la implementación final de la aplicación junto a la corrección de errores. Al finalizar con el proceso de depuración y corrección se evaluaron los resultados obtenidos y se dio por concluido el proyecto. Esta aplicación demandó desarrollo de software, el estudio, la construcción y la automatización de cultivos hidropónicos, que en conjunto permitieron obtener los resultados esperados y alcanzar así todos los objetivos planteados.

Introducción

Actualmente la tecnología juega un papel fundamental en la sociedad y en la vida diaria de las personas, gracias a esto toma fuerza el concepto del internet de las cosas y aumenta la importancia de crear interfaces amigables e intuitivas que mediante internet permitan integrar, monitorear y controlar objetos y sistemas de naturaleza muy variada tales como electrodomésticos, herramientas eléctricas, sistemas de vigilancia, medios de transporte o cualquier tipo de proceso de importancia para las personas o las empresas. Este proyecto pretende desarrollar e implementar una interfaz de usuario específica para el monitoreo de los sistemas de cultivo hidropónico que toman importancia a medida que incrementan problemas como la contaminación, degradación y erosión de los suelos cultivables y la sobrepoblación. Estos sistemas hidropónicos suelen no contar con sistemas digitales de monitoreo ni control o suelen ser implementados mediante sistemas de alto costo y de licencia privada como los PLCs. Teniendo en cuenta que en Colombia existe dificultad en el acceso a las tecnologías de uso privado, se utilizarán para la ejecución del proyecto hardware y software de uso libre y fácil acceso para las personas, asociaciones o pequeñas y medianas empresas.

El problema abordado es la construcción y automatización de un sistema hidropónico que consiste en una estructura de tubos, un tanque, una bomba, un conjunto de sensores y una tarjeta de control; además el desarrollo de una aplicación web basada en HTML y JavaScript que permite el monitoreo y control de las variables del sistema. Para la construcción del cultivo hidropónico se utilizaron materiales de bajo costo como el plástico y el PVC, bombas de baja potencia, cámara, luces led, sensores de flujo, de temperatura y de humedad; todo compatible y controlado con una plataforma de desarrollo tipo Raspberry Pi conectada a la aplicación web mediante WiFi. Dicha aplicación tiene una interfaz gráfica fácil de entender y de usar por el usuario, además fue desarrollada con lenguajes de uso libre y la plataforma Firebase de Google.

Objetivos

Objetivo General:

Implementar una aplicación web para monitorear y controlar un sistema de circulación de agua (cultivo hidropónico) automatizado mediante microcontrolador y comunicado por medio de WiFi, usando las herramientas de uso libre de la plataforma Firebase de Google.

Objetivos Específicos:

Diseñar e implementar una interfaz gráfica para darle seguimiento a un cultivo hidropónico en tiempo real y desde cualquier lugar con conexión a internet, además disponer de una base de datos en la nube para almacenar valores relevantes del sistema.

Fabricar una estructura con tubos de PVC que permita el flujo de agua en su interior y soporte idóneamente un arreglo de plantas, que junto a un contenedor plástico y una bomba conformen un sistema de circulación de agua (cultivo hidropónico).

Automatizar el sistema de circulación de agua (control de la bomba y luces led) con una plataforma tipo Raspberry Pi que además mida la temperatura, el nivel y el flujo del sistema. Así mismo lograr la comunicación entre el hardware y la aplicación web por medio de WiFi.

Marco Teórico

Internet de las cosas:

Toda maquinaria, sistema o proceso industrial automatizado necesita una interfaz amigable con el operario que facilite el monitoreo y el almacenamiento de datos,

además que permita manipular algunas variables del sistema remotamente. Este concepto llega hasta "(…) escenarios en los que la conectividad de red y la capacidad de cómputo se extienden a objetos, sensores y artículos de uso diario que habitualmente no se consideran computadoras, permitiendo que estos dispositivos generen, intercambien y consuman datos (…)" (Rose, 2015, p. 6). Este concepto toma el nombre de internet de las cosas, pretende un mundo de objetos interconectados a disposición de las personas y avanza rápidamente junto con la tecnología y las telecomunicaciones.

Plataforma de desarrollo y software necesario:

La plataforma Firebase ofrece herramientas para el desarrollo de una aplicación web de gran calidad, estabilidad y seguridad. Para integrar Firebase a la aplicación debe estar basada en los lenguajes JavaScript y HTML. Para la comunicación de la Raspberry con la aplicación (base de datos y el almacenamiento en la nube) es necesario el uso de la librería Pyrebase para Python.

Herramientas de la plataforma Firebase:

Realtime Database:

Es una base de datos alojada en la nube. Los datos se almacenan en formato JSON y se sincronizan en tiempo real con cada cliente conectado. De manera gratuita permite 100 usuarios conectados simultáneamente 1 GB de datos almacenados y 10 GB de datos descargados. (Firebase, 2018)

Cloud Storage:

Es un servicio de almacenamiento de objetos potente, simple y rentable, con operaciones de carga y descarga de archivos para las apps de Firebase, sin importar la calidad de la red. Gratuitamente permite almacenar hasta 5GB y tiene un límite de descargas de 1GB al día, además permite 20 000 operaciones de carga y 50 000 de descarga diariamente. (Firebase, 2018)

Firebase Hosting:

Servicio capaz de implementar apps web y contenido estático en forma rápida y sencilla. Para los servicios gratuitos solo permite almacenar aplicaciones de 1 GB de tamaño como máximo y solo permite transferir 10 GB de datos al mes. (Firebase, 2018)

Firebase Authentication:

Proporciona servicios y bibliotecas ya elaboradas para autenticar a los usuarios de la aplicación. Admite la autenticación mediante contraseñas, números de teléfono, proveedores de identidad como Google, Facebook y Twitter. (Firebase, 2018)

Limitaciones de la plataforma Firebase:

Esta plataforma permite su uso gratuito para diseñadores y por ende las capacidades de sus herramientas son limitadas excepto en el caso de la Firebase Authentication, debe tenerse en cuenta que en caso de que la aplicación necesite más capacidad de procesamiento de datos se debe acceder a planes mensuales que cobran por capacidad de uso o por recursos utilizados al mes, los precios no son muy elevados y son viables a medida que el producto tenga buena demanda de clientes.

Biblioteca Pyrebase:

Biblioteca simple para Python que permite la sincronización con aplicaciones de la plataforma Firebase.

Hidroponía:

Es un conjunto de técnicas que permiten el cultivo sin suelo, donde se suministra de nutrientes a las plantas a través de disoluciones minerales en el agua, "La hidroponía permite en estructuras simples o complejas producir plantas principalmente de tipo herbáceo aprovechando sitios o áreas como azoteas, suelos infértiles, terrenos escabrosos, invernaderos climatizados o no, etc."(Beltrano, 2015, p. 10). En la actualidad se utiliza particularmente la técnica de película nutritiva o "NFT" (Nutrient Film Technique), "(…) consiste en la circulación constante de una lámina fina de solución nutritiva que pasa a través de las raíces del cultivo, no existiendo pérdida o salida al exterior de la solución nutritiva (…)" (Carrasco, 1996, p. 13). En estos cultivos es vital garantizar el movimiento del agua y es conveniente monitorear variables ambientales y del sistema para asegurar su integridad. Estos sistemas toman gran importancia ante problemas de sobrepoblación, de escases o perdida de suelos agrícolas y de desabastecimiento de vegetales en las ciudades, además "El modernismo permitió la introducción de los avances de la informática para el control y ejecución de actividades, que han hecho de la automatización del cultivo hidropónico una realidad."(Beltrano, 2015, p. 9).

Metodología

1 Estudio del desarrollo de aplicaciones web basadas en JavaScript y HTML.

Se hizo un estudio a fondo de las funciones y métodos de los lenguajes JavaScript y HTML, gracias a esto se logró diseñar y programar una interfaz gráfica estética y práctica.

Muestra de código HTML:

- ***<!DOCTYPE html>, <html>*** : encabezados propio del lenguaje.
- ***<link>*** : utilizado para enlazar las hojas de estilos CSS.
- ***<body>*** : define el cuerpo principal de la página.
- *** ,<ul>,<li>,<button>,<p>*** : Elementos visibles de la interfaz gráfica.
- ***id, class*** : atributos de los elementos para su referencia, manejo y estilo.

Se presenta a continuación parte del código en la figura 1:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <link href="css/js1.css" rel="stylesheet" >
    <link href="css/web.css" rel="stylesheet" />
  </head>
  <body class="tus">
    <div id="sidebar">
      <ul>
        <li class="titulo">Estado del Sistema</li>
        <li class="text"><button class="accordion">Conexión</button>
                        <div class="panel">
                          <p id="conex">No Establecida</p>
                        </div></li>
```

Figura 1: Parte del código HTML de la página principal de la aplicación. *Fuente propia.*

Muestra de las funciones utilizadas de JavaScript:

- ***if( )*** : algoritmo de decisión.
- ***function*** : declara una función.
- ***document.getElementById('ID'):***crea un puntero hacia el elemento de HTML identificado por el id.
- ***.innerHTML*** : establece un texto en el elemento apuntado.

Se presenta a continuación instrucciones básicas del código en la figura 2:

```
function cargar(){

    ///Estados
    if (t.estado=="on") {document.getElementById('conex').innerHTML="Sistema en Conexión";}
    if (t.estado=="off") {document.getElementById('conex').innerHTML="No Establecida";}
```

Figura 2: Parte del código JavaScript de la página principal de la aplicación. *Fuente propia.*

Muestra del lenguaje CSS:

- ***.sidebar { }*** : define la clase de nombre 'sidebar'.
- ***position*** : atributo que define la posición del elemento respecto a los otros.

- ***width, height*** : atributos que definen el ancho y el alto de un elemento respectivamente.
- ***background*** : define el color de fondo del elemento.
- ***transition***: atributo para la animación de movimiento de un elemento.

Se presenta a continuación parte del código de estilo en la figura 3:

```
.sidebar {
  position: fixed;
  width: 300px;
  height: 100%;
  background: #19718A;
  transition: all 500ms linear;
}
```

Figura 3: Definición de una clase en una hoja de diseño CSS. *Fuente propia.*

2 Identificación de las herramientas de desarrollo web.

Las bases de datos, el servidor, el almacenamiento en la nube y la autentificación de usuarios para el funcionamiento de la aplicación son proporcionados por la plataforma Firebase, en su página se encuentran los tutoriales de inicio para integrar dicha plataforma con nuestra aplicación.

3 Implementación de la aplicación web.

Con el manejo de las herramientas de Firebase se procede a implementar la aplicación. Para comenzar se crea un proyecto en la consola de Firebase llamado hydroponic-420 y luego se agrega la configuración en los códigos HTML de la aplicación:

Para añadir Firebase a la aplicación se utiliza el código de la figura 4:

```
<script src="https://www.gstatic.com/firebasejs/5.5.2/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/5.5.2/firebase-database.js"></script
>
<script src="https://www.gstatic.com/firebasejs/5.5.2/firebase-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/5.5.2/firebase.js"></script>
<script>
  // Initialize Firebase
  var config = {
    apiKey: "AIzaSyCx9PkqdiJhT7Va9vTtRArToYnhaxtXpBo",
    authDomain: "hydroponic-420.firebaseapp.com",
    databaseURL: "https://hydroponic-420.firebaseio.com",
    projectId: "hydroponic-420",
    storageBucket: "hydroponic-420.appspot.com",
    messagingSenderId: "1011877113209"
  };
  firebase.initializeApp(config);
  var database = firebase.database();
  var storage = firebase.storage();
</script>
```

Figura 4: *Inicialización de Firebase en la aplicación. Fuente propia.*

Realtime Database: Funciones para el uso de la base de datos.

- ***Firebase.database( )*** : crea una instancia de la base de datos.
- ***.ref( )*** : crea una referencia donde serán leídos o escritos los datos.
- ***.on("child_added",function( ))*** : crea un objeto con los datos leídos.
- ***.on("child_changed",function( ))*** : crea un objeto con los datos de una referencia que ha sido modificada en sus niveles inferiores.
- ***.set( ) :*** escribe los datos ingresados a la referencia seleccionada, si la referencia ya existe los datos se sobrescriben.

Los códigos necesarios para la lectura de datos se muestran en la figura 5:

```
firebase.database().ref('cultivos/'+cod).on("child_added", function(snapshot) {
    t = snapshot.val() ;
    cargar();
});

firebase.database().ref('cultivos/'+cod).on("child_changed", function(snapshot) {
    t = snapshot.val() ;
    cargar();
});
```

Figura 5: *Rutinas para la lectura de datos de la Realtime Database. Fuente propia.*

Los códigos necesarios para la escritura de datos se muestran en la figura 6:

```
firebase.database().ref('fotos/'+codigo+'/foto/text').set({
        num: 0,
});
```

Figura 6: *Rutinas para la escritura de datos en la Realtime Database. Fuente propia*

La base de datos en tiempo real tendrá una estructura similar a la mostrada en la figura 7:

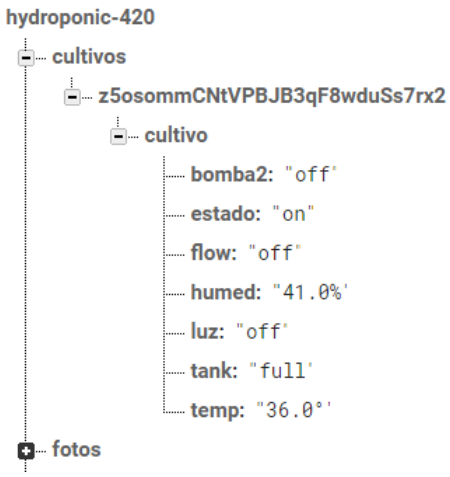

Figura 7: Estructura de la base de datos del proyecto. *Fuente propia.*

Cloud Storage: Funciones para el uso del almacenamiento en la nube.

- ***firebase.storage( )*** : crea una instancia de la carpeta de almacenamiento.
- ***.ref( )*** : *crea una referencia con la dirección de los archivos a descargar de la nube.*
- ***.getDownloadURL( ) :***proporciona la dirección de descarga de los archivos.
- ***XMLHttpRequest().open('GET', url) :*** realiza la petición web con el método elegido y la dirección.
- ***Blob( ):*** crea un objeto tipo blob con los datos obtenidos para que sean interpretados como una imagen.
- ***.createObjectURL( ):*** crea la dirección que será asignada al elemento tipo imagen de HTML.

Se presenta a continuación la figura 8 que contiene la rutina necesaria para descargar archivos de la nube y mostrarlos en la aplicación:

```
firebase.storage().ref('imagenes/'+cod+'/'+i+'.jpg').getDownloadURL().then(function(url) {
    var xhr = new XMLHttpRequest();
    xhr.open('GET', url);
    xhr.responseType = 'arraybuffer';
    xhr.onload = function(event) {
    var arrayBufferView = new Uint8Array( xhr.response );
    var blob = new Blob( [ arrayBufferView ], { type: "image/jpeg" } );
    var urlCreator = window.URL || window.webkitURL;
    var imageUrl = urlCreator.createObjectURL( blob );
    var imas= document.getElementById("imagtex");
    imas.src = imageUrl;
    };
    xhr.send();
  }).catch(function(error) {
    // Handle any errors
  });
```

Figura 8*:* Rutina para la descarga de archivos y posterior cargar en la interfaz de usuario. *Fuente propia.*

Firebase Hosting: pasos para implementar la página web.

- Se debe instalar Node.js y npm a fin de usar Firebase CLI (interfaz de línea de comandos) e instalar sus herramientas, se debe estar en la raíz del directorio del proyecto.
- **login**: conecta el equipo local a tu cuenta de Firebase y obtiene acceso a los proyectos.
- **init**: crea un archivo de configuración firebase.json en la raíz del directorio del proyecto.
- **deploy**: implementa una actualización en el sitio de Hosting predeterminado de tu proyecto.

Con la ayuda de la consola de Node.js se ingresa a la carpeta raíz de la aplicación y se ejecutan las instrucciones que se muestran a continuación en la figura 9 para implementar la aplicación en la web:

```
$ npm install -g firebase-tools
$ firebase login
$ firebase init
$ firebase deploy
```

Figura 9: Rutina utilizada para implementar la aplicación. *Tomada de los manuales de Firebase Hosting*

Firebase Authentication: Funciones para el uso de la autentificación.

- ***firebase.auth( )*** : crea una instancia a la base de datos de autentificación.
- ***.createUserWithEmailAndPassword(email, password)*** : método para registrar usuarios con email y contraseña.
- ***.signInWithEmailAndPassword(email, password)*** : método para autenticar los datos de usuario.
- ***.onAuthStateChanged( )*** : método para monitorear el estado de conexión del usuario.

4 Identificación de los modelos de construcción de los cultivos "NFT".

Entre las estructuras analizadas se escogió el diseño de una estructura con capacidad para veinte plantas, fabricada en tubos de PVC con cubierta de plástico para invernadero, de poca complejidad de fabricación, el costo inicial del cultivo que representa esta estructura puede pensarse como una inversión viable gracias al gran potencial de producción del cultivo.

5 Análisis de los parámetros básicos de funcionamiento en cultivos "NFT".

Se hizo un acercamiento a los parámetros y requerimientos básicos propuestos para este tipo de sistema, tales como el flujo recomendado de la bomba, altura de la película de agua sobre las raíces, temperatura del agua, numero de riegos al día, nutrientes necesarios, pH requerido, entre otros; se hizo lo posible por implementar estos valores en el sistema.

6 Fabricación del sistema y automatización con Raspberry Pi.

Se construyó la estructura de cultivo escogida y se acoplaron los sistemas de circulación de agua y de sensado de variables, además se automatizó el funcionamiento de dichos sistemas mediante la plataforma Raspberry Pi.

La estructura del invernadero (izquierda) y la estructura de tubos que soporta las plantas (derecha) mostrados en la figura 10:

Figura 10: Estructura fabricada con tubos de PVC, en la imagen de la derecha se observan los tubos donde estarán las plantas y fluirá el agua. *Fuente propia.*

Para la automatización del sistema se utilizó una tarjeta de control Raspberry Pi 3 B+ junto a una cámara de 5 MP (figura 11), dos relé de estado sólido para prender y apagar las luces y la bomba de riego (figura 12), para conocer las variables del ambiente se usó un sensor de temperatura y humedad (figura13), una fotorresistencia (figura 14) y un sensor de nivel compuesto por dos cables cercanos que cierran el circuito en contacto con el agua.

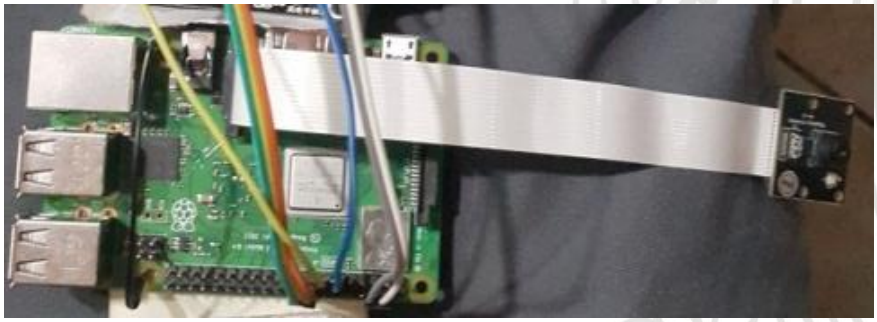

Figura 11: Tarjeta Raspberry Pi con cámara. *Fuente propia*.

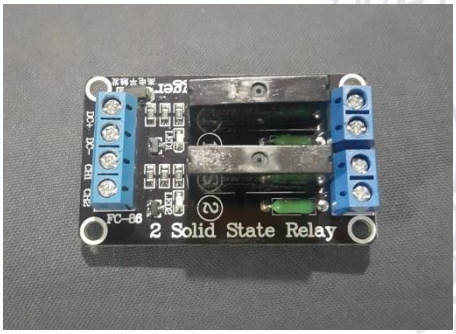

Figura 12: Relé de estado sólido de referencia 2CH. *Fuente propia*.

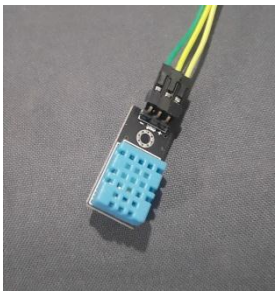

Figura 13: Sensor de temperatura y humedad referencia DHT11. *Fuente propia*.

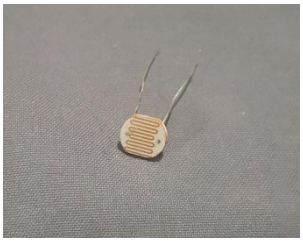

Figura 14: Fotorresistencia que indica luminosidad del sistema. *Fuente propia*.

RPi.GPIO: Funciones de la biblioteca.

- **.setup(número, modo)** : configura el pin del número indicado con el modo de entrada o salida
- **.input(número)** : lee el valor de entrada del pin con el número indicado.
- **.output(número, estado)** : establece el estado de salida (True o False) del pin con número indicado.

Se presenta a continuación la rutina para el encendido de luces en la figura 15:

```
GPIO.setup(27, GPIO.OUT) #luces

print("prender luces")
GPIO.output(27, True)
```

Figura 15: Rutina para el encendido de las luces. *Fuente propia.*

Para automatizar la ejecución de las rutinas se hizo uso del fichero crontab mostrado en la figura 16:

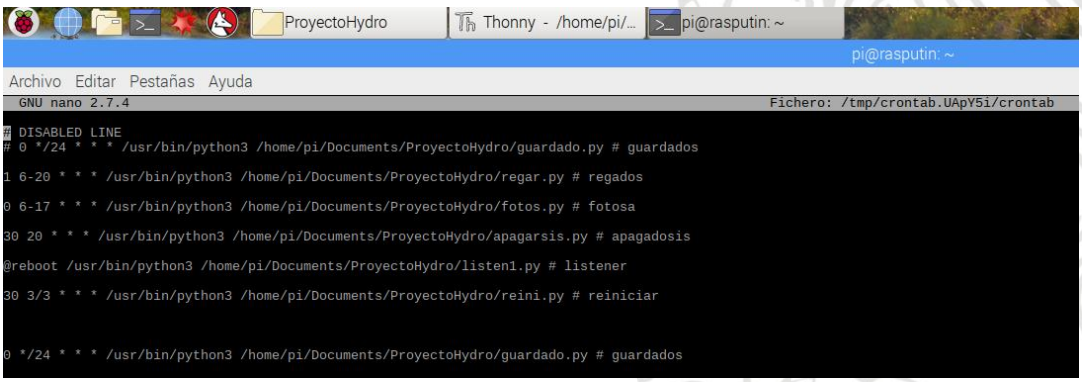

Figura 16: Lista de comandos o tareas en el fichero crontab que serán ejecutadas automáticamente. *Fuente propia.*

Cada vez que se inicie el sistema se llama al código que estará en constante comunicación con la aplicación web (# listener), además el resto de códigos serán ejecutados según la frecuencia definida por el usuario.

7 Comunicación entre la aplicación y el sistema vía WiFi.

Se logró el intercambio de datos entre la aplicación y la Raspberry vía WiFi, mediante la biblioteca para Python Pyrebase, así se pudo obtener un acceso rápido y sencillo a bases de datos de la aplicación.

Pyrebase: Funciones de la biblioteca.

- ***initialize_app(config)*** : accede a los datos de la aplicación con la configuración correspondiente.
- ***.auth()*** : crea una instancia a la base de datos de autentificación.
- ***.sign_in_with_email_and_password(email, contraseña)*** : accede a los datos del usuario.
- ***.database()*** : crea una instancia de la base de datos.
- ***.child(dir).get().val() :*** lee un valor en la base de datos con la dirección indicada.
- ***.child(dir).set(nombre) :*** escribe un valor en la base de datos con la dirección indicada.
- ***.storage() :*** crea una instancia a la carpeta de almacenamiento.
- ***.child(dir).put(nombre) :*** ingresa un archivo a la dirección indicada de almacenamiento.

Para instalar la biblioteca es necesaria la siguiente instrucción mostrada en la figura 17:

```
pip install pyrebase
```

Figura 17: Línea para instalar la librería Pyrebase para Python. *Fuente propia.*

Para acceder a la aplicación desde la Python es necesario enlazarla de la manera mostrada en la figura 18:

```
import pyrebase as py
config = {
        "apiKey": "AIzaSyCx9PkqdiJhT7Va9vTtRArToYnhaxtXpBo",
        "authDomain": "hydroponic-420.firebaseapp.com",
        "databaseURL": "https://hydroponic-420.firebaseio.com",
        "projectId": "hydroponic-420",
        "storageBucket": "hydroponic-420.appspot.com",
        "messagingSenderId": "1011877113209"
}
firebase = py.initialize_app(config)
```

Figura 18: Configuración de acceso a los datos de la aplicación. *Fuente propia.*

Para autenticar al usuario y que pueda acceder a los datos del cultivo es necesario utilizar la siguiente rutina de la figura 19:

```
auth = firebase.auth()
user = auth.sign_in_with_email_and_password(email, contraseña)
cod = user['idToken']
```

Figura 19: Rutina para acceder al id del usuario por medio de su autentificación. *Fuente propia.*

El código para tomarle una foto al cultivo y posteriormente subirla a la nube se presenta en la figura 20:

```
dir="fotos/"+cod+"/foto/fotos"
numfoto = firebase.database().child(dir).get().val()
numfoto=numfoto+1
dir1="/home/pi/Documents/ProyectoHydro/fotos/"+str(numfoto)+".jpg"
subprocess.call(['raspistill','-rot','270' , '-o' , dir1])
times=time.strftime("%H:%M:%S")
dates=time.strftime("%d/%m/%y")
totaldate="Hora: "+times+" Fecha:"+dates
dir2="imagenes/"+cod+"/"+str(numfoto)+".jpg"
firebase.storage().child(dir2).put(dir1);
dir3="fotos/"+cod+"/foto/text/foto"+str(numfoto)
dir4="fotos/"+cod+"/foto/text/num"
num = firebase.database().child(dir4).get().val()
num=num+1
firebase.database().child(dir).set(numfoto)
firebase.database().child(dir3).set(totaldate)
firebase.database().child(dir4).set(num)
```

Figura 20: Rutina para tomar una foto, subirla a la nube y actualizar la base de datos. *Fuente propia.*

Subprocess: Función de la biblioteca.

- **subprocess.call(['raspistill','-rot','270' , '-o' , dir/nombre])** : corre el comando necesario para tomar una foto con el módulo de la cámara de Raspberry, la rota 270 grados y la guarda con el nombre y dirección indicada.

Time: Función de la biblioteca.

- **time.strftime()** : devuelve un texto con los datos requeridos tales como hora, minuto, segundo, día, mes, año.

8 Pruebas y corrección de errores.

Se puso a prueba todas las funciones, procesos y estados posibles de la aplicación web y del sistema, fue necesario modificar varias veces el código de funcionamiento de la aplicación y de la tarjeta de control, así se garantizó su correcta automatización y comunicación, con el fin de evitar errores o mal funcionamiento que puedan afectar la salud de las plantas.

Resultados y análisis

Se desarrolló e implementó una aplicación web de nombre HYDROPONIC y dirección https://hydroponic-420.web.app/.

Se construyó una estructura tipo invernadero de 2.4 metros de largo, 1.9 de ancho y 2.4 de alto, con capacidad para 20 plantas. Esta estructura contiene un tanque de 6 galones con una bomba sumergible, contiene los tubos de PVC donde circula el agua y crecen las plantas (Figura 10) y también se encuentra en su interior un contenedor donde está la Raspberry Pi, la cámara y los relés (Figura 11).

Este sistema de cultivo se automatizó correctamente y se garantizó la comunicación con la aplicación web. Finalmente se hicieron las pruebas de funcionamiento tanto de la aplicación como del sistema y se pudo corregir una variedad de errores que terminarían por afectar la salud y el crecimiento de las plantas.

El usuario interactúa con el sistema a través de la página web, la cual presenta el estado del cultivo (variables sensadas y fotografías), además permite programar el funcionamiento del cultivo definiendo la frecuencia de los riegos, la frecuencia de la toma de fotos, la frecuencia del guardado de datos, la hora de prendido y apagado de las luce y la hora de apagado del sistema. Las órdenes ingresadas son guardadas en la base de datos y posteriormente ejecutadas por la tarjeta de control.

Se presenta un diagrama de flujo del proyecto donde se evidencian por bloques los componentes más relevantes del sistema y la conexión o interacción con los demás, en los bloques la figura 21 se presenta las partes y funciones antes descritas:

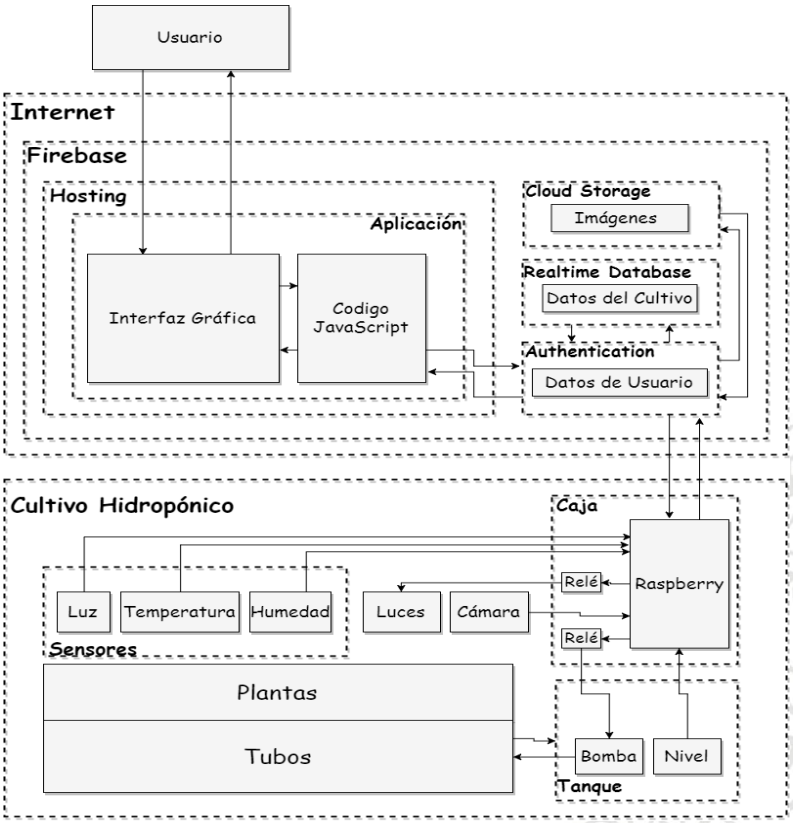

Figura 21: Diagrama de flujo del proyecto. *Fuente propia.*

Actualmente en la industria se encuentran sistemas similares, pero estos utilizan generalmente un equipo robusto y de elevado costo como los PLC y la interfaz gráfica es desarrollada utilizando software privativo que requiere de la adquisición de licencias para su respectivo uso.

Se obtuvo como resultado final un cultivo hidropónico automatizado dotado de una interfaz de usuario realizada con herramientas de libre acceso y que permite el monitoreo del sistema desde cualquier sitio con conexión a internet. La interfaz de usuario de la aplicación web se muestra en la figura 22:

Figura 22: Interfaz de usuario. *Fuente propia.*

Descripción y funcionamiento de la interfaz ilustrada en la Figura 22:

1 Botón de *estado:* Se utiliza para ingresar a la interfaz mostrada donde se puede acceder a las variables y las fotos del sistema, además se puede programar el funcionamiento del cultivo.

2 Botón de *historial:* Se utiliza para acceder al historial de datos del sistema, donde se almacena la información de las variables guardadas periódicamente.

3 Botón de *ayuda*: Se utiliza para acceder al manual de usuario donde se explica detalladamente cada función de la aplicación.

4 Estado del sistema, *Conexión*: Indica el estado de conexión entre la página web y la Raspberry.

5 Estado del sistema, *Bomba de circulación*: Permite conocer el estado de la bomba de circulación así como activarla o desactivarla.

6 Estado del sistema, *Tanque principal*: Permite conocer el nivel de agua en el tanque, si está lleno, medio o vacío.

7 Estado del sistema, *Flujo de agua*: Útil para tener la certeza de que el agua está fluyendo por el sistema.

8 Estado del sistema, *Humedad*: Indica la medida en tiempo real de la humedad relativa del ambiente.

9 Estado del sistema, *Temperatura:* Indica la medida en tiempo real de la temperatura del sistema.

10 Datos fotografías: Permite conocer la hora y la fecha en las cuales se tomó la fotografía visualizada en la pantalla, además permite elegir la foto que se desea ver.

11 Programar, *Fotografías*: Permite establecer la hora y la frecuencia en que se tomarán las fotografías al sistema.

12 Programar, *Tiempos de circulación*: Permite establecer la hora, la frecuencia y el tiempo de riego del cultivo.

13 Programar, *Tiempos de iluminación*: Permite establecer la hora de prendido y apagado del sistema de luces.

14 Programar, *Guardado de datos*: Permite establecer la hora y la frecuencia en que se guardarán los datos del sistema.

15 Programar, *Apagar sistema*: Permite apagar y reiniciar el sistema, además permite establecer la hora de apagado automático.

Con la aplicación también es posible llevar registro del crecimiento de las plantas como se muestra a continuación en la figura 23:

Figura 23: Conjunto de cinco imágenes que evidencian el crecimiento de una planta de tomate. *Fuente propia.*

Conclusiones

En la actualidad el medio ingenieril requiere de profesionales capaces de explorar alternativas y de innovar en el desarrollo de proyectos, de esta manera el ingeniero debe aprovecharse del cambiante medio tecnológico el cual ofrece nuevas opciones y soluciones cada vez más versátiles y económicas en los distintos campos de aplicación de la ingeniería. Con el presente proyecto se logró el aprovechamiento de las tecnologías de uso libre para lograr el desarrollo de una aplicación web en el campo del internet de las cosas.

Desde un principio el propósito del proyecto surgió de un profundo análisis de sistemas útiles para la vida contemporánea que podrían ser mejorados con la aplicación de la ingeniería. La temática propuesta, el trabajo invertido y la

disposición por integrar nuevos sistemas al campo del internet de las cosas por medio de la ingeniería, ayudaron a que este proyecto se desarrollara de la mejor forma, aprovechando los desafíos que finalmente llevaron a la culminación del trabajo cumpliendo los objetivos planteados en un principio.

El desarrollo del proyecto permitió aportar alternativas de diseño, monitoreo y control al campo de los sistemas de cultivo. La solución implementada representa una muestra de la necesaria relación entre usuario y máquina, que además podría implementarse fácilmente para otros sistemas de la vida cotidiana en el campo de la industria, el hogar, el transporte, la medicina, entre otros; ya que a medida que avanza la tecnología se hace más fácil e incluso necesario la integración del mundo físico al mundo de las telecomunicaciones, todo con el fin de hacer un mundo más interconectado que ayude al mejoramiento de la calidad de vida de las personas, que represente una alternativa de generación de empleo para los ingenieros y que motive a la formación de empresa.

Con la finalización del proyecto se concluye que las empresas pueden tomar en cuenta el desarrollo de estos sistemas de bajo costo y uso libre que garantizan igual rendimiento y calidad en las aplicaciones requeridas. De esta manera se estaría impulsando el desarrollo ingenieril del país de la mano de profesionales cada vez más capaces de afrontar un entorno cambiante para sacar el mejor provecho.

El software de código abierto representa un apoyo fundamental en la formación del ingeniero y en el desarrollo de aplicaciones profesionales reales. En el presente trabajo se destaca el uso de herramientas de libre acceso, las cuales disminuyen los costos asociados a licencias de uso y de esta manera hacen más rentables los proyectos. Es por esta razón que toma importancia motivar el desarrollo de proyectos de ingeniería basados en herramientas de uso libre que brindan al desarrollador distintas alternativas y posibilidades de diseño.

La experiencia y los conocimientos adquiridos durante el proceso de fabricación, montaje y automatización del sistema hacen de este proyecto una importante experiencia formativa en el campo profesional de la ingeniería. Estos conceptos y conocimientos contribuyen a la formación en distintas áreas del ejercicio profesional como el desarrollo web, la integración de procesos u objetos al internet de las cosas, los sistemas embebidos, el desarrollo de proyectos, entre otros; siendo un complemento importante de las bases aprendidas en el estudio de la ingeniería electrónica.

Referencias Bibliográficas

Beltrano, J., Giménez, D. O. (2015). *Cultivo en hidroponía*. Recuperado de http://sedici.unlp.edu.ar/handle/10915/46752

Ben Croston (2018). *RPi.GPIO 0.6.5*. Delaware, EU.: Python Software Foundation. Recuperado de https://pypi.org/project/RPi.GPIO/

Carrasco, G., Izquierdo, J. (1996). *MANUAL TECNICO LA EMPRESA HIDROPONICA DE MEDIANA ESCALA: LA TECNICA DE LA SOLUCION NUTRITIVA RECIRCULANTE ("NFT")*. Talca, Chile: Editorial Universidad de Talca.

Firebase (2018). *Cómo agregar Firebase a tu proyecto de JavaScript*. Recuperado de https://firebase.google.com/docs/web/setup

Firebase (2018). *Firebase Realtime Database, introducción*. Recuperado de https://firebase.google.com/docs/database/

Firebase (2018*). Cloud Storage, introducción*. Recuperado de https://firebase.google.com/docs/storage/

Firebase (2018). *Firebase Hosting, introducción*. Recuperado de https://firebase.google.com/docs/hosting/

Firebase (2018). *Firebase Authentication, introducción*. Recuperado de https://firebase.google.com/docs/auth/

James Childs-Maidment (2017). *Pyrebase 3.0.27*. Delaware, EU.: Python Software Foundation. Recuperado de https://pypi.org/project/Pyrebase/

Rose, K., Eldridge, S., Chapin, L. (2015). *La Internet de las Cosas—Una breve reseña Problemas y desafíos de un mundo más conectado.* Geneva, Suiza: Internet Society.