UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE GUAYAQUIL

CARRERA DE INGENIERÍA ELECTRÓNICA

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO ELECTRÓNICO

PROYECTO TÉCNICO:

"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA ELECTRÓNICO PARA CRONOMETRAR COMPETENCIAS DE NATACIÓN Y SUPERVISIÓN DEL ESTADO DEL AGUA, MEDIANTE UN VISUALIZADOR ELECTRÓNICO PARA LA PISCINA DE LA UNIVERSIDAD POLITÉCNICA SALESIANA SEDE GUAYAQUIL"

AUTORES:

OSCAR ANDRÉS TINAJERO GUERRA

CÉSAR ANDRÉS LARA CEREZO

TUTOR:

ING. PABLO ECHEVERRÍA

GUAYAQUIL – ECUADOR

2020

DECLARACIÓN DE RESPONSABILIDAD DE AUTORÍA

Nosotros, César Andrés Lara Cerezo con número de cédula de identidad de ciudadanía 0950012500 y Oscar Andrés Tinajero Guerra con número de cedula de identidad de ciudadanía 0931788186, estudiantes de la carrera de Ingeniería Electrónica respectivamente, podemos corroborar que los conceptos desarrollados, diseño realizado, resultados analizados, las ideas y las propuestas de solución así como los criterios en la totalidad del presente trabajo técnico de titulación, son de exclusiva responsabilidad de los autores.

A través de la presente declaración otorgamos los derechos de propiedad intelectual correspondiente a este trabajo, a la UNIVERSIDAD POLITÉCNICA SALESIANA DEL ECUADOR, según lo establecido por la ley de la Propiedad Intelectual, por su reglamento y por la Normativa Institucional Vigente.

Andres Lara C.

César Andrés Lara Cerezo

C.I.:0950012500

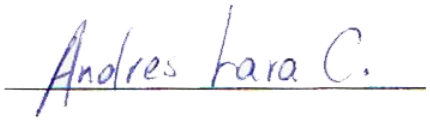
Oscar Tinajero G.

Oscar Andrés Tinajero Guerra

C.I.:0931788186

DECLARACIÓN DE CESIÓN DE DERECHOS DE AUTOR

Nosotros, César Andrés Lara Cerezo con cédula de ciudadanía N°0950012500, y Oscar Andrés Tinajero Guerra con cédula de ciudadanía N°0931788186 manifestamos nuestra voluntad de ceder a la UNIVERSIDAD POLITÉCNICA SALESIANA DEL ECUADOR la titularidad sobre los derechos patrimoniales en facultad de que somos autores del proyecto de técnico titulado: “DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA ELECTRÓNICO PARA CRONOMETRAR COMPETENCIAS DE NATACIÓN Y SUPERVISIÓN DEL ESTADO DEL AGUA, MEDIANTE UN VISUALIZADOR ELECTRÓNICO PARA LA PISCINA DE LA UNIVERSIDAD POLITÉCNICA SALESIANA SEDE GUAYAQUIL”, el mismo cual ha sido diseñado e implementado para obtener el título de Ingeniero Electrónico, quedando a nuestra universidad facultada para hacer uso plenamente los derechos de esta obra cedidos anteriormente con fines académicos o de investigación para beneficio de comunidad universitaria.



César Andrés Lara Cerezo

C.I.:0950012500

Oscar Andrés Tinajero Guerra

C.I.:0931788186

DECLARACIÓN DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

En calidad de TUTOR DEL PROYECTO TÉCNICO “Diseño e implementación de un sistema electrónico para cronometrar competencias de natación y supervisión del estado del agua, mediante un visualizador electrónico para la piscina de la Universidad Politécnica Salesiana sede Guayaquil”, elaborado por César Andrés Lara Cerezo y Oscar Andrés Tinajero Guerra, declaro y certifico la aprobación del presente proyecto técnico basándose en la supervisión y revisión de su contenido.

Dado en la Ciudad de Guayaquil, a los 23 días del mes de marzo del 2020.

Msc. Ing. Pablo Echeverría.

Docente Tutor del Proyecto Técnico

DEDICATORIA

Dedico este proyecto de tesis primordialmente a Dios y a mis padres César Bolívar Lara Espinoza y Martha Cecibel Cerezo Morán por ser mis pilares fundamentales de vida por ser mi fuente de inspiración para seguir adelante, gracias por la compresión que demandaron las largas jornadas de tiempo para terminar este proyecto. A mis hermanos por ser ejemplos de perseverancia, de que las cosas se las obtiene superando los obstáculos por más duros que sean. A mis familiares por sus consejos y enseñanzas que de alguna manera u otra han estado ahí para mí cuando más los necesité, a Dios le sigo agradeciendo por darme sabiduría y persistencia para desarrollar los retos planteados.

César Andrés Lara Cerezo.

DEDICATORIA

Dedico este proyecto de tesis a mis padres Oscar Luis Tinajero León y Mercedes Zoraida Guerra Castro por ser los pilares fundamentales en mi vida, apoyo importante en mis estudios, los cuales mediante valores, consejos y enseñanzas me ayudaron a cumplir mis objetivos como persona y estudiante para la culminación de mi carrera de Ingeniería Electrónica; y a mis hermanos, que son un pilar fundamental para poder ser mejor cada día, y a toda mi familia y amigos.

Oscar Andrés Tinajero Guerra.

AGRADECIMIENTO

Agradezco primordialmente a Dios por habernos dado la oportunidad de estudiar, haberme brindado día a día la fortaleza, sabiduría y perseverancia para poder terminar esta hermosa carrera y nuestra tesis de grado.

A mis padres que estoy seguro de que han sido mi motor y motivación, mis hermanos un apoyo incondicional, y hoy les dedico mi mayor logro, esto fue posible gracias a ellos.

César Andrés Lara Cerezo

AGRADECIMIENTO

Agradezco a Dios primeramente por darme salud y por permitirme salir adelante y a mi familia por haber apoyado en cada ciclo de mi vida, para poder haber llegado hasta el día de hoy .

También estoy muy agradecido a mis compañeros y docentes universitarios con los que hemos tenido el gusto de compartir durante nuestra etapa universitaria.

Oscar Andrés Tinajero Guerra.

RESUMEN

El presente proyecto técnico plantea el diseño e implementación de un sistema electrónico para visualizar y cronometrar competencias de natación entre los estudiantes de la comunidad universitaria, el cual registra el tiempo exacto y preciso de los competidores según su orden de llegada, posición y a su vez asigna estos parámetros a una base de datos MySQL donde se almacenará los resultados que se acontecen.

Lo interesante y novedoso de esto es que se podrá sensar y visualizar la característica del agua de la piscina, como del pH (indica la alcalinidad o la acidez), temperatura y turbidez, mediante sensores (basado en el entorno de Arduino) subacuáticos ubicados en una parte estratégica de la piscina.

Se diseñó, armó e instaló paneles subacuáticos de toque mediante el uso de sensores piezoeléctricos (detectan vibraciones) que serán ubicados en el área de la plataforma de salida de la piscina, los paneles registrarán el tiempo exacto de cada competidor, garantizando así evitar cualquier tipo de error en la medición de tiempo, que previamente era realizado por personas (Jueces) con reloj en mano, método rústico, así dando más credibilidad a futuras competencias internas o entre sedes.

La señal generada por los paneles de toque de llegada y los datos de los sensores que indica el estado del agua respectivamente, serán enviados a través de la tecnología WIFI IEEE 802.11 mediante el uso de la tarjeta wemos D1 mini, que es una placa basada en el chip ESP8266, el cual permite comunicarse de manera práctica e inalámbricamente con el módulo central que en este proyecto es un raspberry pi 3.

El instructor o juez podrá establecer diferentes estilos de natación que se podrán competir, así también ingresar los nombres de competidores y carriles que utilizarán para la competencia, a su vez dar inicio a la misma, todos estos parámetros podrán ser accedidos y asignados desde cualquier dispositivo electrónico conectado a una red interna WLAN (Wireless Local Area Network) a través de una vistosa interfaz web dinámica, la misma que fue desarrollada y almacenada en un servidor http que en este caso será la raspberry Pi3, esta interfaz será completamente adaptable para dispositivos electrónicos móviles como celulares, tabletas y laptops.

ABSTRACT

This technical project proposes the design and implementation of an electronic system to visualize and time swimming skills among students of the university community, which records the exact and precise time of the competitors according to their order of arrival, position and in turn assigns these parameters to a MySQL database where the results that occur will be stored.

The interesting and novel thing about this is that you can censor and visualize the characteristic of water, such as pH (indicates alkalinity or acidity), temperature and turbidity of water, using sensors (based on the Arduino environment) underwater located in a strategic part of the pool.

Underwater touch panels were designed, assembled and installed through the use of piezoelectric sensors (detects vibrations) that will be located in the area of the pool's exit platform, the panels will record the exact time of each competitor, thus ensuring to avoid any type of errors in the measurement of time, which was previously made by people (Judges) with clock in hand rustic method, thus giving more credibility to future internal competitions or between venues.

The signal generated by the touch panels of arrival and the data of the sensors that indicate the state of the water respectively, can be sent through the IEEE 802.11 WIFI technology by using the wemos D1 mini card, which is a board based on The ESP8266 chip, which allows you to communicate practically and wirelessly with the central module in this project is a raspberry pi 3.

The instructor or judge may establish different swimming styles that may be competed, as well as enter the names of competitors and lanes that they will use for the competition, in turn to initiate it, all these parameters may be accessed and assigned from any connected electronic device access of a private WLAN network (Wireless Local Area Network) through a colorful dynamic web interface that will be developed and stored in a http server that in this case it will be the raspberry pi3, this interface will be fully adaptable for mobile electronic devices such as cell phones, tablets and laptops.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

GLOSARIO

IDE: Integrated development environment “Entorno de desarrollo integrado”

GPIO: General Purpose Input Output “Salida de entrada de propósito general”

HTML: Hyper Text Markup Language “Lenguaje de marcado de hipertexto”

JS: JavaScript

LAN: Local Area Network “Red de área Local”

AP: Access point “Punto de acceso”

pH: “Potencial de Hidrógeno”

TDS: “Total de sólidos disueltos”

HDMI: High-Definition Multimedia Interface “Multimedia de interfaz en alta definición”

WIFI: Wireless Fidelity “Fidelidad o campo Inalámbrico”

TCP/IP: Modelo de un protocolo para comunicación en redes

LED: “Diodo emisor de luz”

HTTP: “Hyper Text Transfer Protocol”

URL: Uniform Resource Locator “Localizador Uniforme de Recursos”.

PCB: Printed Circuit Board “Placa de Circuito Impreso”

GND: Siglas representativas para Tierra.

INTRODUCCIÓN

La Universidad Politécnica Salesiana con sede en Guayaquil posee en el campus centenario varios bloques en la cual resalta el área deportiva la misma que cuenta con un espacio de piscina para que los estudiantes cada vez más ocupen su tiempo realizando cualquier tipo de actividad física y recreativa en ella, por lo que hoy en día es necesario poner interés a este deporte acuático que es uno de los más completos que existe a nivel internacional.

Debido a lo anteriormente mencionado, en la universidad cada vez los estudiantes optan realizar entrenamientos en las instalaciones de la piscina, ya sea en sus ratos libres, o después de los estudios ya sea para practicar, entrenarse para competencias de la índole acuática. En vista de que en la universidad específicamente en el área de piscina no cuenta con un tablero electrónico que permita registrar tiempos de competidores, razón por la cual a estudios previamente realizados por los tesistas se decidieron focalizar esta área del deporte donde también anteriormente se tenía poco control sobre las características fisicoquímicas del agua y durante las competencias del deporte los registros de tiempos se realizaban de manera rústicas con reloj en mano, momento de tomar tiempos de llegada para cada competidor brindando un margen de error e insatisfacción entre los deportistas.

Por los motivos antes mencionados se decidió por implementar sensores indispensables para tratar de tener un mantenimiento apropiado del agua, e implementar o equipar a esta área deportiva con un visualizador electrónico lo suficientemente visible, implementar sensores que determinen las características óptimas para el uso del agua en el área deportiva, y además paneles de toque para competencias.

En este proyecto se diseñaron circuitos electrónicos para matrices y segmentos de diodos led (diodos emisores de luz) para ser adaptados en placas montables para el visualizador electrónico, paneles de toque subacuáticos y sensores programables en IDE de arduino para el estado del agua que toman parte de este proyecto. La comunicación de los sensores de paneles de toque y los sensores que determinan el estado (característica fisicoquímica) del agua es de manera inalámbrica con la tarjeta wemos mini D1 que incorpora el chip Wifi ESP8266 mediante la norma IEEE 802.n., mismo que enviará los datos a la unidad de procesamiento central en este caso es la raspberry pi 3, la misma que trabaja en conjunto con varias tarjetas arduino de familia nano para lograr el funcionamiento de visualizador electrónico.

Como novedoso en este proyecto de titulación se planteó crear un servidor http enlazado a una base de datos para poder realizar toda la configuración de los diferentes parámetros de una competencia de natación como por ejemplo los respectivos nombres de los deportistas, estilos de competencia, asignar ubicación de carriles, todos estos se podrán establecer vía web.

1 EL PROBLEMA

1.1 Descripción del problema

En las instalaciones de la Universidad Politécnica Salesiana del Ecuador sede Guayaquil ubicado en el edificio "F" del Campus Centenario, cuenta con grandes espacios físicos y equipos de última tecnología que facilitan el desarrollo académico y científicos de los estudiantes, la parte olvidada cuenta con espacios deportivos para todos los estudiantes de la universidad; canchas sintéticas para futbol, canchas de básquet, vóleibol y piscina.

Se puede observar que en la actualidad los espacios deportivos a bien de ser recreativo y cursos paracadémicos, también se ven vinculados eventualmente en escenarios competitivos, y en el caso especial focalizaremos la piscina que no cuenta con tecnología que mejore la experiencia de este deporte por lo cual se decidió incorporar un visualizador electrónico para cronometrar competencias de natación, tiempos precisos de llegada y estilos de este deporte acuático. Además, información del estado (características fisicoquímicas) del agua, lo cual es importante saber al momento de ingresar a ella.

Por ende, se pretende que la implementación de un visualizador electrónico deportivo para el área de piscina de la Universidad (UPS), sea una herramienta útil para los usuarios (Espectadores, nadadores y operario) que desean llevar la información exacta y detallada de todo lo que ocurre, garantizando una mejor vivencia dentro del encuentro deportivo acuático.

1.2 Antecedente

Los visualizadores electrónicos hoy en día son muy utilizados en todo tipo de evento deportivo o social alrededor del mundo, debido a eso existen un sin número de empresas que se dedican al diseño y manufactura de los mismos, existen diferentes tipos de competencias deportivas donde es indispensable tener un visualizador que muestre lo que acontece en dicha competencia de cualquier evento de deporte concurrido para mejorar la experiencia de los espectadores, competidores y jueces, vale la pena recalcar que antes se utilizaba tableros que necesitaban cambiarlos manualmente mediante el uso de tizas o carteleras con el fin de mostrar los resultados que se acontecían, así mismo al momento de registrar los tiempos registrados por los competidores que anteriormente se realizaban con reloj en mano, pero hoy en día esto ha mejorado gracias a los visualizadores electrónicos donde tan solo se necesita acceso a la red o en su defecto a un dispositivo smartphone para realizar cambios mediante una app móvil o una interfaz web.

La Universidad Politécnica Salesiana sede Guayaquil dentro de su infraestructura se sabe que solo cuenta con visualizadores electrónicos en las canchas de fútbol y básquet, pero no así en el área de piscina, lo cual genera una necesidad por parte

de organizadores de competencias deportivas acuáticas el poder gestionar, administrar y visualizar sus eventos deportivos.

1.3 Importancia y alcance

El trabajo de titulación y su implementación en las instalaciones de la Universidad Politécnica Salesiana con su sede en Guayaquil es de suma importancia porque se busca facilitar y dotar de tecnología de un visualizador electrónico así creando una plataforma (software y hardware) para gestionar competencias de natación, asignando parámetros importantes tales como nombre del competidor, estilos de natación entre otros, todo esto mediante una interfaz web interactiva. Se decidió focalizar la sección de deporte acuático en la piscina ya que previamente no contaba con la misma, dando así poca certeza en competencias que se suscitaban en ella, debido a esto se construyó un visualizador con la finalidad de proyectar el cronometraje de tiempos en competencias deportivas garantizando que las personas amantes del deporte acuático tengan una mayor experiencia en vivo y que disfruten de futuras competencias, y para complementar el cronometraje antes mencionado que es el objetivo principal del proyecto, se decidió implementar sensores que estén constantemente obteniendo datos del estado físico-química del agua precautelando la salud de los deportistas y estos datos estarán disponibles para ser visualizados, se debe enfatizar que todos estos datos adquiridos tales nombres de competidores y tiempo de marcación para posteriormente ser almacenados y visualizados en un servidor de base de datos.

1.4 Delimitación

La implementación del presente proyecto técnico de titulación fue realizado en instalaciones del área de piscina de la Universidad Politécnica Salesiana sede Guayaquil, edificio F que se encuentra ubicada en Gral. Francisco Robles entre Limberg y 37 SE.

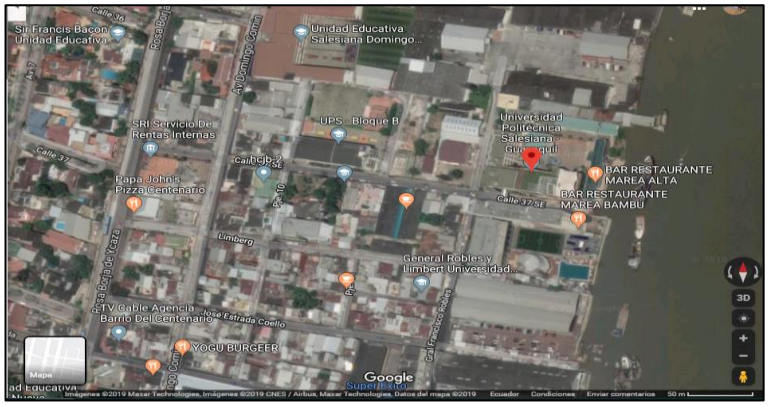


Figura 1 - Ubicación geográfica de la U.P.S. (Google Maps, 2019).

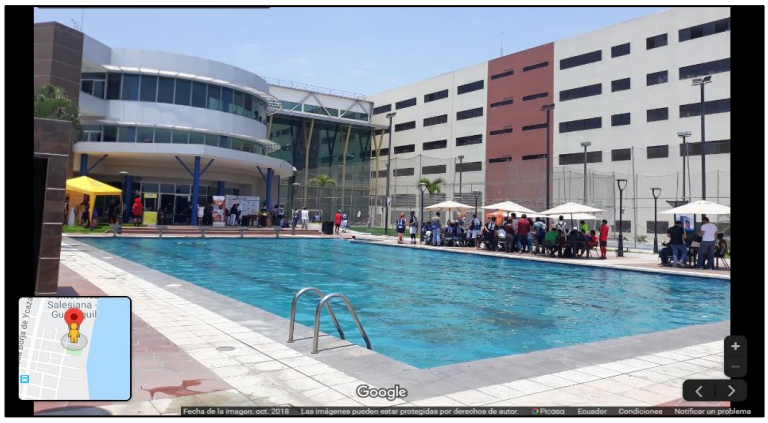


Figura 2- Ubicación de la implementación en la U.P.S. (Google Maps, 2019).

1.5 Delimitación del Problema

El Proyecto se desarrolló pensando en los estudiantes, docentes, deportistas de la Universidad Politécnica Salesiana y en especial para los espectadores amantes del deporte acuático, los cuales podrán visualizar tiempos exactos de competencias, los parámetros del estado del agua, todo esto para mantener informado al espectador mientras dure la competencia.

El administrador puede asignar, editar y eliminar información de competidores y datos de competencia acuáticas a través de la aplicación web dinámica orientada al usuario, almacenadas por el servidor web (Raspberry), el sistema también cuenta con dispositivos electrónicos destinados a sensar parámetros fundamentales de las características del agua (pH, temperatura y turbidez), los cuales registran datos para posteriormente procesarlos, enviarlos para la visualización.

Se instaló placas metálicas de toque implementados con sensores piezoeléctricos para registrar precisamente el orden de llegada.

1.6 Objetivos

1.6.1 Objetivo General

Diseñar e implementar un sistema electrónico para cronometrar competencias de natación y supervisión de estado del agua, mediante un visualizador electrónico para la piscina de la universidad politécnica salesiana sede Guayaquil.

1.6.2 Objetivos Específicos

- Diseñar un circuito electrónico para los arreglos y segmentos de leds que se requieren para el visualizador electrónico.
- Diseñar los paneles subacuáticos de toque mediante el uso sensores piezoeléctricos y componentes electrónicos, que permitirán determinar el tiempo de llegada de cada uno de los competidores.
- Programar los sensores de estado del agua (pH, turbidez y temperatura), basado en el entorno de la tarjeta arduino.
- Crear una base de datos MySQL en la raspberry pi 3 para el registro de los resultados de competencias que se ejecutan
- Desarrollar una ventana de entorno ejecutable en la tarjeta raspberry pi 3, para tener control y configuración del visualizador electrónico.
- Visualizar la información detallada de los resultados de tiempo cronometrado, estilo de natación, nombre de los nadadores y estado del agua, todos estos parámetros mostrados en tiempo real.
- Construir el armazón del visualizador electrónico e implementarlo en las instalaciones de la universidad para beneficio de la comunidad.

2 MARCO TEÓRICO

2.1 Raspberry Pi

La Raspberry Pi es una minicomputadora por sus dimensiones física con un bajo costo que puede ser adaptable a un monitor de computadora o TV gracias a una salida HDMI (High-Definition Multimedia Interface, interfaz multimedia de alta definición por sus siglas en ingles), que además se puede utilizar un teclado y mouse estándar debido a que posee dos puertos USB. Este pequeño dispositivo electrónico permitir a personas de todas las edades explorar en la informática y aprender a programar en lenguajes como Python. Está minicomputadora es capaz de realizar todas las tareas que un programador o una persona común se esperaría que computadora de escritorio hiciera, desde navegar por Internet mediante conexión inalámbrica y alámbrica ya que consta de una antena wifi y un puerto Rj45, además reproducir videos de alta definición, hasta hacer hojas de cálculo, procesamiento de textos, bluetooth y juegos. A la placa se la energiza de manera sencilla y fácil gracias a un puerto micro USB.

Además, la Raspberry Pi con sus 40 puertos de propósito general entrada/salida (GPIO) permite tener la capacidad de controlar e interactuar con el mundo exterior, permitiéndonos así utilizar una extensa gama de proyectos sean escolares o de desarrolladores digitales experimentados tales como crear desde máquinas de música y detectores de padres hasta estaciones meteorológicas. Unos de los principales objetivos de raspberry como organización es que su hardware Raspberry Pi sea utilizado por niños de todo el mundo para aprender a programar y comprender cómo funcionan las computadoras.

2.2 Raspberry Pi 3

Es un miniordenador con un procesador BCM2837 siendo el primero en ofrecer soporte de 64 bits, integrado un soporte inalámbrico de doble banda con un radio capaz de conectarse a redes Wifi de 2,4 GHz y dispositivos Bluetooth, además de un encabezado de 40 pines GPIO.

Útil para el desarrollo de nuestro proyecto de titulación, el cual permitirá gestionar todas las funcionalidades y el control de todos los elementos electrónicos del visualizador, de modo que se necesita tocar un botón en tu smartphone para dar inicio al funcionamiento del visualizador electrónico.

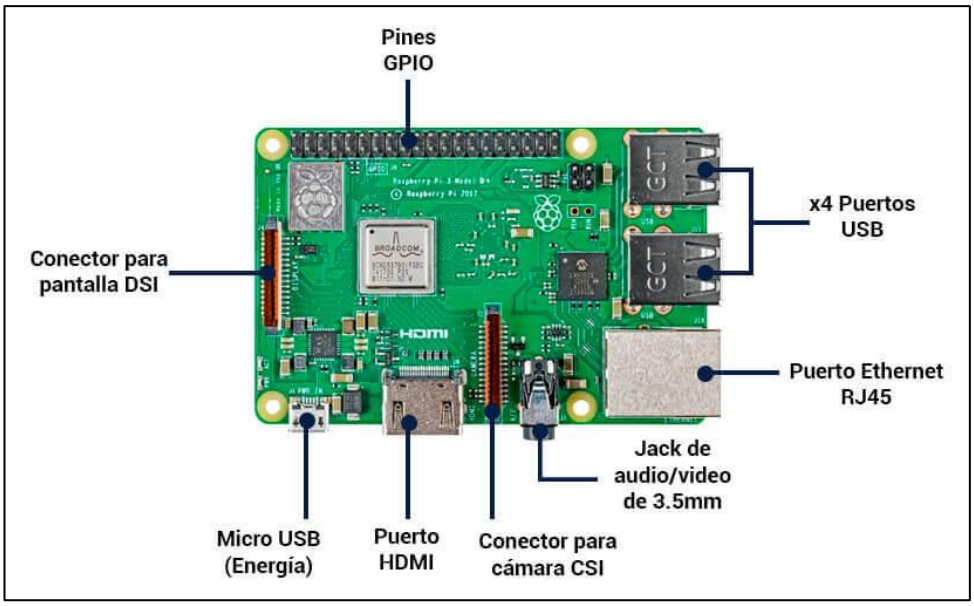


Figura 3- Raspberry Pi 3. (Molloy, 2016).

2.3 Arduino

El famoso Arduino en el mundo de los proyectos electrónicos, Arduino básicamente es una plataforma de hardware a costos muy accesibles y software (IDE) de código abierto, Arduino con su IDE pueden ser programados desde cualquier sistema operativo ya sea Windows, Mac, Linux. Por estas principales ventajas que nos ofrece de accesibilidad y capaz de adaptarlo a cualquier tipo de proyecto electrónico fue elegido para el proyecto de titulación, el Arduino se diseñó y fabricó mediante una placa que incorpora un excelente microcontrolador, entre los más comunes el Atmega168, Atmega328, Atmega1280, ATmega8, la programación de cada una de las diferentes placas es súper sencilla y es realizada mediante su entorno de desarrollo integrado (IDE), flexible y fácil de utilizar para cualquier pequeño programador y desarrollador.

Figura 4- Arduino Uno. (Arduino, 2008).

2.4 IDE Software de ARDUINO

Es el entorno de desarrollo integrado, llamado así IDE por sus siglas en inglés (Integrated Development Environment), el IDE es un entorno de programación que ha sido desarrollado como un programa de aplicación; es decir, que realiza tareas en específico que consisten en un editor de código, un compilador, depurador y un constructor de interfaz gráfica (GUI). También en el caso de arduino que incorpora las herramientas para cargar el programa ya compilado en la memoria flash del hardware (placa Arduino tales como UNO, MEGA, NANO) a través del bootloader. El software de Arduino está disponible para poder ser instalado y ejecutado desde varios tipos de sistemas operativos como, Linux, Windows, Mac OS.

2.5 Arduino nano

El Arduino Nano es una pequeña, completa y amigable placa basada en el chip ATmega328P; ofrece la misma conectividad y especificaciones de la placa UNO en un factor de forma más pequeño.

El Arduino Nano se programa utilizando el Software Arduino (IDE), nuestro entorno de desarrollo integrado común a todas nuestras placas y que se ejecuta tanto en línea como fuera de línea.

(Arduino, 2019)

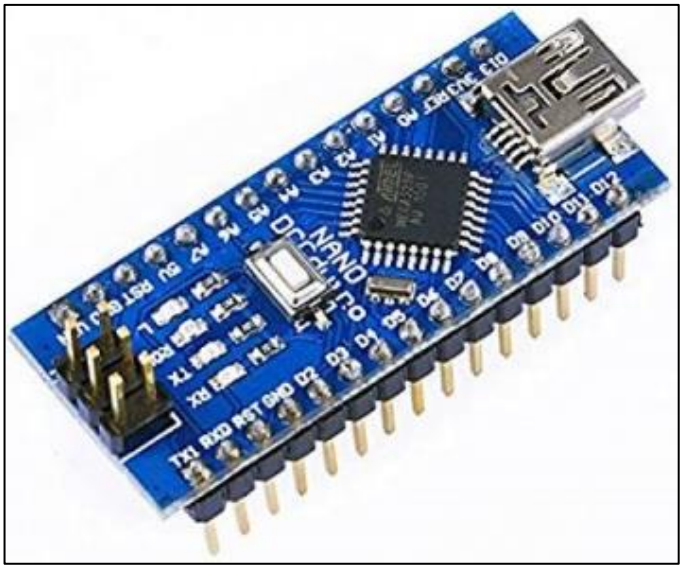


Figura 5- Arduino nano. (Arduino, 2019)

2.6 Wemos D1 mini

Es una pequeña placa de desarrollo basada en un popular chip wifi ESP8266 programable con el IDE (entorno de desarrollo integrado), de bajo costo que utiliza un protocolo TCP/IP, el Wemos mini consta con las siguientes dimensiones de su placa de 34.2mm x 25.6mm que la convierte a esta placa una versión de menor tamaño a comparación de la Wemos D1. Además, su peso es de unos 3g. casi imperceptible

Se decide elegir esta mini placa wifi en este proyecto ya que es ideal porque se necesita poco espacio y de una conexión inalámbrica, además aporta con muchas más ventajas como poder incorporar un puerto Micro SD y conversor serial, cuenta también con un regulador de voltaje que permite alimentarlo a 5v de corriente directa, una fash memory de 4Mbytes, potenciado con un procesador de 80MHz/160MHz, y por último la Wemos mini es fácil de programar con el mismo IDE de Arduino.

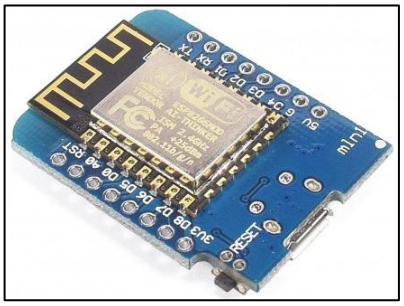


Figura 6- Wemos D1 mini.

2.7 Turbidez

La turbidez es el efecto que causa un aspecto de nubosidad o reducción de la transparencia de un fluido en este caso el agua de la piscina causada por elementos o partículas duras (sin disolver) presentes, siempre en movimiento que por lo general son imperceptibles a simple vista. Entonces se puede decir sin temor a equivocarse que una señal de contaminación del agua es la turbidez ya que implica la presencia de varias sustancias solidas o microorganismos que pueden causar daño a la salud pública o causar algún tipo de interferencia en procesos industriales de manufacturas.

Los niveles de turbidez más altos a menudo se asocian con niveles más altos de virus, parásitos y algunas bacterias porque a veces pueden adherirse a la tierra en el agua. Por lo tanto, debemos tener cuidado con la turbidez del agua ya que generalmente tiene más patógenos, por lo que realizar algún tipo de actividad o deporte en ella aumentan las posibilidades de enfermedades.

Figura 7- Turbidez del agua de una piscina. (poolspanews,2010)

2.8 Temperatura

La temperatura en los líquidos básicamente se refiere al grado de calor o frío y puede ser medidos en grados Celsius (Unidad de medida térmica derivada del sistema internacional de unidades). La temperatura en el agua necesita ser monitoreado regularmente como dentro de un rango tolerable, ya que por efectos de la temperatura las enfermedades y el estrés se vuelven más evidentes en las personas. La temperatura perfecta del agua de una piscina va a depender de varios factores tales como su ubicación, características y el uso que se le da a la misma. Entonces podemos asegurar que en el agua de una piscina al aire libre (descubierta) la temperatura debería de estar dentro del rango de unos 28 a 30 grados Celsius.

Además, la temperatura del agua de una piscina es un factor físico de suma importancia ya que cuando la temperatura del agua llega a ser muy elevada, el PH que es un factor químico muy importante se ve alterado, y por ende esto afecta de manera directa a condiciones de salubridad, y por lo consiguiente se concurre en poner en riesgo la seguridad de las personas y futuros competidores.

2.9 Medida de acidez o alcalinidad en líquidos (pH)

El pH corresponde a una de las características químicas presentes en los líquidos, en nuestro proyecto en el agua de piscinas, además, el pH es un parámetro clave alrededor de la cual se equilibran todos los demás componentes químicos presentes en el mantenimiento del agua de una piscina y, debido a esto, tiene una relación de manera directa con la turbidez. El agua a un pH más bajo tendrá una mayor capacidad para disolver minerales. El impacto en la turbidez radica en los cambios en el pH. Por ejemplo, si el agua alcanza el equilibrio a un pH de 7.2 pero luego se eleva a un pH de 8.0, el agua se saturará en exceso con minerales disueltos. El

exceso de minerales se volverá insoluble y precipitará del agua como sales metálicas, óxidos metálicos o compuestos a base de calcio que pueden adherirse a las superficies y formar escamas. La única forma de eliminar estas partículas es la redisolución mediante la reducción del pH o la eliminación mediante filtración.

(Karen Rigsby,2012)

Se debe especificar que a temperaturas muy elevadas el pH del agua de la piscina se ve alterado directamente y afecta a condiciones de salubridad volviéndose más turbia dando las condiciones perfectas para la proliferación de algas o microorganismos posiblemente patógenos, el rango aceptable de pH debe de mantenerse dentro del rango de 7,2 y 7.6, si existen variaciones de alza o baja afectan a la acidez o la alcalinidad del agua, donde cuando se exceden los valores establecidos, se pueden producir diferentes afectaciones a la integridad de las personas como por ejemplo irritaciones oculares y picazón en la piel.

2.10 Material piezoeléctrico

Un material piezoeléctrico tiene la capacidad de convertir una tensión mecánica es decir empujar, presionar o apretar en una diferencia de potencial eléctrico (carga eléctrica) a este evento particular se le denomina efecto piezoeléctrico. El efecto piezoeléctrico es efecto reversible, lo que significa que cuando aplicamos tensión mecánica al material piezoeléctrico, obtenemos algo de carga eléctrica en la salida. Igual que cuando alimentamos la carga eléctrica al sensor, se estira o comprime.

Es una tarea simplemente sencilla usar un sensor piezoeléctrico, basta con solo conectar el terminal positivo y negativo a un circuito electrónico y hay que presionar la parte superior del sensor. Al momento de presionar, debido a la presión mecánica, se crea una diferencia de potencial eléctrico en la salida. Simplemente puede conectar un LED al sensor piezoeléctrico, como se muestra en el circuito a continuación. Cada vez que presiona el sensor, el LED emitirá un destello.

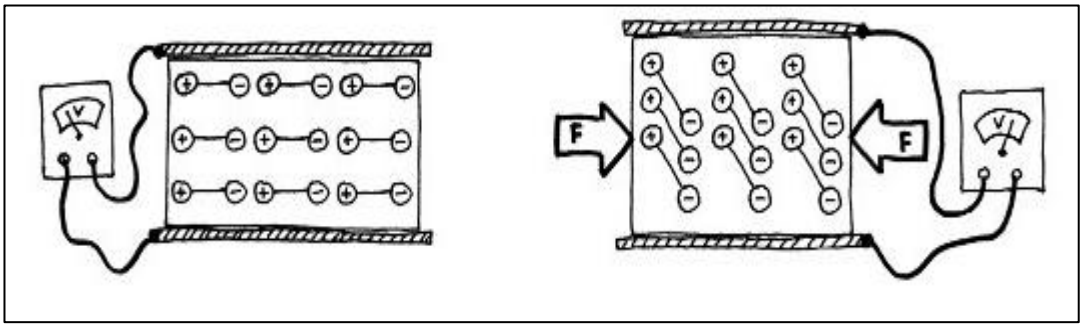


Figura 8- Efecto piezoeléctrico (material piezoeléctrico comprimido)
(www.openmusiclabs.com)

2.11 Matrices de diodos emisores de luz

Básicamente una matriz LED es un arreglo de diodos emisores de luz, dispuestos en filas y en columnas. El Led al igual que otros diodos, la electricidad fluye a través de el en una única dirección del ánodo (terminal positiva +) hacia el cátodo (terminal negativa -); al efectuarse este fenómeno físico de la electricidad se produce la iluminación del diodo.

Al organizar los ánodos (terminal positivo) y los cátodos (polo negativo) de una manera en particular, podemos lograr una matriz y gracias a estas características podemos solicitar a cada LED trabaje individualmente enviando señales de voltajes altos y bajos desde un dispositivo Arduino.

Las matrices led por lo general vienen en dos tipos de arreglos, ánodo o cátodo común.

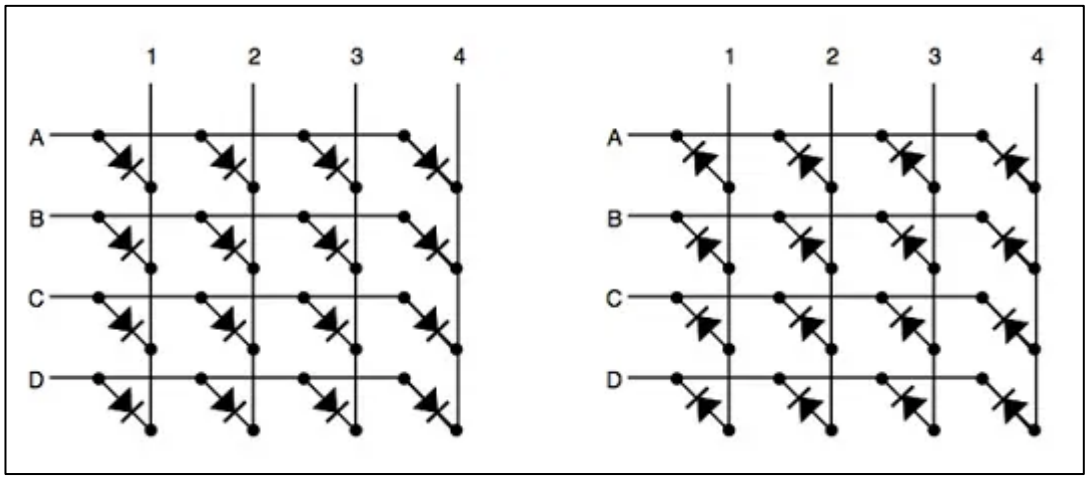


Figura 9- Arreglo de matrices con Leds Ánodo y cátodo común respectivamente

2.12 IC (circuito integrado) MAX7219

El circuito integrado MAX7219 multiplexador de caracteres y dígitos de bajo costo, es idealmente diseñado para poder controlar una MATRIZ de diodos LED 8x8 (8 filas X 8 columnas) o hasta un máximo de 64 LEDs de forma individual, y que solo requiere de una resistencia externa para la conexión de los 64 LEDs. El IC controla a los Display de diodos con entrada en serie con conexión de cátodo común (negativo en común), además este microcontrolador se puede conectar con display's LED numéricas de 7 segmentos de hasta 8 dígitos.

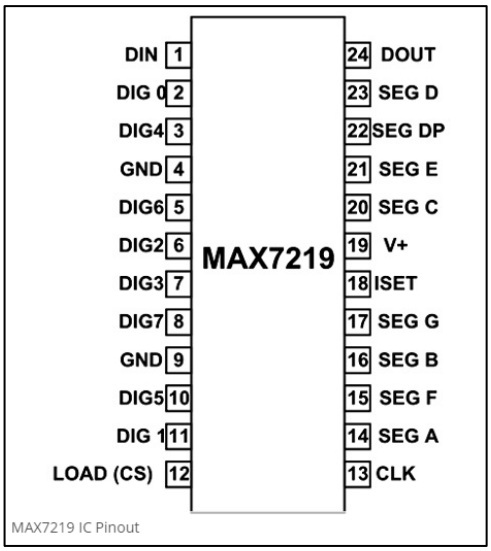


Figura 10- MAX7219. (www.components101.com)

2.13 Aplicación web

2.13.1 Herramienta de editor de texto

Para el desarrollo de nuestra interfaz web dinámica, utilizamos una herramienta accesible para cualquier desarrollador web un Editor de texto, ya que es un software que fue pensado y desarrollado con el principal objetivo de ayudar a escribir código fuente que posee en diversos lenguajes de programación para el desarrollo de diferentes tipos de aplicaciones.

Los editores de código trabajan comúnmente con diferentes tipos lenguajes de programación de alto nivel al mismo tiempo tales como Python, HTML, JavaScript entre otros, se usó un editor de texto porque nos ofrece ayudas en sintaxis al momento de escribir o la visualización del código de programación de las aplicaciones.

2.13.2 Software Visual Studio Code

El editor de texto más usado para escribir y editar código fuente en el desarrollo de aplicaciones en los diferentes lenguajes de programación es Visual Studio Code, se decidió elegir y adaptarlo a nuestro proyecto de titulación para escribir el código fuente de la interfaz web, es posible instalarlo y ejecutarlo en los diversos sistemas operativos en nuestro caso WINDOWS y LINUX con su distribución para el raspberry Raspbian.

Visual Studio Code consta con soporte integrado para escribir texto en HTML, CSS, JavaScript, Vue.Js y Node.js para la programación del frontend, y además de proveer extensiones para otros lenguajes de programación tales como C++, C#, Java, Python y sus frameworks, PHP para la programación del backend.

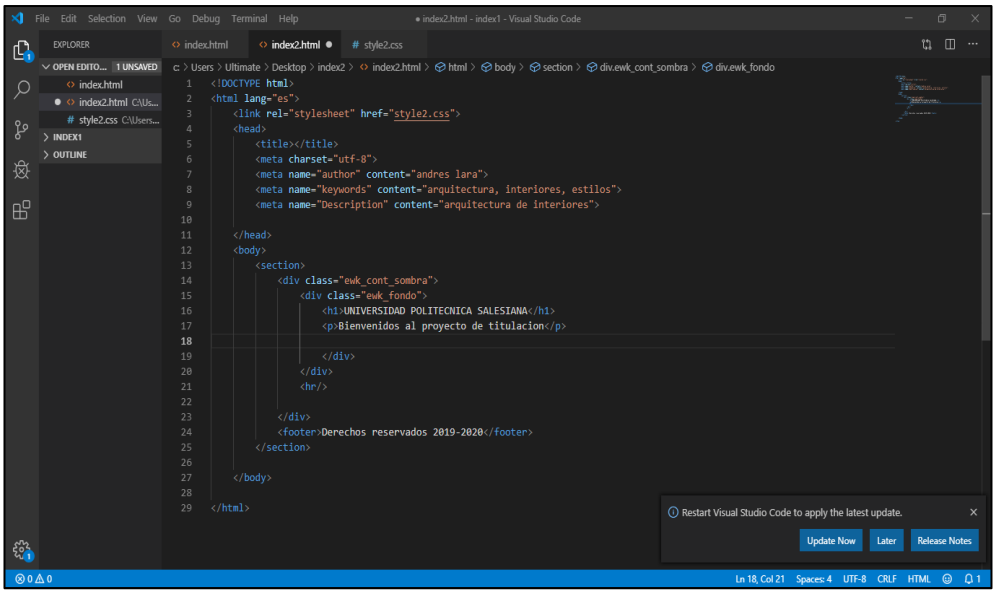


Figura 11- Visual Studio Code.

2.13.3 Lenguaje de marcado HMTL

HTML no es un lenguaje de programación, sino que es un lenguaje de marcado que define la estructura de su contenido (como quieres que se vea en la ventana de un navegador web). HTML consiste en una serie de elementos representados por etiquetas, que se usan para encerrar o ajustar diferentes partes del contenido para que parezca de cierta manera o actúe de cierta manera. Las etiquetas que encierran pueden hacer un hipervínculo de una palabra o imagen a otro lugar, pueden poner en cursiva las palabras, pueden hacer que la fuente sea más grande o pequeña, etc.

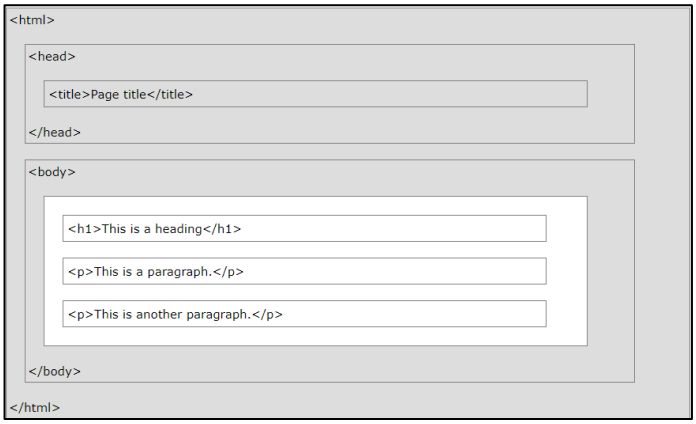


Figura 12- Estructura básica de documento HTML. (www.w3schools.com.)

2.13.4 CSS

Las siglas CSS significa hojas de estilo en cascada y describe cómo se deben mostrar los elementos HTML en la pantalla, papel o en otros medios

CSS ahorra mucho trabajo ya que puede controlar el diseño de varias páginas web a la vez y estos diseños se almacenan en archivos CSS.

2.13.5 Vue.js

Los desarrolladores web están utilizando diferentes herramientas y técnicas para lanzar aplicaciones exitosas. Los desarrolladores también están utilizando muchas tecnologías de software para hacer que las aplicaciones sean más rápidas, más atractivas y fáciles de usar. Vue.js es una de esas nuevas tecnologías de software que se utilizan ampliamente en todo el mundo para el desarrollo web. Vue.js es en realidad un framework (plantilla) de JavaScript con varias herramientas opcionales para construir interfaces de usuario.

Las ventajas de utilizar vue.js en desarrollos de aplicaciones web son los siguientes: fácil de entender y desarrollar aplicaciones web, simplicidad de integración por ser popular entre los desarrolladores, flexibilidad.

2.14 Sistema Operativo

Un sistema operativo es un conjunto de programas y utilidades básicas que hacen que su computadora funcione. El centro de un sistema operativo es el núcleo (kernel). El núcleo es el programa más importante en la computadora, realiza todo el trabajo básico y le permite ejecutar otros programas.

(D*ebian,2019)*

2.14.1 Raspbian

Raspbian es un sistema operativo libre basado en Debian (Distribución de GNU/Linux) potenciado para la poderosa minicomputadora Raspberry Pi. Un sistema operativo es el conjunto de programas básicos y utilidades que hacen que su hardware Raspberry Pi funcione. Sin embargo, Raspbian proporciona más que un sistema operativo puro: viene con más de 35,000 paquetes, software precompilado incluido en un formato agradable para una fácil instalación en su Raspberry Pi.

(Raspbian,2019)

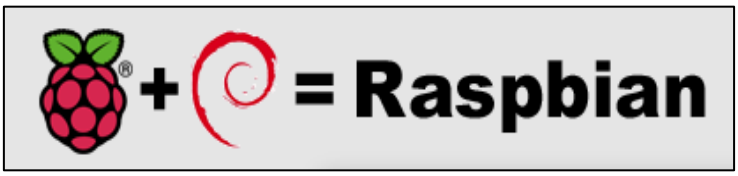


Figura 13- Raspbian. (www.raspbian.com.)

En la actualidad se encuentran disponibles dos versiones del sistema operativo Raspbian que pueden ser instaladas en el hardware de la Raspberry Pi que las mencionaremos a continuación tales como la Raspbian Pixel y Raspbian Lite; donde la primera versión es completa cuenta con un entorno grafico (escritorio con menús, ventanas, iconos, fondo de pantalla, etc.) y en la segunda versión viene reducida es decir sin entorno grafico es decir se puede navegar solo en modo consola (sin gráficos). Particularmente se utiliza la versión Pixel para adaptarla a este proyecto de titulación.

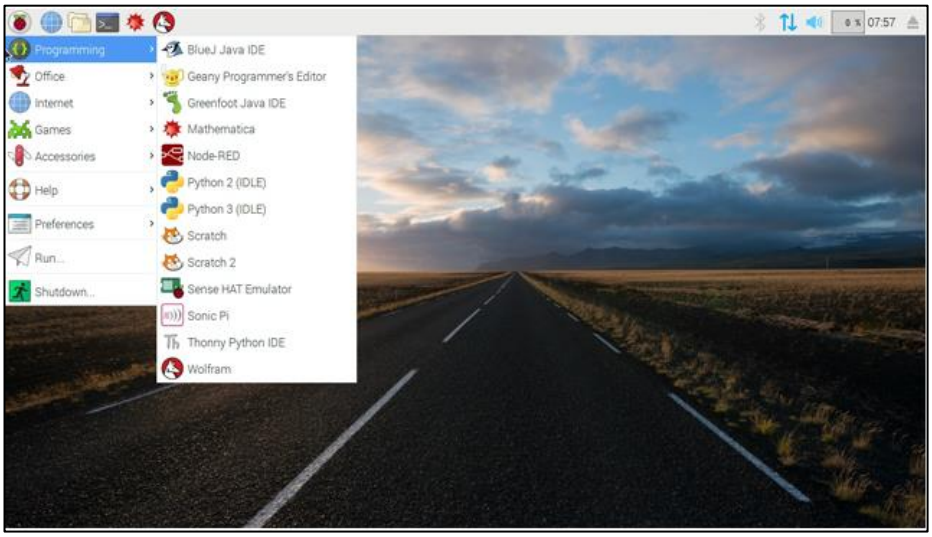


Figura 14- Sistema Operativo Raspbian para Raspberry.

2.15 Servidor Web

También conocido como http server (HyperText Transfer Protocol) se refiere a hardware y software, en donde ambos trabajan en conjunto, por lo tanto en el lado del hardware el servidor http es básicamente un computador en el cual se pueden almacenar el software del servidor y los componentes de los archivos de los sitios web (documentos con textos HTML, imágenes, Hojas de estilos CSS y archivos JS), el computador que almacena todos estos documentos debe tener una conexión a la red como el internet, y que puede intercambiar datos con otros dispositivos conectados dentro de una misma red como el internet.

Ahora en el lado del software del servidor web, este controla como los usuarios web acceden a los archivos almacenados. El software http entiende URLs (direcciones web) y HTTP (protocolo que nos permite realizar peticiones de archivos webs).

El servidor web funciona cuando un cliente a través de un buscador como google necesita acceder a un archivo alojado en el servidor web, el buscador realiza la petición del archivo mediante el protocolo HTTP enviando una petición get, entonces cuando la petición encuentra al servidor correcto (hardware), el software del servidor http acepta y procesa dicha petición, una vez que encuentra el documento web (Html, imágenes, CSS y Js, etc.), lo envía de vuelta al computador del buscador donde se realizó la petición, de igual forma a través del protocolo HTTP enviando un método llamado POST, se tiene que aclarar que si no es posible encontrar el archivo que es pedido, entonces el servidor igual responde con un documento web en el cual se visualiza un mensaje de error 404 que es muy común, tan común que hoy en día muchos desarrolladores web pasan mucho tiempo diseñando este tipo de páginas.

2.15.1 Apache Server

Un servidor Apache es un tipo de servidor web o servidor HTTP gratuito y de código libre, además de ser el servidor HTTP número uno en el internet y con la capacidad de poder ser instalado y ejecutado en diversos tipos diferentes de sistemas operativos, además de ser muy complejo que destaca por ser seguro y posee un gran rendimiento.

El servidor Apache HTTP puede almacenar tanto páginas web estáticas basado en HTML como aplicativos web o web dinámicas, tiene la capacidad de poder adicionar módulos especiales para utilizar diversos tipos de lenguajes como PHP.

Apache Software Foundation es el encargado del desarrollo del servidor Apache como un proyecto HTTP server.

Ventajas

- Software de código abierto para instalación y/o configuración.
- El servidor web Apache HTTP es de coste gratuito.
- Funcional y Soporte por parte de individuos que aportan con mejoras que son disponibles.
- Puede ser instalados y ejecutados en muchos sistemas operativos informáticos.
- Posee un rendimiento alto para la gestión de más de un millón de visitas por día.
- Posee un soporte de seguridad SSL y TLS.

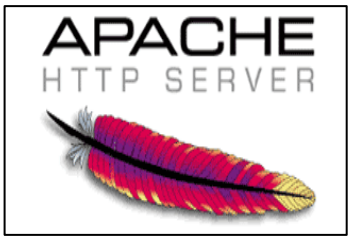


Figura 15- Apache HTTP server. (Acción Network Global SL, 2016)

2.16 Software Python

2.16.1 Introducción

Python es un lenguaje de programación back-end interpretado de alto nivel orientado a objetos. Python es el lenguaje de programación más popular hoy en día debido a que su código es mucho más fácil de entender, y a su gran número de librerías disponibles que nos facilitan la programación. Python es accesible (Código libre), además tiene como característica que puede ser instalado y ejecutado desde cualquier sistema operativo como Linux con sus distribuciones o Windows.

El software Python generalmente instalado por defecto en el sistema operativo Raspbian que utilizan el hardware raspberry.

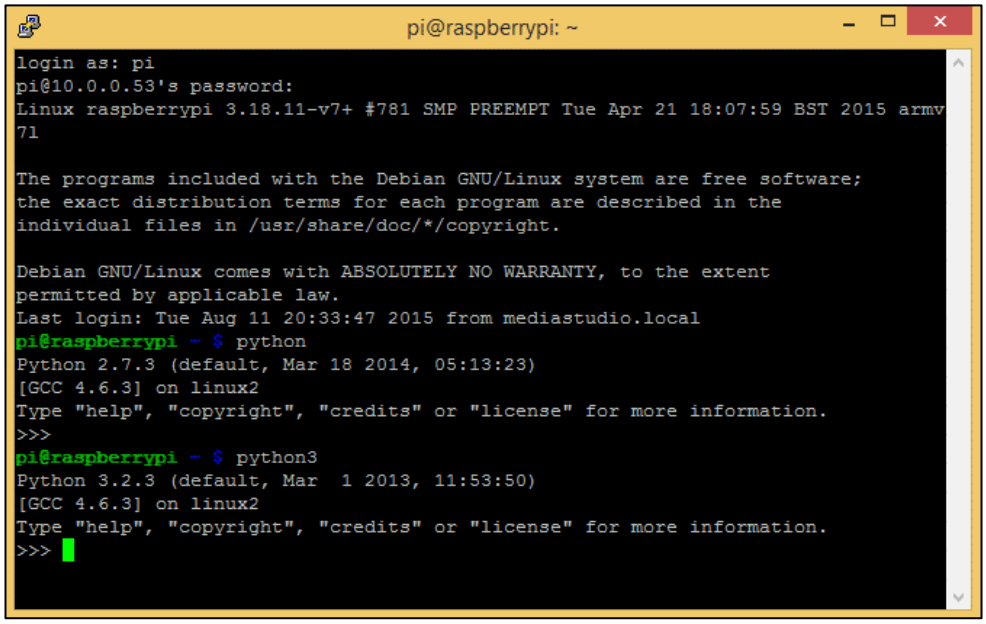


Figura 16- Python. (www.circuitbasics.com).

Características

- Lenguaje de programación creado para realizar todo tipo de programas.
- Lenguaje adaptable a muchos sistemas informáticos.
- Interpretado, no se necesita una compilación compleja del programa para poder ejecutarlo, su compilación es transparente para el programador.
- Interactivo al producir un resultado visible al momento de ejecutar la programación.
- Orientado a objetos.
- Posee la propiedad de poder realizar funciones además de tener muchas librerías que se pueden importar para tratar temas específicos de una programación.
- Posee una sintaxis sencilla y clara basado en la tabulación como margen de programación.

2.16.2 Formato de trabajo flask

Flask es un framework desarrollado en el lenguaje de programación Python que ofrece simplicidad y potencia, proporciona herramientas, bibliotecas y tecnologías que permiten crear una aplicación web de manera más rápida, además permite evitar el código repetitivo, además de desarrollar aplicaciones web avanzadas. Estas aplicaciones web pueden ser páginas web o blogs.

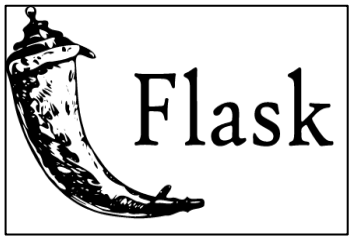


Figura 17- Framework Flask basada en Python. (www.medium.com)

2.17 Software MySQL

2.17.1 Introducción

MySQL es una plataforma de gestión de bases de datos (Database Management System, DBMS) para bases de datos relacionales. MySQL es un aplicativo desarrollado para la administración de archivos.

MySQL es la opción perfecta para proporcionar datos a través de Internet debido a su capacidad para manejar cargas pesadas y sus medidas de seguridad avanzadas.

Existen varios tipos de bases de datos, como un archivo de almacenamiento hasta sistemas orientados a objetos.

MySQL tiene la propiedad de ser de código abierto, MySQL es la construcción de base de datos que permite que PHP y Apache trabajen juntos para acceder y mostrar datos en un formato legible para un navegador. Es un servidor de lenguaje de consulta estructurado diseñado para cargas pesadas y procesamiento de consultas complejas. Como un sistema de base de datos relacional, MySQL permite que muchas tablas diferentes se unan para lograr la máxima eficiencia y velocidad.

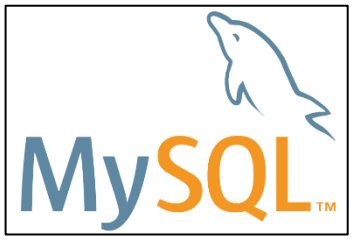


Figura 18- Programación de Base Datos en MySQL.

2.17.2 Administración de la base de datos mediante phpMAdmin

Para que sea más fácil la visualización, el entendimiento y la administración de toda una base de datos y sus respectivas tablas se procedió a adaptar una interfaz web de código abierto llamada PhpMyAdmin, además de ser segura debido a que incorpora una autenticación para el usuario garantizando seguridad.

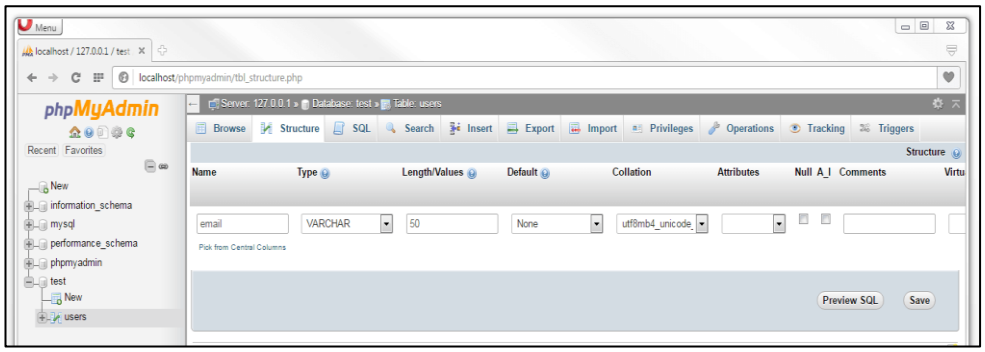


Figura 19- Interfaz Gráfica de MySQL mediante phpMyAdmin.

Características

- Asociados con aplicativos webs y la publicación en línea
- Es un administrador de base de datos en multihilo y multiusuario, lo que permite ser utilizado por múltiples usuarios a la vez y realizar varias consultas al mismo tiempo.
- Puede ser ejecutada en diferentes sistemas operativos.
- Es una marca registrada y es de código abierto.
- Ofrece aplicaciones que permiten acceder a las sentencias del gestor de base de datos

2.18 Router inalámbrico tp-link TL-WR840N

El router 300Mbps Wireless N TL-WR840N es un dispositivo combinado con cable/inalámbrico de conexión de red diseñado específicamente para las necesidades de pequeños ambientes en empresas y oficinas domésticas de redes. El TL-WR840N crea un rendimiento inalámbrico excepcional y avanzado, lo que hace ideal para streaming de video de alta definición. Además, posee Wi-Fi Protected Setup (WPS), WPA2 para la prevención de la red de intrusiones externas.

(TP-LINK,2020)

El router TL-WR840N que es de fácil configuración ya que incorpora una interfaz de usuario rápido y sencillo de entender, además posee la característica de una velocidad inalámbrica de 300Mbps para descargas de archivos grandes que satisface necesidades de redes exigentes, y entre sus características más relevantes es que posee IP QoS un control de ancho de banda esto permite administra la red interna de tal forma que se pueda determinar que ancho de banda se le asigna a cada usuario (IP), el router es compatible con IEEE 802.11b/g/n se puede añadir que viene dotado de tecnología CCA (Clear Channel Assessment) que ofrece una señal estable y que eleva el rendimiento de la señal inalámbrica.

Figura 20- Router TL-WR840N. (TP-LINK,2019)

3 METODOLOGÍA

3.1 Diseño y elaboración de las placas electrónicas de diodos leds

Para el desarrollo del visualizador electrónico se contó con dos diferentes diseños de placa electrónica de diodos leds, cuales son: display de 7 segmentos y matriz 8x7 de leds.

Primero se diseña la placa del circuito impreso con sus respectivas pistas para lograr las ubicaciones de los diferentes componentes que placa electrónica requiere, mediante la ayuda del programa EAGLE PCB.

Una vez obtenida las pistas del circuito de los dos tipos de placas electrónicas y a su vez con la correcta posición de los elementos electrónicos que la componen a ambas, se procede al corte de las ‘baquelitas’ que están hechas con una materia de fibra de vidrio en dimensiones más pequeñas.

Para el caso de las placas de displays de 7 segmentos cuenta con las dimensiones de la placa de 17,5cm de largo x 7,5cm de ancho.

Figura 21- Corte de placas 7 segmentos.

Y para el caso de las placas de matrices 8x7 cuentan con las dimensiones 13,5 cm de ancho y 17,5 cm de largo. Se muestra su figura a continuación.

Figura 22- Corte de placas de matrices 8X7.

Luego de tener las respectivas placas ya cortadas con sus dimensiones deseadas mencionadas anteriormente para ambos casos, procedemos a serigrafiar, lo cual necesitaremos de varios pasos a seguir:

Paso 1. Limpiamos el área de la placa de cobre con un lustre de aluminio, luego se enjuaga con detergente, con el fin de lograr que la placa no tenga ninguna suciedad y poder colocar el circuito impreso (PCB) sin ningún problema.

Figura 23- Limpieza de placas.

Paso 2. Procedemos con el serigrafiado del circuito impreso (PCB). Teniendo la PCB impresa en papel fotográfico, la colocamos junto a la placa de cobre, a su vez la fijamos bien con cinta térmica para evitar que se mueva, luego de eso hacemos el uso de una plancha casera cual permite generar calor y mediante eso lograr que las partículas de polvo de tinta se fijen en la placa.

Figura 24- Planchado de placas display 7 segmento numérico.

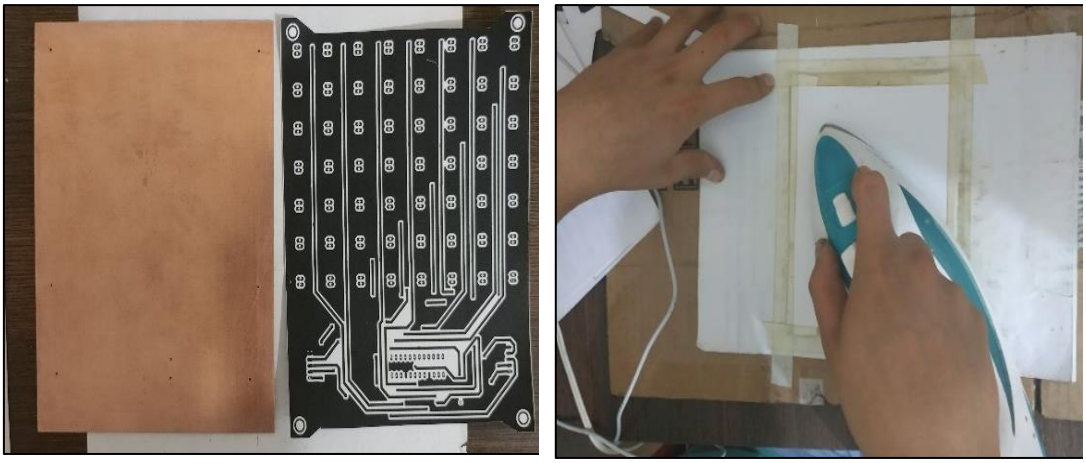


Figura 25- Planchado de matriz 8x7.

En el caso de lo serigrafiado de la placa de 7 segmentos tan solo necesitamos unos 10 min para lograr que se fije en la placa la pista del circuito y para la placa de matriz 8x7 unos 15min, pero tomar en cuenta que en placa de matriz se serigrafía en ambos lados de cara.

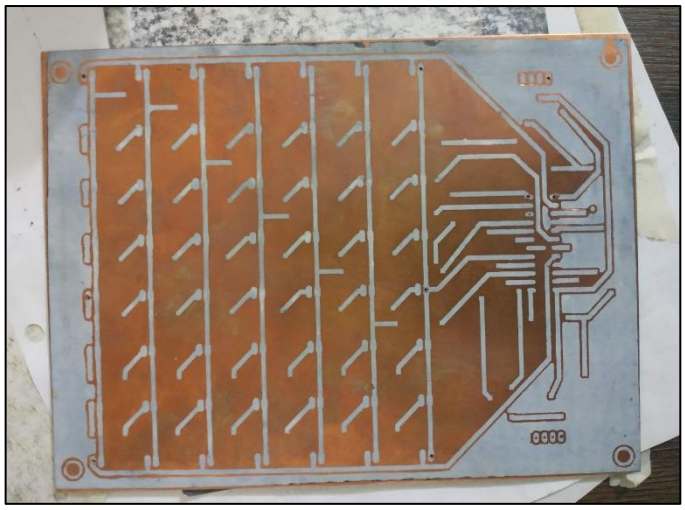


Figura 26- Serigrafiado de matriz 8x7 vista trasera.

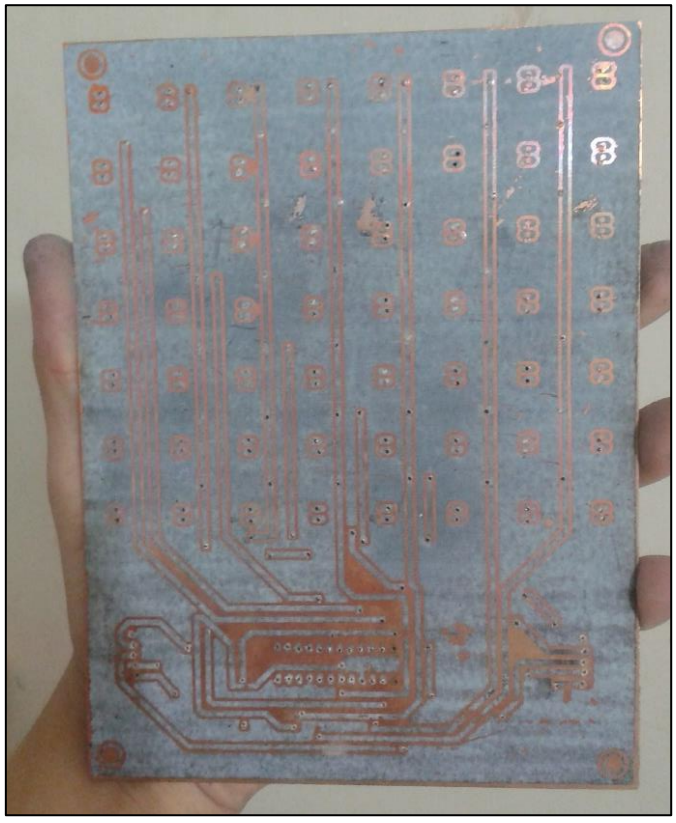


Figura 27- Serigrafiado de matriz 8x7 vista delantera.

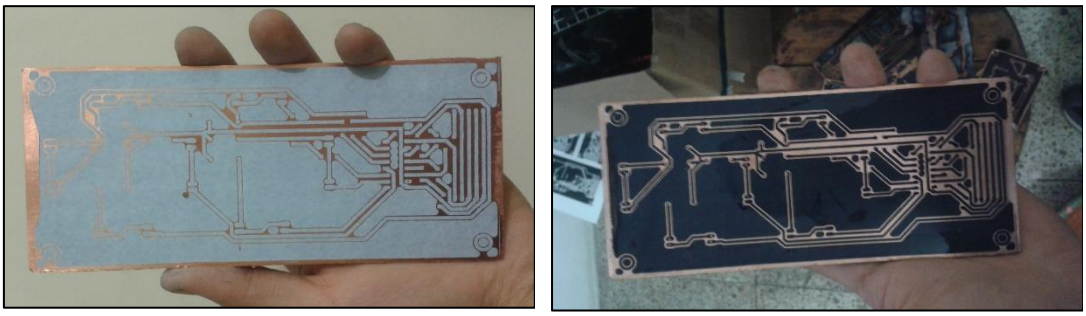


Figura 28- Serografiado de placa display 7 segmentos.

Paso 3. Verificar que el serografiado no tenga zonas en que no se haya adherido la pista, si es el caso mediante un marcador permanente se remarca en la zona afectada.

Procedemos a perforar las placas en sus respectivos agujeros, mediante el uso de un mini taladro, con número de broca de 1mm.

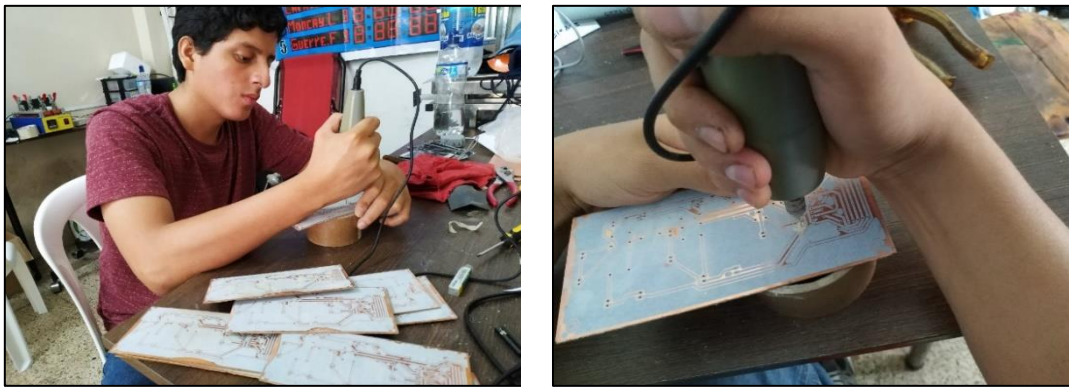


Figura 29- Perforación de placa display 7 segmentos.

Figura 30- Perforación de matriz 8x7 vista frontal.

Figura 31- Perforación de matriz 8x7 vista trasera.

Paso 4. Luego se procede a sumergir la placa en agua, más el ácido de Percloruro de Hierro, el cual se puede comprar en cualquier electrónica. El tiempo que toma el ácido en disolver el cobre no deseado es entre 10 a 15 minutos.

Figura 32- Ácido actuando en placas 7 segmentos.

Figura 33- Ácido actuando en matrices 8x7.

Paso 5. Limpiar con un lustre las placas, para exponer el cobre del circuito impreso. Y proceder a soldar, comenzando en el caso si hay puentes de alambre y luego con los componentes de menor dimensión, tales como los diodos leds, resistencias, pines de conexión y zócalos de circuitos integrados. Es recomendable montar zócalos para los circuitos integrados, dado que, cuando sea necesario cambiarlos en futuros arreglos será fácil en quitar y colocar otro.

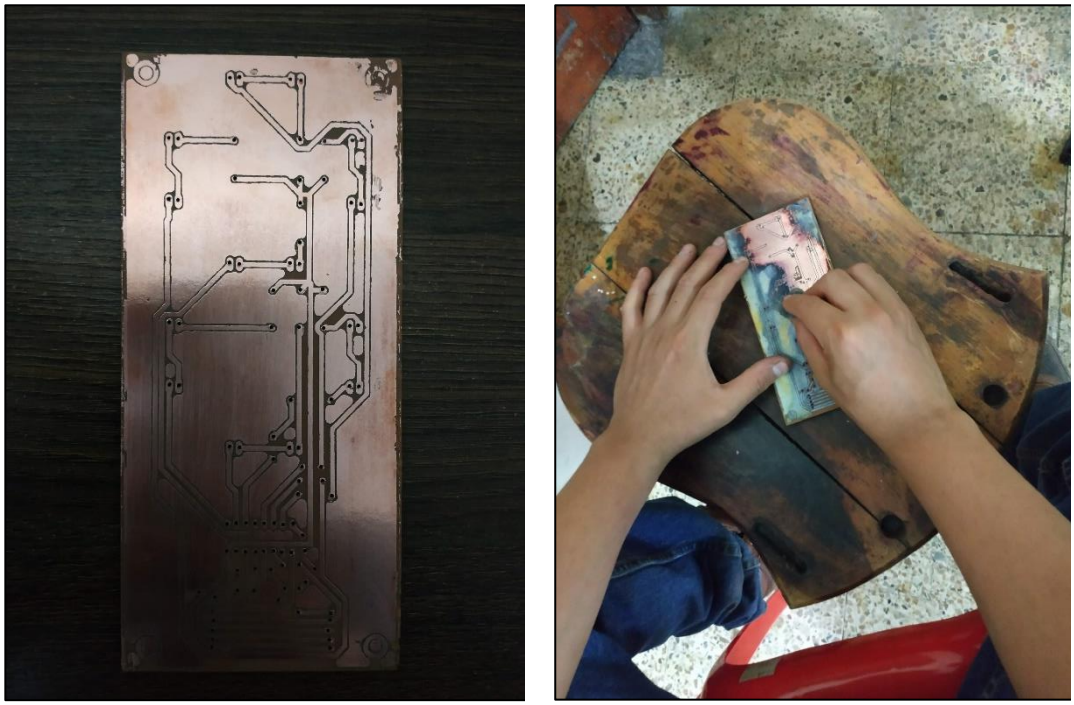


Figura 34- Preparación display 7 segmentos y matrices 8x7 para soldar.

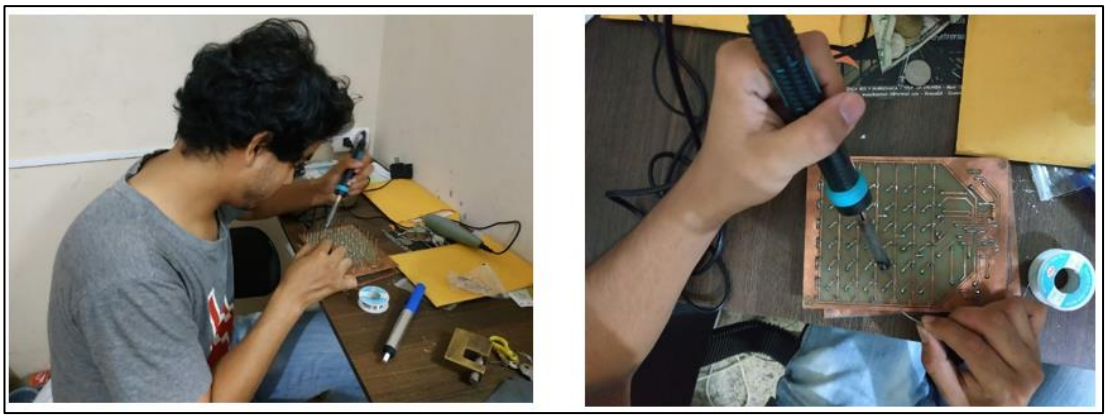


Figura 35- Proceso de soldado vista trasera de placa de Matriz 8x7.

Figura 36- Proceso de soldado vista delantera de placa de Matriz 8x7.

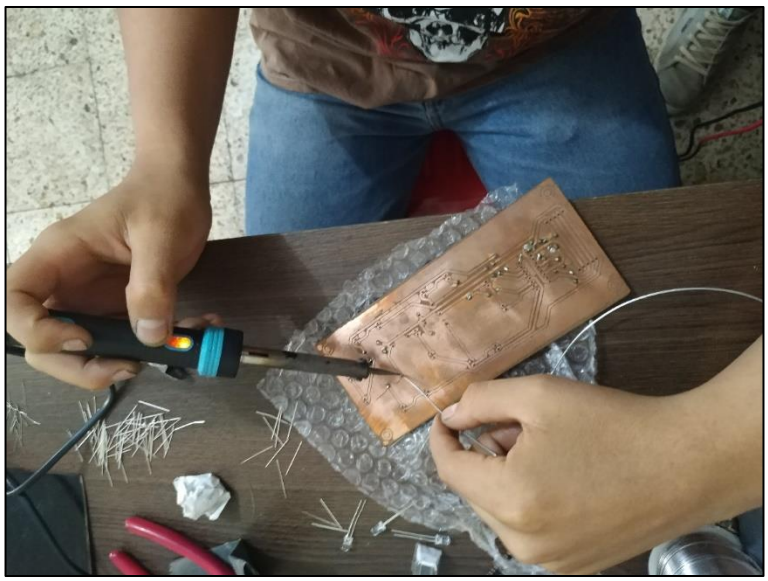


Figura 37- Proceso de soldado, placa de displays 7 segmentos.

El diseño de las placas de matrices de 8x7 está conformado de 56 leds de alta luminosidad de color rojo y verde controlados mediante el integrado **Max7219,** este chip capaz de controlar hasta 8 dígitos de 7 segmentos o 64 diodos LED independientes. El chip incorpora un decoder BCD code-B que realiza un multiplexado y una memoria RAM para almacenar el valor de cada dígito. Únicamente se requiere de una resistencia de 10k ohm externa para hacerlo funcionar.

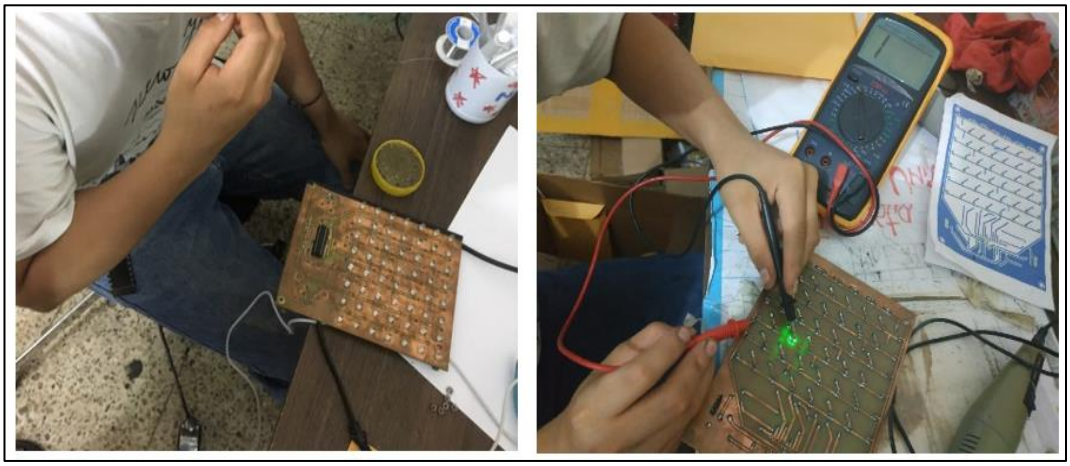


Figura 38- Matriz 8x7 terminadas, vista frontal y delantera.

En las placas de 7 segmentos está constituido de 14 leds de alta luminosidad de color rojos, donde consta por cada segmento dos leds en serie con resistencia de 330ohm, conectados al integrado de registro de desplazamiento 74hc595, con sus respectivos pines de entrada y salida.

Figura 39- Placa display 7 segmentos terminada.

Teniendo listas las placas de diodos leds procedemos agruparlas para que formen las hileras o segmentos que permitirán visualizar la información enviada desde la interfaz web.

Para las hileras que muestran el estilo de natación, nombres de los competidores y carril se agrupó 5 matrices de 8x7 para garantizar la correcta lectura en una hilera, en total tendremos 6 hileras de matrices en el visualizador electrónico y para mostrar el tiempo cronometrado se agrupó 5 placas displays 7 segmentos formando la hilera, que en total serian 5 para los tiempos de cada competidor, y una hilera agrupada de 6 placas displays 7 segmentos para mostrar el estado del agua.

3.2 Diseño y elaboración de la tarjeta electrónica para el microcontrolador arduino nano de las hileras led

Para el diseño y elaboración de la placa que consta del uso una placa arduino nano para control y funcionamiento correcto de las hileras de matrices y display 7segmentos.

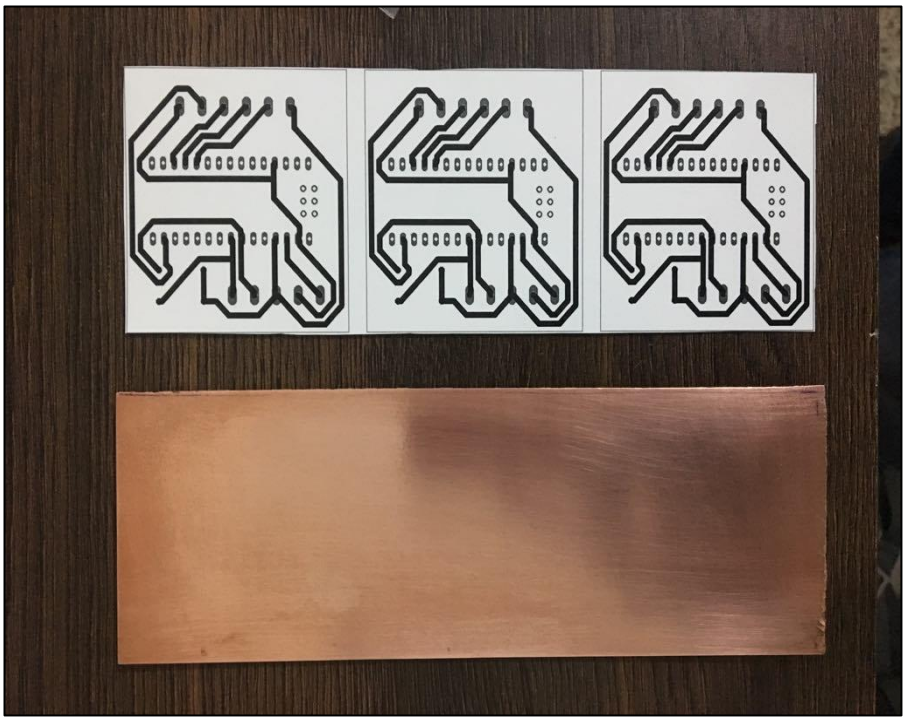


Figura 40- Circuito impreso para el microcontrolador arduino nano para las hileras.

Figura 41- Pulido de placas y ácido actuando sobre la PCB para el microcontrolador arduino nano.

Figura 42- Perforación y corte de placas PCB para el microcontrolador arduino nano

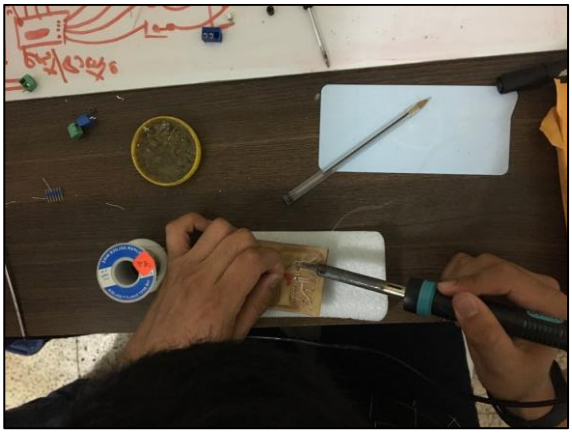


Figura 43- Soldado de componentes electrónicos en la placa para el microcontrolador arduino nano.

La placa terminada para el control de las hileras de las matrices y display 7 segmentos están conformadas con el microcontrolador arduino nano y con sus respectivas conexiones para envió de datos hacia las hileras led, también posee los pines necesarios para lograr una comunicación i2c desde la unidad central (Raspberry) el cual que se encargará del envío de los datos ingresados desde la interfaz para ser mostrados en el visualizador electrónico.

Consta de pines diferentes para las hileras de matrices y display 7 segmentos.

TABLA I

PINES DE CONEXIÓN ENTRE EL MICROCONTROLADOR ARDUINO NANO CON HILERAS DE MATRICES LEDS

Microcontrolador arduino nano	**I.C. MAX7219**
Pin D10	Pin CLK
Pin D9	Pin CS
Pin D8	Pin DIN
Vcc	Vcc
GND	GND

TABLA II

PINES DE CONEXIÓN ENTRE EL MICROCONTROLADOR ARDUINO NANO CON HILERAS DE DISPLAY 7SEGMENTOS

Microcontrolador arduino nano	**I.C. 74HC595**
Pin D4	Pin LATCH
Pin D3	Pin CLK
Pin D2	Pin DATA
Vcc	Vcc
GND	GND

Figura 44- Prueba de funcionamiento de las placas.

3.2.1 Comunicación I2C entre la Raspberry pi3 y arduino nano controlador de las hileras de matrices y 7 segmentos.

En la comunicación que sirve para envío de datos entre la Raspberry (maestro) y con los diferentes arduino nanos (esclavos) que controlarán las diferentes hileras de placas de visualización, se utilizará el protocolo de envío de datos I2C. Este estándar facilita la comunicación de datos entre los diferentes microcontroladores a una velocidad aceptable. En el protocolo de I2C se involucran a continuación las siguientes señales: SCL (System Clock) que cumple con la línea de los pulsos de reloj que sincronizan a todos los dispositivos del sistema, SDA (System Data) es la línea por el cual se mueven los datos entre los dispositivos, el pin SDA del bus requiere de una resistencia pull-up a VCC, y por ultimo GND o tierra debe ser común entre los dispositivos, que debe estar la Raspberry con los arduinos nanos.

Se especifica que el dispositivo maestro será la minicomputadora raspberry pi3, el cual determina la dirección del tráfico de datos, Para usar el I2C en Arduino nanos como esclavos hay que incluir la librería Wire con **#include <Wire.h>.** Donde a cada uno le asignaremos una dirección con el cual podemos comunicarnos individualmente.

3.3 Diseño y elaboración de la estructura del panel de toque subacuático

El diseño y elaboración de la estructura consta de material acero inoxidable con dimensión de 40 x50 cm en la parte frontal de toque y una base que sobresalga con altura de 10cm para la colocación de un cajetín que llevará dentro la comunicación hacia el visualizador electrónico.

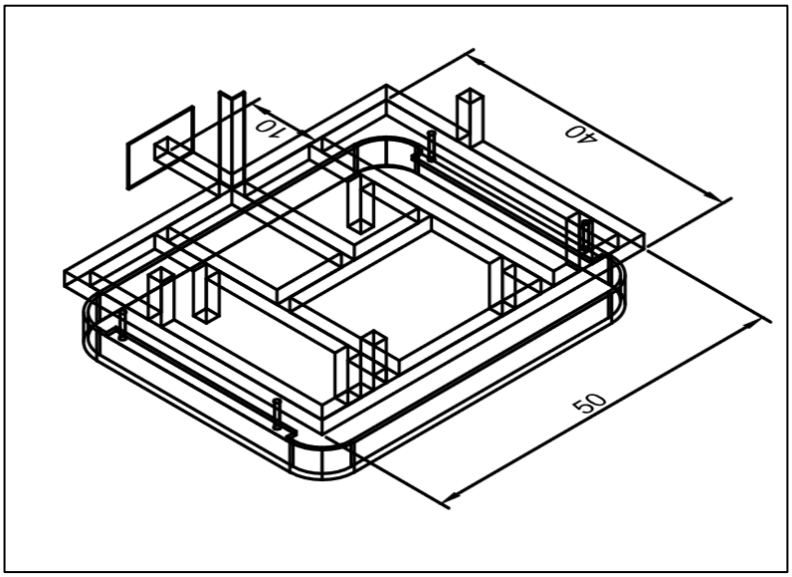


Figura 45- Diseño de estructura de panel de toque en AutoCad.

En la parte del diseño de sensores en la estructura interna del panel se colocaron 5 piezoeléctricos que están conectados independientemente, cubiertos de material impermeable para evitar cualquier inconveniente con el circuito, con el fin que permita generar las lecturas en señales analógica hacia el cajetín de procesamiento y comunicación.

Figura 46- Piezoeléctricos.

Figura 47- Soldado y barnizado de piezoeléctricos

Figura 48- Colocación y sellado de piezoeléctricos en estructura.

El cajetín de comunicación consta con el diseño y elaboración de una tarjeta que hace la función del módulo con sus respectivos elementos electrónicos y energizada mediante una pila recargable, para montar el Wemos Mini D1 que nos servirá para la comunicación del envió de datos mediante la tecnología wifi a la unidad central de proceso y de ahí al visualizador electrónico.

Figura 49- Diseño y elaboración de placa para comunicación wemos mini D1.

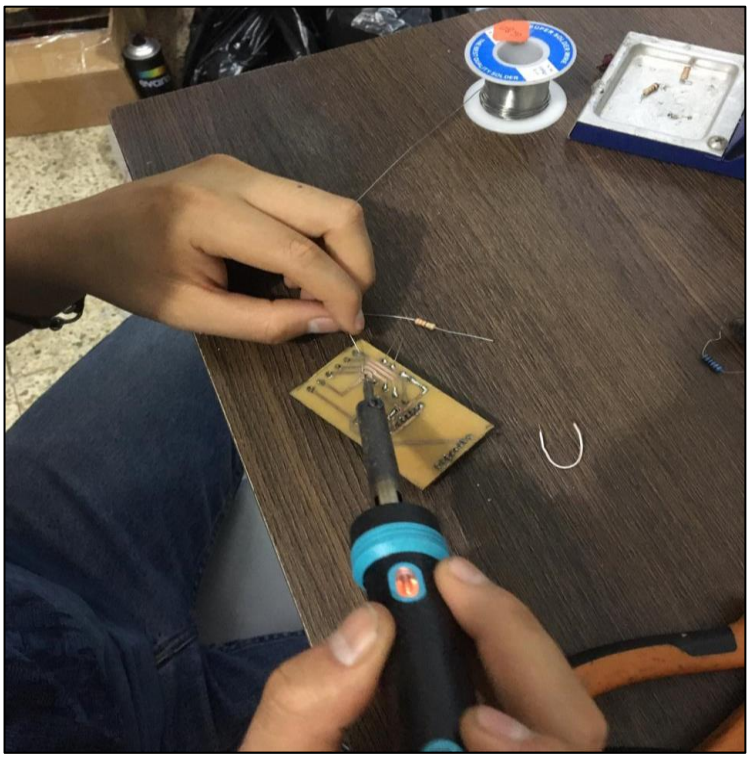


Figura 50- Soldado de placa de comunicación wemos mini D1 y sus componentes.

La realización y montaje del módulo de comunicación wemos mini D1 en su respectiva placa diseñada y elaborada, consta de 6 borneras laterales en que 5 son de la entrada de señal analógica de los piezoeléctricos y uno de GND común, con resistencias de 330 ohm por cada piezoeléctrico el cual contamos con 5 de ellos, todos están conectados en paralelo con el fin de evitar que si uno de los piezoeléctricos no llegase a ser tocado los otros puedan mitigar la perdida de toque logrando que la señal de voltaje analógico de entrada llegue al Pin A0 y a su vez el

común en GND del módulo wemos mini D1, cual tendremos asegurado que la sensibilidad del toque sea precisó y pueda logar que genere la señal para que el módulo wemos mini D1 trasmita mediante la red interna wifi a la que esta enlazada a la unidad central (raspberry pi3) para lograr detener el tiempo cronometrado de los competidores y ser visualizado.

Además, la placa cuenta con una resistencia de 1Mohms y un diodo zener 1n4738a en paralelo en la entrada del Pin A0 con el fin de utilizarlo como regulador de tensión. También contamos con una bornera adicional para la alimentación del módulo de la tarjeta wemos mini D1, el cual será energizado mediante una batería de 3.3V recargable, a su vez con un interruptor para encendido y apagado del mismo.

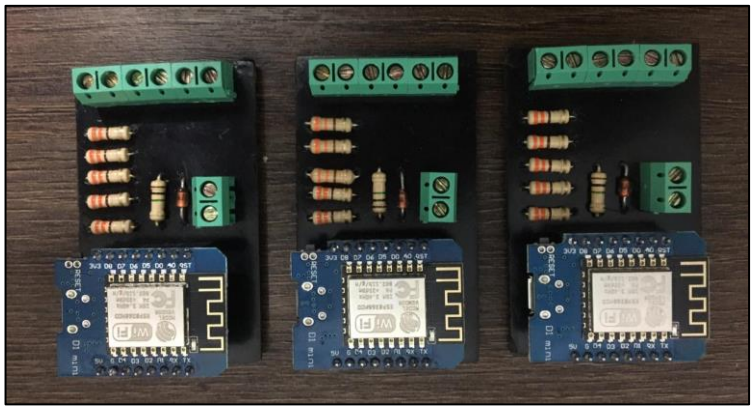


Figura 51- Montaje de placa wemos mini D1.

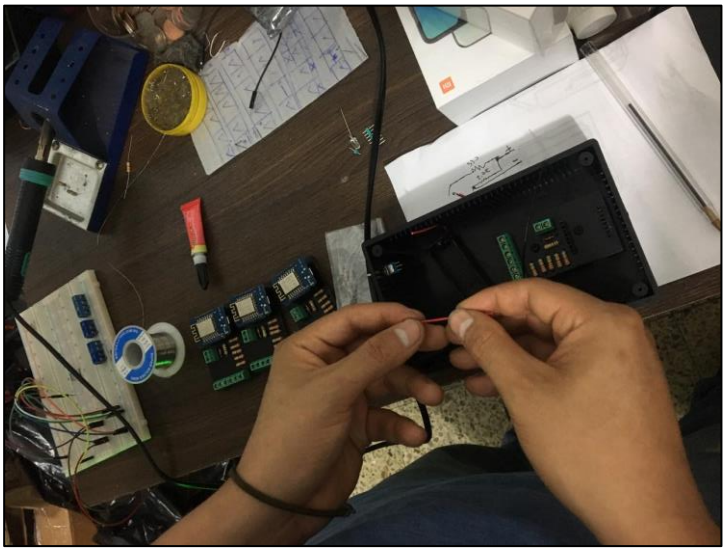


Figura 52- Armado del módulo cajetín de wemos mini D1.

Figura 53- Armado completo de la estructura del sensor toque.

3.4 Sensores del estado del agua

3.4.1 Diseño y elaboración de la tarjeta electrónica para los sensores de estado del agua

El diseño de la placa electrónica consta de módulos externos para la conexión de los sensores de pH, temperatura y turbidez, los cuales estarán directamente conectados a un Arduino nano y el mismo a una placa Wemos que permitirá realizar la comunicación inalámbrica mediante wifi para el envío de datos hacia la raspberry pi 3 para que este luego procese estos datos y pueda enviarlos al visualizador electrónico.

TABLA III

PINES A USAR PARA LA COMUNICACIÓN SERIAL ENTRE EL MICROCONTROLADOR ARDUINO NANO Y WEMOS D1

Arduino nano	**wemos D1**
Gnd	Gnd
Vcc	Conexión a 5 VDC
Rx(D8)	Rx(D4)
Tx(D7)	Tx(D5)

Figura 54- Placa de conexión entre sensores del estado físico químico del agua arduino nano-wemos D1

3.4.2 Implementación de los módulos externos del sensor del agua

3.4.3 Sensor de pH analógico

El sensor de pH está conformado por una sonda que es la que recoge la lectura, y mediante el conector BCN se conecta hacia un módulo o tarjeta de procesamiento de datos, que a su vez este envía los datos obtenido de la lectura del estado del pH del agua hacia la entrada analógica del microcontrolador arduino nano.

En cuanto a el pH, es una medida de acidez o alcalinidad de un líquido. Y por lo general, el rango pH se encuentra entre 0 y 14. Por lo consiguiente el pH = 7, significa que la solución es neutra; un pH <7, indica que la solución es ácida; pH> 7, la solución es Alcalina. En el caso particular del proyecto de titulación el rango del pH en la piscina siempre se debe ubicar entre 7.2 y 7.6.

TABLA IV

PINES A USAR EN EL SENSOR DE PH ANALÓGICO HACIA LA PLACA MICROCONTROLADOR ARDUINO NANO:

Sensor de Ph analógico	**Arduino nano**
Po	Pin A0 Valor de pH en voltaje.
GND	GND
V+	Conexión a 5 VDC

3.4.4 Sensor digital de Temperatura impermeable ds18b20

El sensor de temperatura ds18b20 es un dispositivo electrónico preciso, tan solo necesita una resistencia de 4.7 kohm entre los pines data in y vcc para poder funcionar, por lo tanto, es ideal para obtener datos físicos en soluciones y en este caso en particular del agua de la piscina, este sensor es una versión que viene incorporada con la característica de ser a prueba de agua lo cual lo hace totalmente sumergible ideal para la adquisición de datos en una piscina.

Para la utilización e implementación de este sensor en el proyecto se requiere la instalación de dos librerías DallasTemperature.h y oneWire, para poder escribir comandos simples para obtener lecturas de temperatura del sensor y la utilización de un solo pin digital en la comunicación con la placa del microcontrolador arduino nano respectivamente.

TABLA V

PINES A USAR EN EL SENSOR DIGITAL DE TEMPERATURA HACIA LA PLACA MICROCONTROLADOR ARDUINO NANO

Sensor digital de Temperatura Ds18b20	**Arduino nano**
Gnd	Gnd
Data	Pin(D2) digital con una resistencia pull-up de 4.7k ohm
Vcc	Conexión a 5 VDC

3.4.5 Sensor Analógico TDS

Este sensor nos permitirá ver la calidad del agua mediante TDS (Total de sólidos disueltos) que indica cuántos miligramos de sólidos solubles se disuelven en un litro de agua. En general, cuanto más alto sea el valor de TDS, más sólidos solubles se disuelven en agua y por lo consiguiente menos limpia será el agua, produciendo que el agua se vea más turbia. Por lo tanto, el valor TDS puede ser utilizado como un marco de referencias para reflejar la limpieza o turbidez del agua.

El sensor analógico TDS es compatible con la plataforma de arduino y por lo tanto con arduino nano. Este sensor cuenta con un módulo o placa adaptador de señal y sonda impermeable que se puede sumergir en agua durante largos períodos de tiempo, lo que hace ideal para implementar en este proyecto en particular.

Es importante mantener el valor de TDS en 280ppm mediante esto podemos medir la calidad de agua y evitar que se ponga turbia, el rango aceptable se debe ubicar entre 200 y 300 ppm para a tener una calidad de agua optima y no turbia. Si mantenemos la alcalinidad óptima el nivel de turbidez del agua también será ideal.

TABLA VI

PINES A USAR EN EL SENSOR ANALÓGICO TDS HACIA LA PLACA MICROCONTROLADOR ARDUINO NANO

Sensor Analógico TDS	**Arduino nano**
GND	GND
Data	Pin(A1)
Vcc	Conexión a 5 VDC

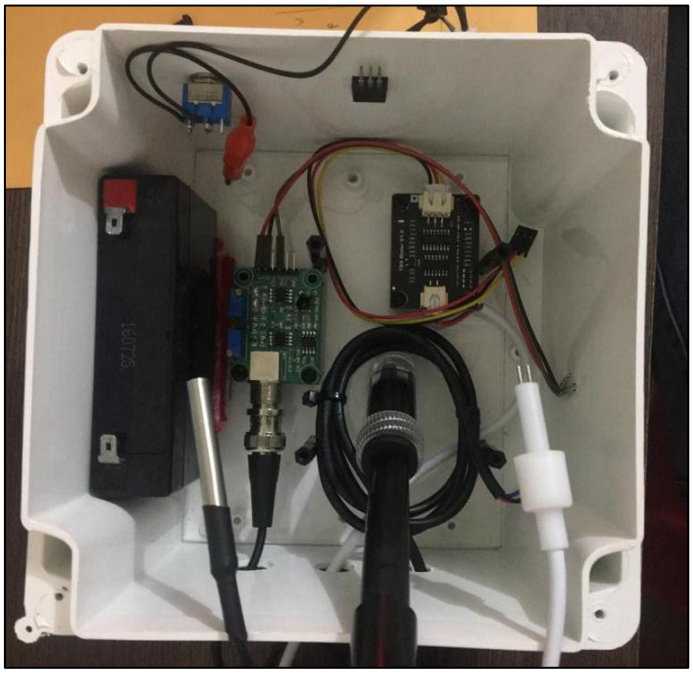


Figura 55-Montaje de los módulos externos de los sensores del estado del agua.

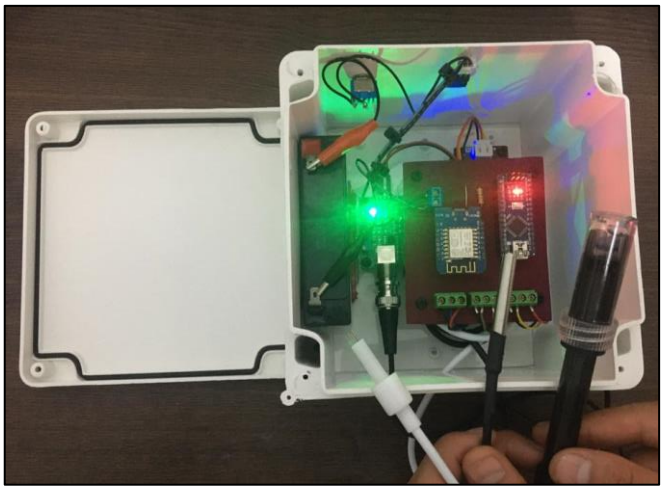


Figura 56- Implementación del módulo arduino nano-wemos para el estado del agua.

3.5 Diseño y elaboración de placa electrónica para el sonido de inicio de competencia, reset y quite de energía para el visualizador electrónico

Se usó un microcontrolador arduino nano para el diseño del circuito del sistema de sirena indicadora de inicio de competencia de natación, contando con una sirena que trabaja a 12v DC y que tendrá un dispositivo electrónico relé que está conectado en el pin de normalmente abierto (N.O), solo cuando se genere la petición de inicio pulsando el botón que indica “Iniciar competencia “dentro de la interfaz web alojada en el servidor, el dato que indica la acción se envía mediante la comunicación i2c hacia un microcontrolador arduino nano (esclavo) ubicado físicamente dentro del módulo central. Se enviará el dato al relé e instantáneamente cambia de modo normalmente abierto (N.O), a normalmente cerrado (N.C), por un tiempo determinado de 2 segundo y luego pasado ese tiempo automáticamente regresa a su estado de inicio.

Se pensó para limpiar los datos del visualizador electrónico usar el pin de reset en común en todos los microcontroladores arduino nanos , y el uso del quite de energía con el fin mitigar un tipo de pérdidas de datos en las hileras de matrices y display de 7 segmentos, gracias al uso de otro dispositivo electrónico relé que están todos los microcontroladores arduino nanos del visualizador electrónico energizados atreves del paso del pin del relé en normalmente cerrado (N.C), menos el microcontrolador arduino nano que esta energizado independiente para lograr enviar el dato de N.C a N.O por pocos segundos, donde regresará su estado de inicio.

La placa cuenta también con la comunicación hacia la unidad central raspberry pi 3(maestro) para conectarse los pines de comunicación de i2c entre lo microcontroladores arduino nanos (esclavos) del visualizador electrónico.

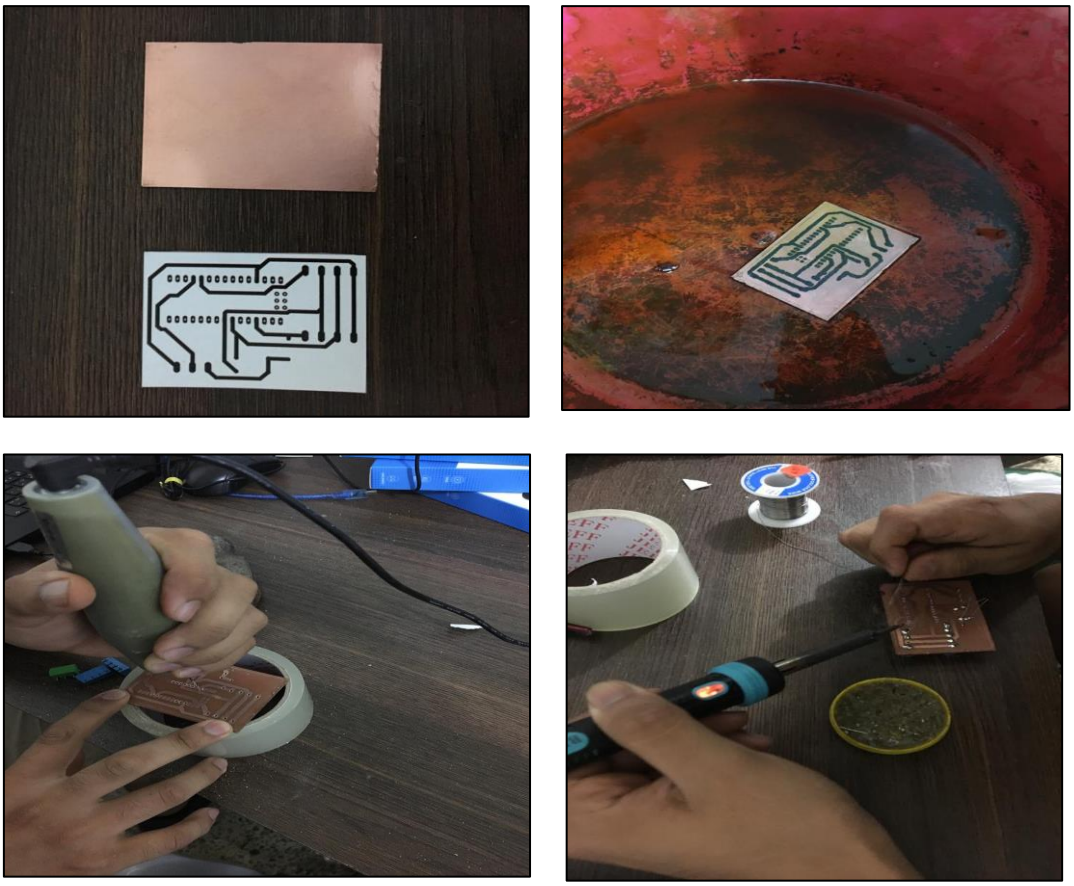


Figura 57- Proceso del desarrollo placa electrónica

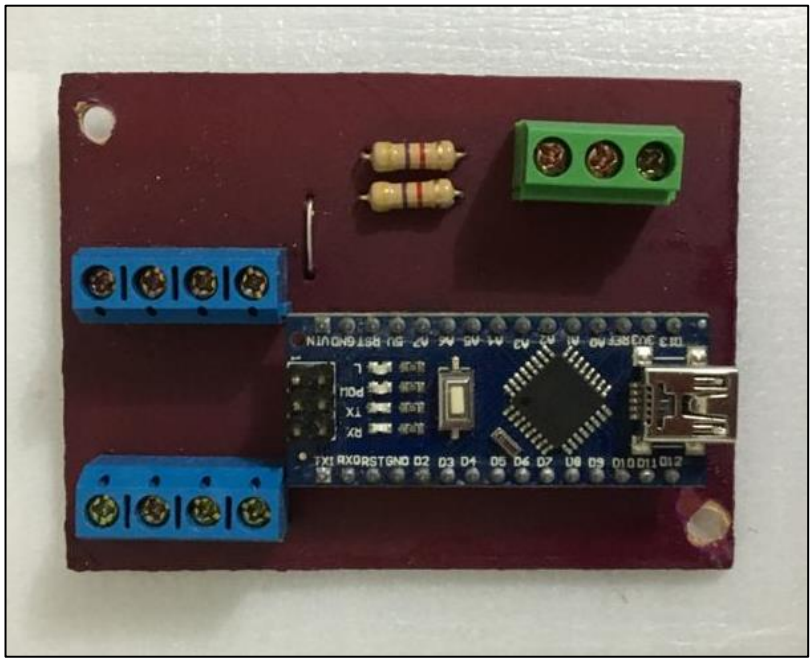


Figura 58- Placa terminada sirena indicadora, reset y quite de energía.

TABLA VII

PINES A USAR EN LA PLACA ELECTRÓNICA PARA EL SONIDO DE INICIO DE COMPETENCIA, RESET Y QUITE DE ENERGÍA PARA EL VISUALIZADOR ELECTRÓNICO

Microcontolador Arduino nano	**RELE # 1**	**Microcontroladores Arduino nanos(Visualizador electronico)**	**RELE # 2**
GND	GND	GND	GND
Vcc	Vcc	X	Vcc
Pin(X)	Pin(D9)	X	X
Pin(X)	X	X	Pin(D10)
RST	X	RST	X

3.6 Configuración red LAN interna

A la red lan interna del proyecto de titulación se la configuro de tal manera que tengan la dirección ip 192.168.20.0. dentro de la red se encontrará el servidor http que obligatoriamente se sebe configurar con una dirección ip estática asignada por el servidor dhcp del router conociendo la mac del enlace de conexión a la red, esto para que la ip no cambie cada vez que se prenda el computador que brinda el servicio. para conocer la mac por cada forma de conexión a la red se debe escribir el siguiente comando en la terminal del computador raspberry.

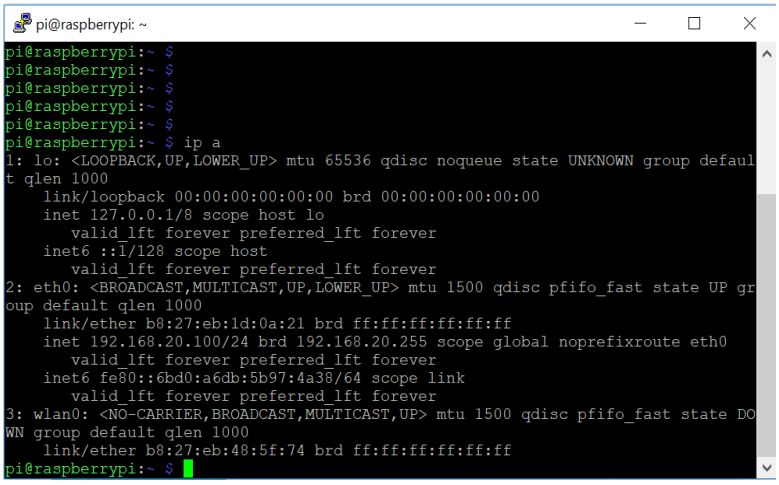


Figura 59- Direcciones mac para cada enlace a la red del raspberry pi 3.

Una vez que se sabe las direcciones Mac de los enlaces físico o inalámbrico se debe entrar a la configuración interna del router para luego dar click en la opción dhcp, todo es con el fin de reservar y habilitar una dirección ip estática

asignada por el servidor dhcp, lo antes mencionado se logra mediante la Mac del enlace físico por el cual se accede a la red.

Figura 60- Ingreso a la configuración interna del router.

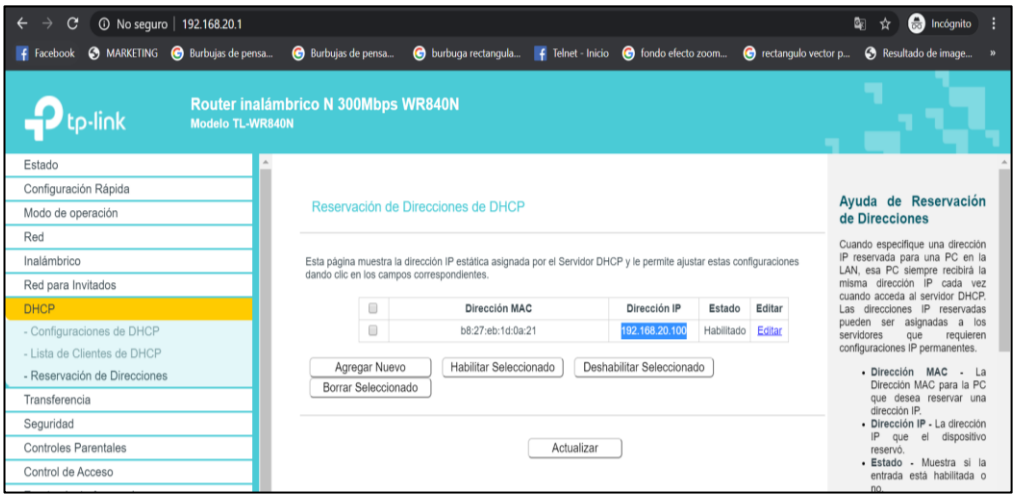


Figura 61- Reservación de dirección ip estática para la raspberry pi 3

TABLA VIII

DIRECCIONAMIENTO IP (RED IP:192.168.20.0)

DISPOSITIVO	**DIRECCION IPV4**	**ESTADO**
ROUTER	192.168.20.1	HABILITADO
SERVIDOR HTTP	192.168.20.100	HABILITADO

3.7 Diseño y programación de la interfaz web (Front-End) para la administración de competencias de natación.

En el control, administración y supervisión de competencias deportivas de natación será de forma remota y para se pueda cumplir lo mencionado anteriormente se necesita la utilización de una interfaz web que realice dichas operaciones un dispositivo celular o computador deberá acceder inalámbrica a una LAN interna privada donde se encontrará el servidor de Python (Raspberry) en este servidor se encuentra alojada nuestra interfaz web que será dinámica y fácil de usar e interpretar garantizando la interacción con la persona encargada “Juez”.

El desarrollo de la interfaz web para la administración del visualizador electrónico fue bajo una herramienta de plantillas o framework para desarrolladores web llamada vue.js ya que es fácil de leer, comprender y aplicar a nuevos proyectos además que es de código abierto.

Para poder acceder al servidor y directamente a nuestra página web será de una manera muy sencilla y fácil de comprender para cualquier persona como se mencionó previamente, aunque no posea conocimientos básicos de informática, todo el proceso se simplifica en simples pasos a seguir donde el primer paso será abrir cualquier explorador conocido como Chrome y escribir la dirección ipv4 asignada al servidor estáticamente por el router tp-link y será **192.168.20.100** además seguidamente (:5000) que indica el puerto disponible para acceder al servicio. Y por último se da un ENTER para enviar la solicitud de petición. Recibida la respuesta el servidor envía la página principal para poder iniciar una competencia.

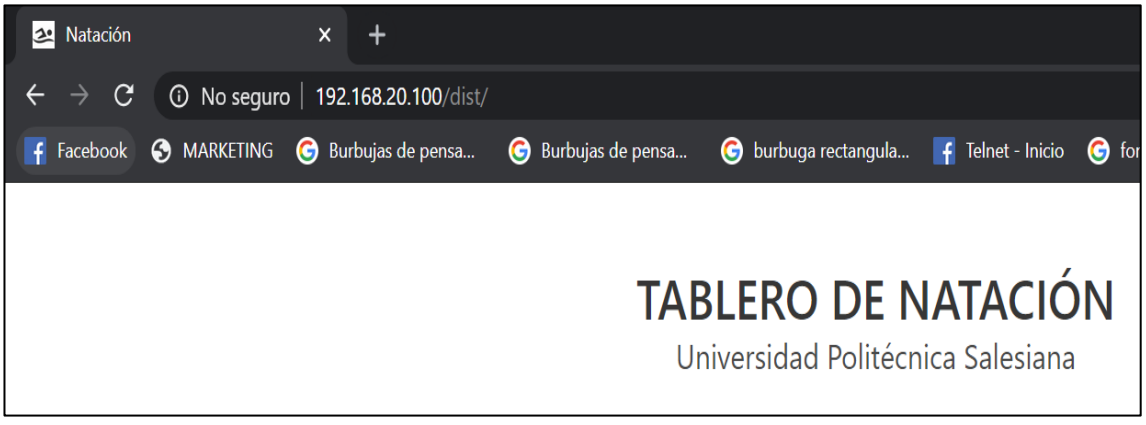


Figura 62- Dirección IPv4 estática para acceder al servidor http donde se encuentra alojada nuestra interfaz web.

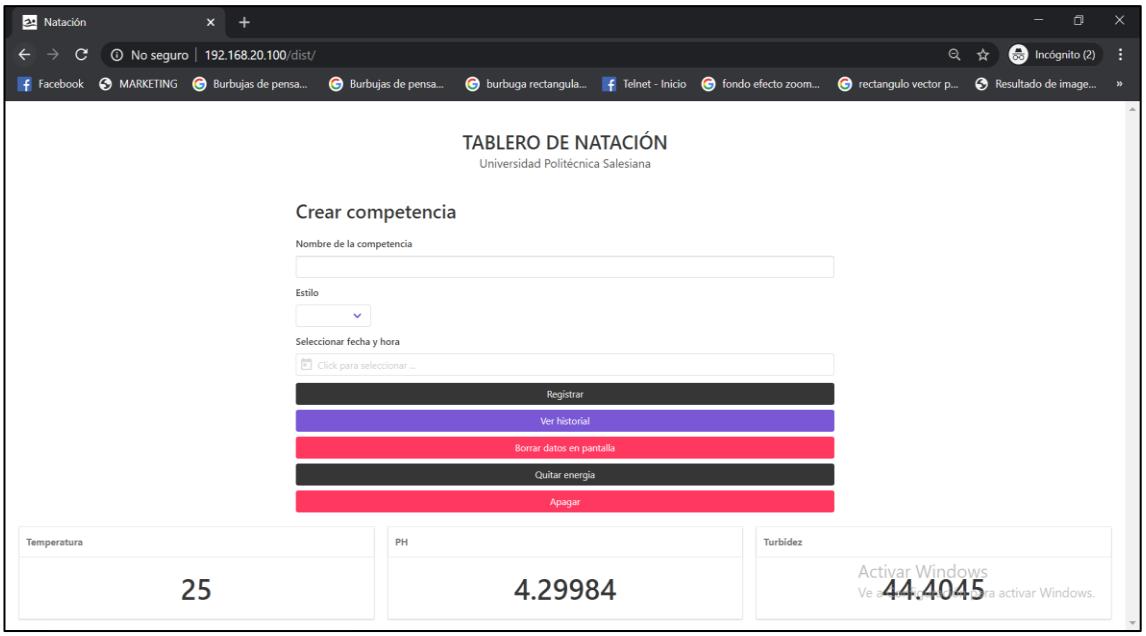


Figura 63- Vista de la página principal accedida desde un computador.

Figura 64-Vista de la página principal accedida desde un smartphone.

3.7.1 Configuración de la interfaz web como acceso directo

Esta operación facilita y agilita el acceso a nuestra plataforma para cronometrar competencias, para realizar la siguiente operación de ubicar un acceso directo desde cualquier dispositivo electrónico, con esto cabe recalcar que ya no es necesario escribir una dirección IP con su puerto de acceso o sus diferentes rutas que es un poco trabajoso y complicado.

Se muestra a continuación los pasos para crear un acceso directo a nuestra interfaz web en la pantalla de inicio o “escritorio” de un celular, computador respectivamente.

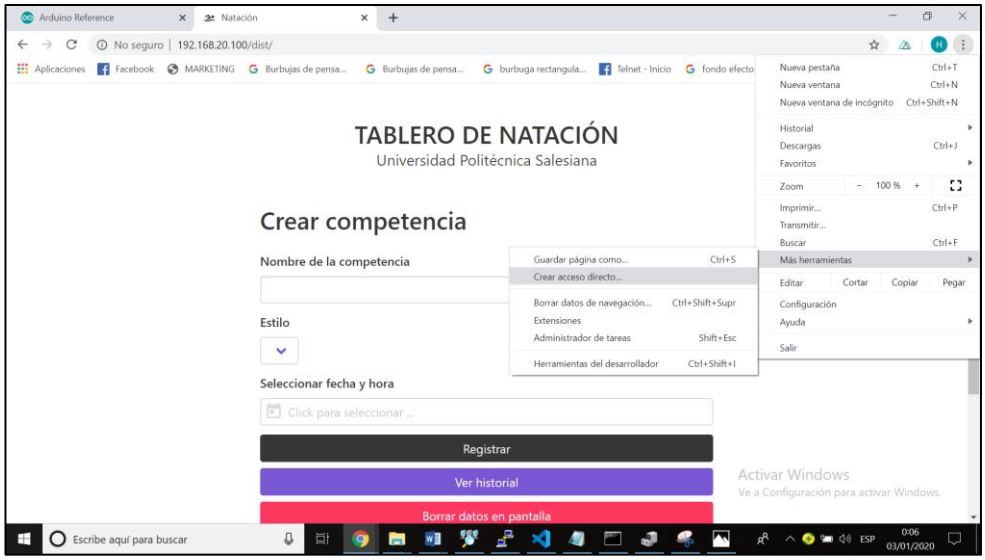


Figura 65- Pasos demostrativos para crear un acceso directo.

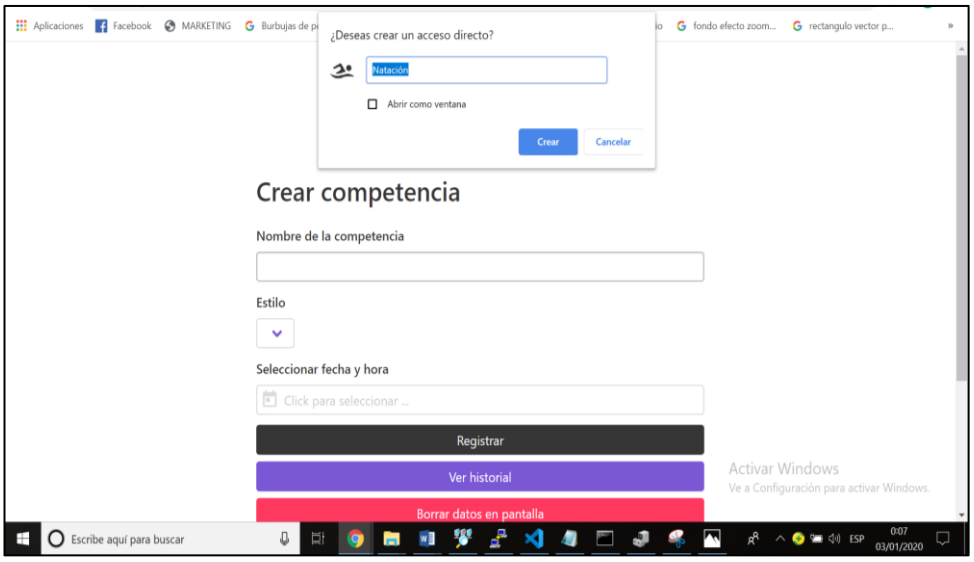


Figura 66- Asignar un nombre para el acceso directo.

Figura 67- Icono de acceso directo vista desde un computador.

3.7.2 Python3 y flask programación web (Back-end)

El servidor HTTP que brindará el servicio alojar y acceder a la interfaz web será python y con su framework FLASK, que permite reutilizar código y ayuda que esta app web sea escalable, pero para poder utilizar todos los beneficios de este framework y todas sus librerías es necesario primero proceder a descargar desde la terminal del dispositivo con el siguiente comando install flask, una vez instalado flask podrá utilizar los principales e importantes métodos http tales como son POST, GET, PUT, DELETE, para poder desarrollar la interfaz web y crear rutas para poder interactuar cliente-servidor como se muestra en la figura a continuación.

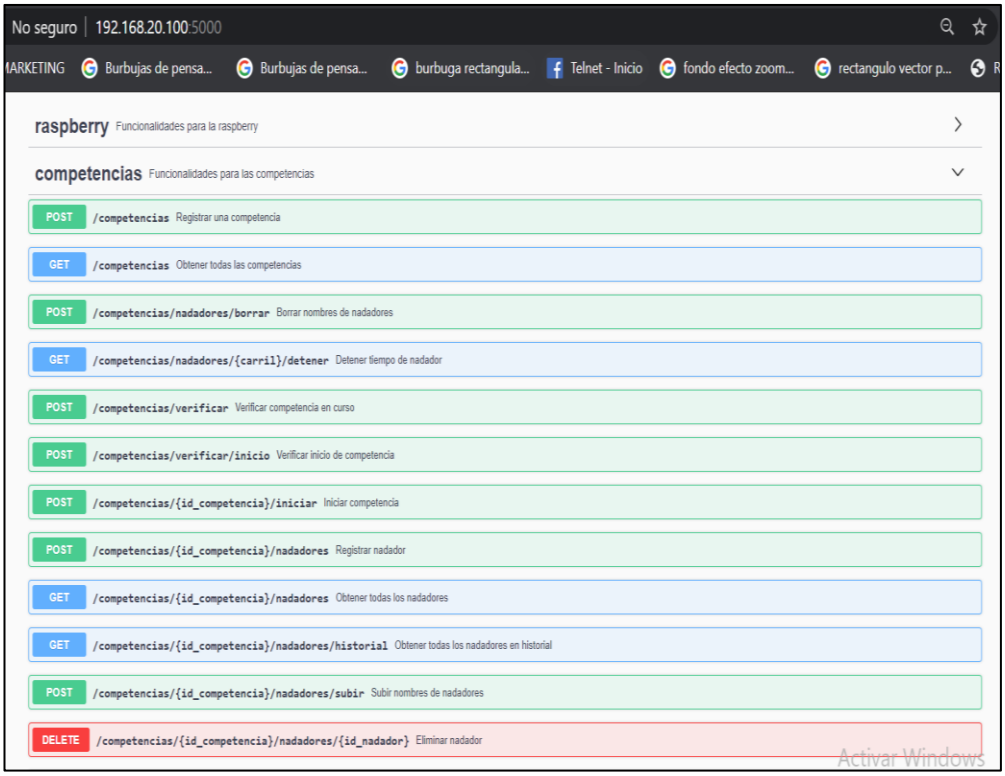


Figura 68- Principales métodos HTTP para la programación del lado del servidor.

Con el framework Flask escrito en ptython y compatible con la versión de python3 utilizada en el proyecto, permite la creación de diferentes rutas para acceder a los diferentes ambientes con sus principales acciones que realizan para la administración de una competencia de natación y que se procede a mencionar a continuación.

La primera ruta que se creó con Python-flask (**http://192.168.20.100/**) muestra una interfaz que es destinada para la creación de competencias, donde claramente se compone de tres partes, la primera para llenar campos obligatorios para iniciar alguna competencia tales como nombre de la competencia (método post), estilo de competencia (método get) ya que toma datos de la base de datos y la opción de seleccionar fecha y hora (método post) que realiza la opción de asignar valores a la base de datos. En la segunda sección se compone de botoneras utilizadas para registrar (botón) la competencia a realizarse con sus características (método post), Ver historial (método get) realizando una petición desde la base de datos los cuales se mostrarán en una página web en otra ruta diferente la cual se explicará más adelante detalladamente, el botón borrar datos en pantalla limpia todos los datos previamente mostrados en el visualizador utilizando (método post), el botón que indica quitar energía realiza un paro emergente en caso de emergencia para cuidar componentes electrónicos dentro del sistema, y por último el botón sirve para apagar correctamente nuestra Raspberry utilizando el método post.

Y por último la parte inferior que constantemente mostrará valores obtenidos desde la tarjeta de sensores de agua en la piscina utilizando el método GET.

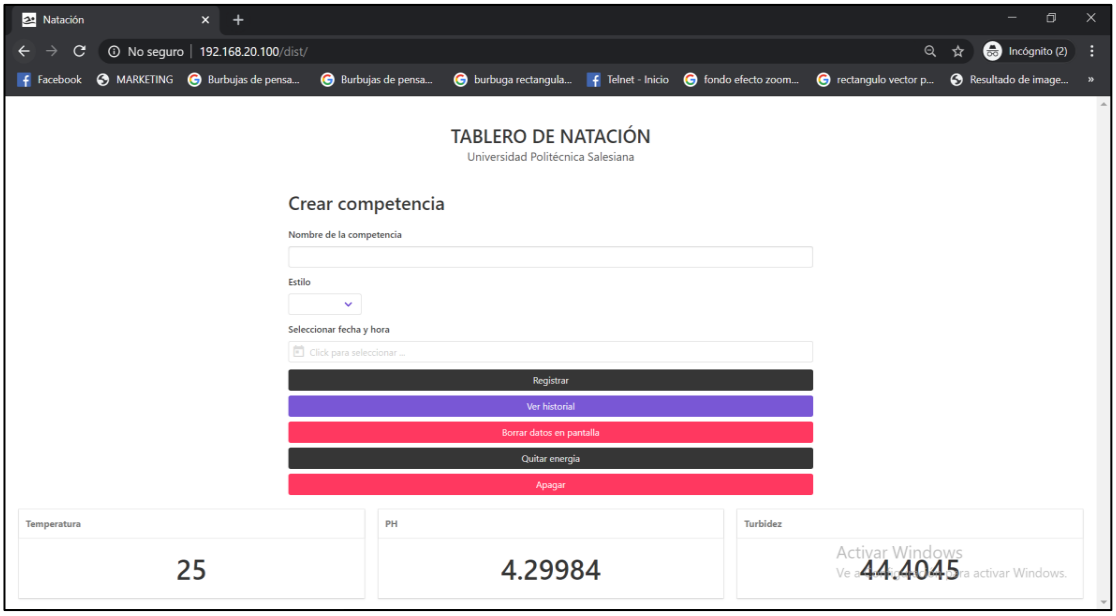


Figura 69- Página principal de la interfaz web vista desde un computador.

Para la siguiente ruta (http://192.168.20.100/nadadores) se podrá acceder una vez completada la ruta anteriormente descrita, y es donde una vez creada la competencia y estilo se podrá registrar los competidores, se ingresan campos necesarios mediante la utilización de una herramienta llamada formulario para nombre y apellido en la programación en el lado del servidor toma aquellos parámetros y los asigna a una tabla en la base de datos mediante el método POST, la selección de edades, semestre, carrera universitaria y carril utilizan métodos GET obteniéndolos desde la base de datos, la segunda parte es un único botón que utiliza el método POST para ahora si permitir agregar los datos características de cada uno de los competidores, y la última parte se compone de los datos sensados anteriormente mencionados. Cabe recalcar que no se podrá avanzar a una nueva ruta con flask hasta que no se hayan llenado todos los campos obligatorios por llenar en esta ruta.

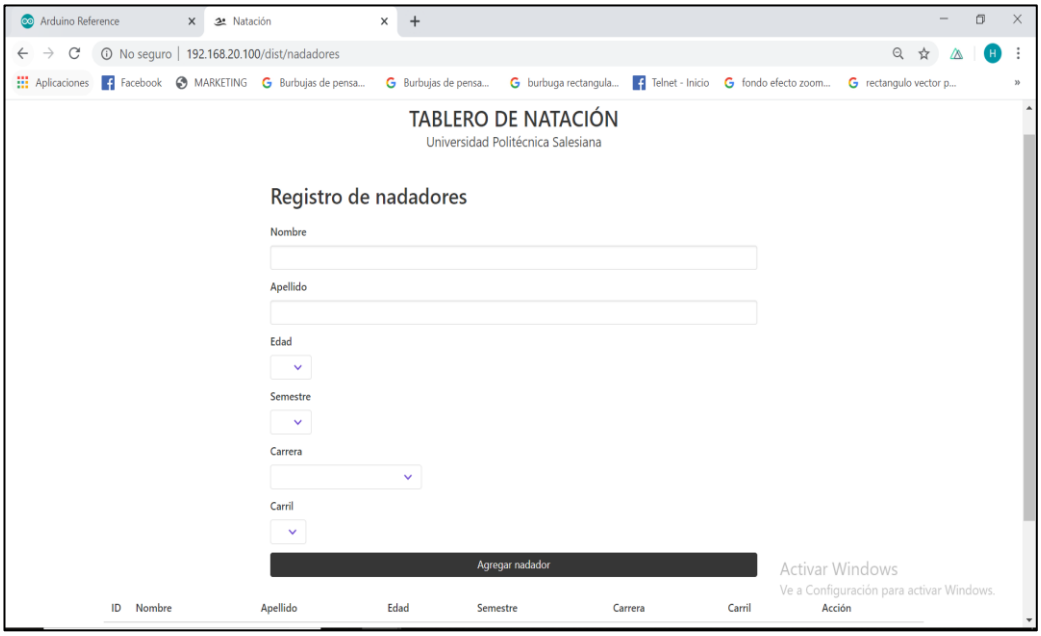


Figura 70- Página destinada para el registro de nadadores vista desde un pc.

Una vez ingresados todos los datos de los competidores se pueden eliminar cual se desee con utilizar un método en Python-flask llamado DELETE que sirve obviamente para eliminar datos. Como se muestra en la siguiente figura. Además, se agregaron funciones con botoneras entrelazadas a la programación del servidor utilizadas para diferentes fines, Iniciar carrera que se lo configura de tal manera que realice una operación de redirigir hacia a una siguiente ruta (competencia en curso) de nuestra interfaz además utiliza el método POST, el botón con función de enviar datos en pantalla utilizando el método POST, borrar datos en pantalla POST y quitar energía (paro emergente) utilizando el método POST.

Figura 71- Datos de competidores preparados para dar inicio de carrera.

3.8 Base de datos

Es la base de datos donde almacenaremos todo el contenido de la información de las competencias de natación como sus competidores el semestre actual de estudio, la carrera de estudio, edades y unos de los parámetros más importantes que es el carril, además todos estos datos se guardarán en orden cronológico con respecto al tiempo, y finalmente todos esos datos podrán ser accedidos posteriormente con fecha y hora y visualizados mediante una página web con una ruta especifica si fuera necesario.

Se planteó para que la base de datos se un respaldo y sea una fuente de la verdad de los datos ya almacenados.

La estructura de todos los datos alojados en la base de datos se describe a continuación mediante la siguiente figura.

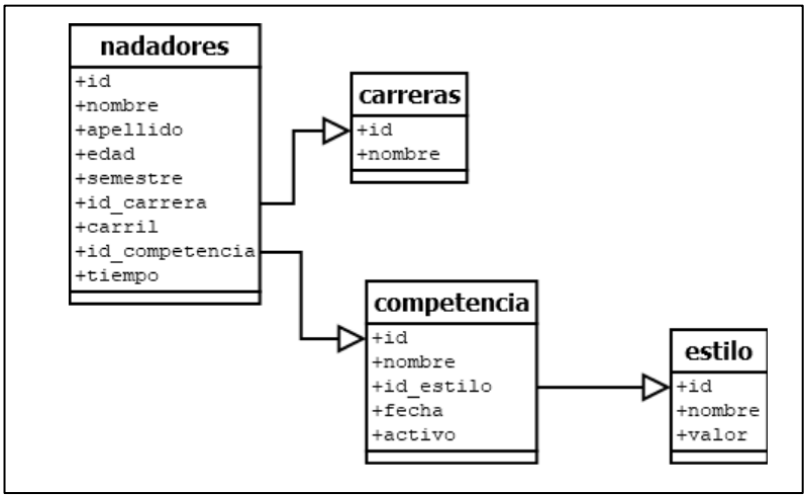


Figura 72- Esquema estructural de la base de datos del proyecto.

3.8.1 Visualización de la base de datos con phpMyAmin para administración

El utilizar una herramienta de interfaz gráfica de código libre para la visualización facilita el uso, lectura y la administración de los datos ingresado en una base de datos es decir evita tener que escribir código SQL y los genera de forma automática en la base de datos de manera fácil con simples (clicks), además permite posteriormente ingresar a generar consultas en ella y poder utilizar los datos de acuerdo a los requerimientos de las competencias. Esta interfaz web se alojará en un servidor HTTP con servicio Apache, pero cabe recalcar que al ser código libre implementar esta herramienta al proyecto no genera costos adicionales.

Mediante esta interfaz gráfica phpMyAdmin se crearon todas las tablas necesarias en el proyecto de titulación tales como la tabla de nombre de competencia, estilo, fecha y hora, nombre de competidores, edad, semestre y carrera de estudio, carril y por último los datos del estado físico químico del agua.

Obviamente para poder acceder a los datos, poder leerlos, escribirlos o reescribirlos se necesita de una autenticación con su USUARIO y LOGIN con el fin de proteger de un intento de ingreso de un usuario no deseado, para acceder a la interfaz alojada en apache se procede a escribir la siguiente url: http://192.168.20.100/phpmyadmin desde cualquier explorador.

En el proceso de instalación de la interfaz gráfica es sencillo con tan solo ejecutar diferentes comandos desde la terminal de consola del sistema operativo Debian utilizado en la placa Raspberry, el primer paso será abrir la terminal, escribir el comando sudo apt install phpmyadmin, como se muestra en la siguiente figura.

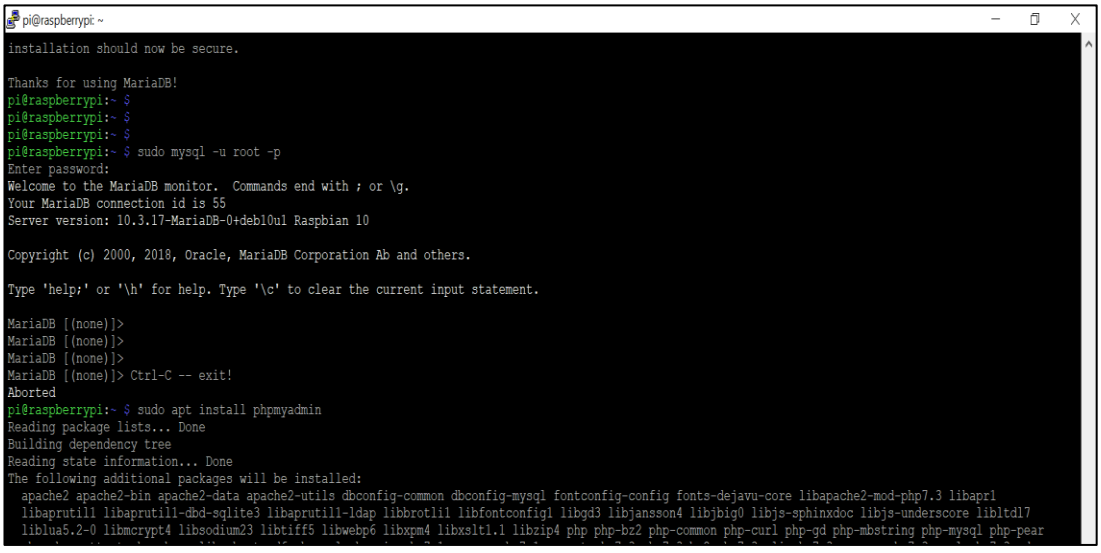


Figura 73- Instalación de phpadmin en raspberry Pi3.

3.9 Principales funciones de la interfaz web para cronometrar competencias

3.9.1 Página de la aplicación web para registro de nuevas competencias

Para acceder a la página principal del aplicativo web se debe de digitar la siguiente ruta url: http://192.168.20.100/ y será en donde se podrá crear una nueva competencia y se mencionará a continuación como llenar los parámetros necesarios para iniciar la competencia, a continuación se muestra como es el aspecto de la página principal vista desde un computador.

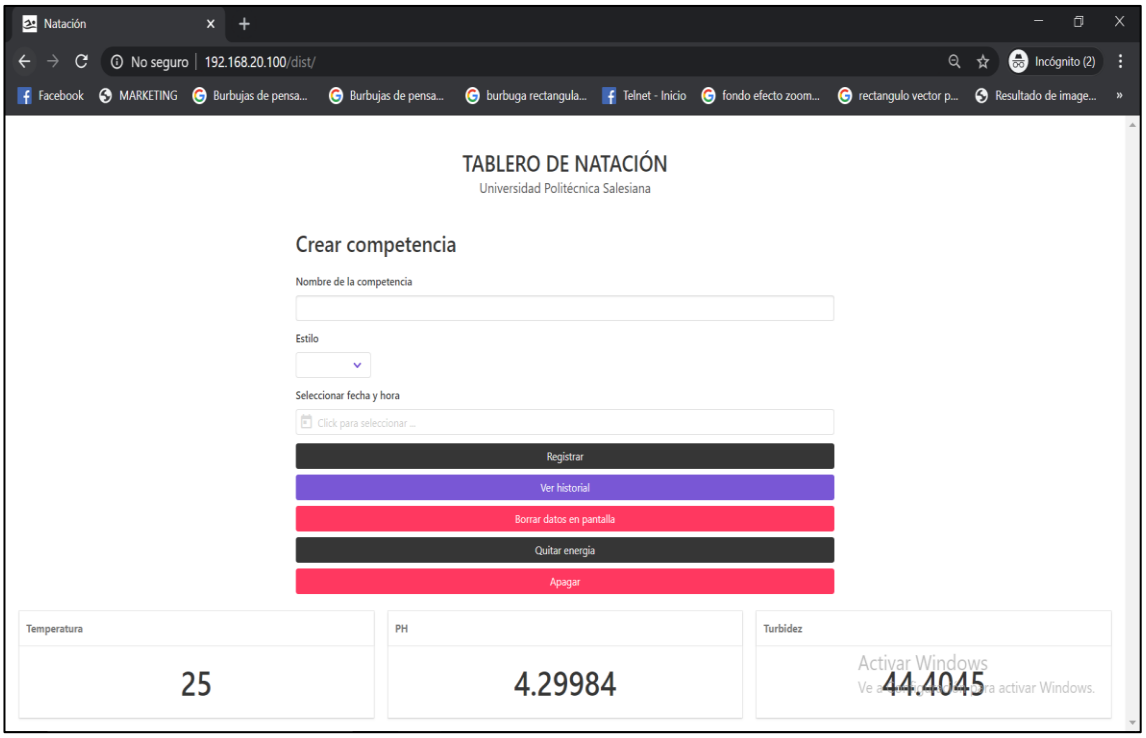


Figura 74- Ventana principal.

3.9.2 Ingreso del nombre de la competencia

El primer dato que debe ingresar una vez dentro de la ruta de la interfaz es el nombre de la competencia a realizarse es un dato obligatorio e importante y en donde se recomienda poner nombres diferentes a cada competencia debido a que este dato es un identificador único para la competencia al momento de requerir toda la información de competencias ya acontecidas.

3.9.3 Selección de estilo de competencia

Una vez ingresado el nombre de competencia se debe de elegir un estilo para la natación competitiva entre una lista de cuatro opciones que posee la base de datos como se muestra en la siguiente figura, una vez seleccionado el estilo definirá la forma de nado que los participantes competirán y que hoy en día estos cuatro estilos se encuentran reconocidos por el comité olímpico internacional de natación. Esto es obligatorio e indispensable para continuar al registro de nadadores, y durante el transcurso de la competencia se necesitará un control para su buena ejecución.

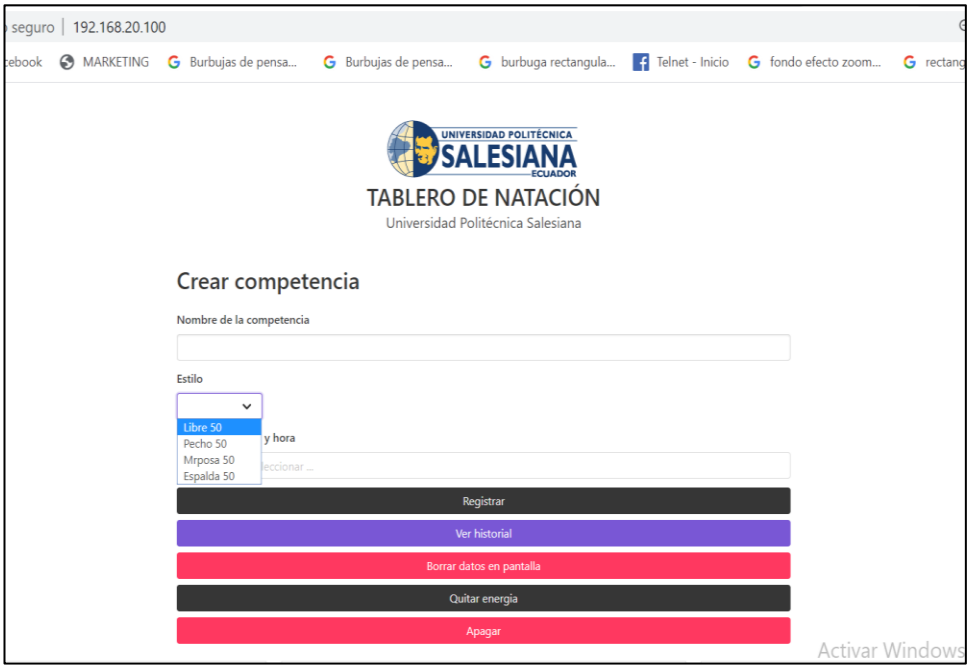


Figura 75- Menú desplegable para la elección del estilo de competencia de una lista de opciones

3.9.4 Ingresar datos de fecha y hora de la competencia

Este campo es necesario e imprescindible completarlo ya que será también un registro único e identificador para cada una de las competencias al momento de acceder a ellas posteriormente, entonces cuando una nueva competencia sea creada se guardará con su respectiva fecha y hora identificación única. Con esto se lograría contrarrestar posibles confusiones en competiciones ya acontecidas con nombres similares.

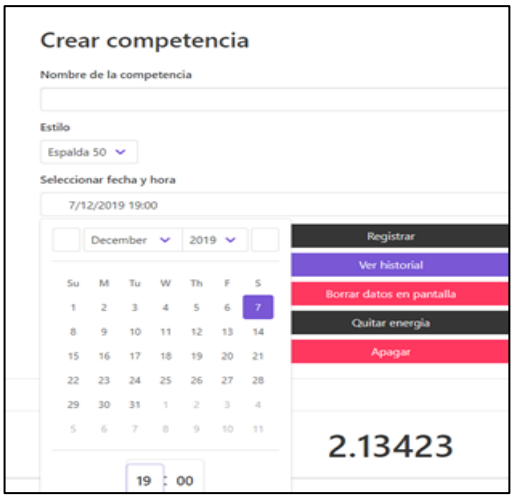


Figura 76- Opción desplegable para insertar hora y fecha de inicio de competencias.

3.9.5 Acciones principales de botoneras para interactuar con la interfaz web

Dentro de la primera interfaz web destinada para el registro de la competencia existen cinco botoneras con acciones importantes y principales dentro de la interfaz para administrar competencias.

Figura 77-Acciones principales del repositorio de competencias.

Primer botón (Registrar) de color negro permite registrar la nueva competencia y agregarla a la base de datos, donde una vez pulsado este botón permite redirigir a la nueva ruta para luego ingresar nombres de nadadores siempre y cuando se hayan completado los datos anteriores necesarios.

Un segundo botón (Ver historial) permite redirigir hacia una nueva ruta que permitirá visualizar mediante una página web con aspecto de repositorio visualizar y acceder

a antiguas competencias almacenadas en la base de datos como se muestra en la siguiente figura.

Figura 78- Repositorio de competencias deportivas de natación.

Figura 79-Repositorio de competencias deportivas de natación, vista de una competencia.

Con el tercer botón de color rojo (Borrar datos en pantalla) funciona con el fin principal de limpiar los datos que se encuentran mostrados en el visualizador.

El cuarto botón (Quitar energía) realiza la función de desconectar exclusivamente a los arduinos nano microcontroladores de la energía eléctrica protegiéndolos así de futuras picos de corrientes peligrosos no deseados en nuestra red eléctrica, también de posibles congelamientos del programa arduino debido a la gran cantidad de procesamiento de datos a alta velocidad, entonces debido a esto se planteó y se

decidió aplicar esta herramienta para que la transmisión de datos al visualizador no se vea afectada.

Y por último el botón (apagar) que simplemente realiza la acción de apagar correctamente la Raspberry con el fin de precautelar su vida útil dentro del proyecto.

3.9.6 Interfaz web para el registro de nadadores

Como anteriormente se menciona que una vez registrado el nombre de competencia y con su estilo de nado en la anterior ruta (192.168.20.100/) con dar un click en el botón “Registrar” nos redirige a una nueva ventana la cual nos muestra opción de ingresar información de los competidores relacionados a la competencia en curso y está dentro de la ruta (*1*92.168.20.100/nadadores) la vista de la siguiente interfaz con los siguientes recursos principales que me muestra en la figura a continuación.

La primera petición de información de esta ventana de la interfaz web utiliza un formulario para el ingreso de Nombre donde se debe de registrar el nombre del competidor, instantáneamente este dato se envía para guardarse en la base de datos.

El segundo recurso permite registrar Apellido relacionado al competidor, El tercer elemento utiliza una herramienta de menú desplegable que permite elegir entre las diferentes edades posibles dentro de la base de datos, a continuación semestre que de igual manera utiliza un menú desplegable y ahí se elige el semestre en el cual se encuentra cursando sus estudios el competidor tanto así su carrea ubicado en el cuarto recurso de la interfaz y el último recurso con menú permite elegir el carril disponible para que el competidor realice su actividad de competencia.

Una vez llenada la información individual de cada competidor obligatorio para cada uno de ellos se debe pulsar un botón (Agregar nadador) que permite agregar el competidor en la base de datos como su principal función.

Figura 80- Formulario para el registro de competidores.

3.9.7 Opción para eliminar competidores

Al terminar con el registro de la información de los competidores en esta interfaz se cerrará la opción del registro y cambiará de apariencia en la cual se agrega una acción que es un botón que utiliza un método http DELETE que su principal acción en el aplicativo web es de borrar competidores como se muestra en la siguiente figura y automáticamente se borran desde la base de datos.

Figura 81- Tabla de datos ingresados de competidores.

3.9.8 Inicio de carrera y visualización del cronometraje en tiempo real

Después de haber pulsado el botón que indica Iniciar carrera y que realiza una de las principales e importantes funcionalidades del proyecto y es la de dar inicio a una nueva carrera de nado y que a su vez este botón redirige a una nueva ruta (192.168.20.100/nadadores/competencia) y es en esta interfaz web donde el tiempo de cada uno de los competidores transcurrido durante una competencia se podrá visualizar de manera instantánea y periódica hasta el momento que el primer competidor llega a línea de meta para que su tiempo se pare automáticamente. Entonces una vez que todos los competidores hayan llegado a línea de meta con sus respectivas marcas de tiempos se podrá dar por terminada la competencia, pero no se podrá dar sin antes los datos visualizados en el marcador se actualicen de tal forma que se ordenen con respecto a la llegada esta particular característica servirá para facilitar la lectura de los espectadores y conocer los resultados instantáneamente.

3.9.9 Repositorio de anteriores competencias

El proyecto de titulación tendrá la capacidad de almacenamiento de todas las competencias acontecidas y la visualización posterior de ellas en la aplicación web con el fin de que tener expedientes como se muestra en la siguiente figura.

Figura 82- Repositorio de competencias deportivas de natación.

3.9.9.1 Transmisión de los datos ingresados en la interfaz web al visualizador

La comunicación entre la raspberry (servidor) con los diferentes arduinos nano controladores de las diferentes hileras de las placas led será de forma alámbrica utilizando el protocolo de comunicación I2C para la transmisión de datos.

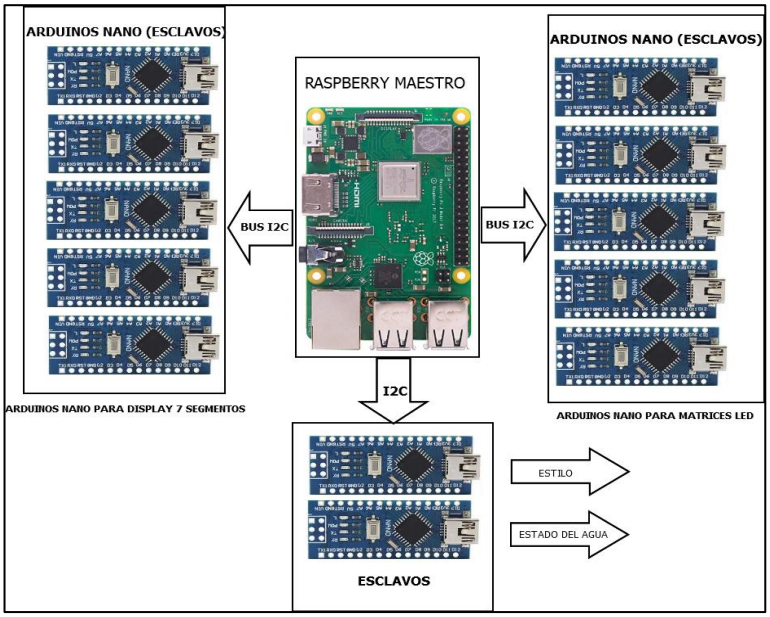


Figura 83- Comunicación entre raspberry con arduinos nano.

4 ANÁLISIS EXPERIMENTAL DE LA PLATAFORMA

4.1 Implementación del proyecto

En el presente proyecto de titulación y su implementación se propuso como principal objetivo el desarrollo y la realización de la plataforma para la visualización y cronometraje de competencias deportivas de índole acuático, a continuación, se muestra el esquema principal de funcionamiento.

4.2 Diagrama de bloques de la plataforma para visualizar y cronometrar competencias de natación.

El diagrama de bloque del esquemático del funcionamiento del proyecto se muestra en la siguiente figura.

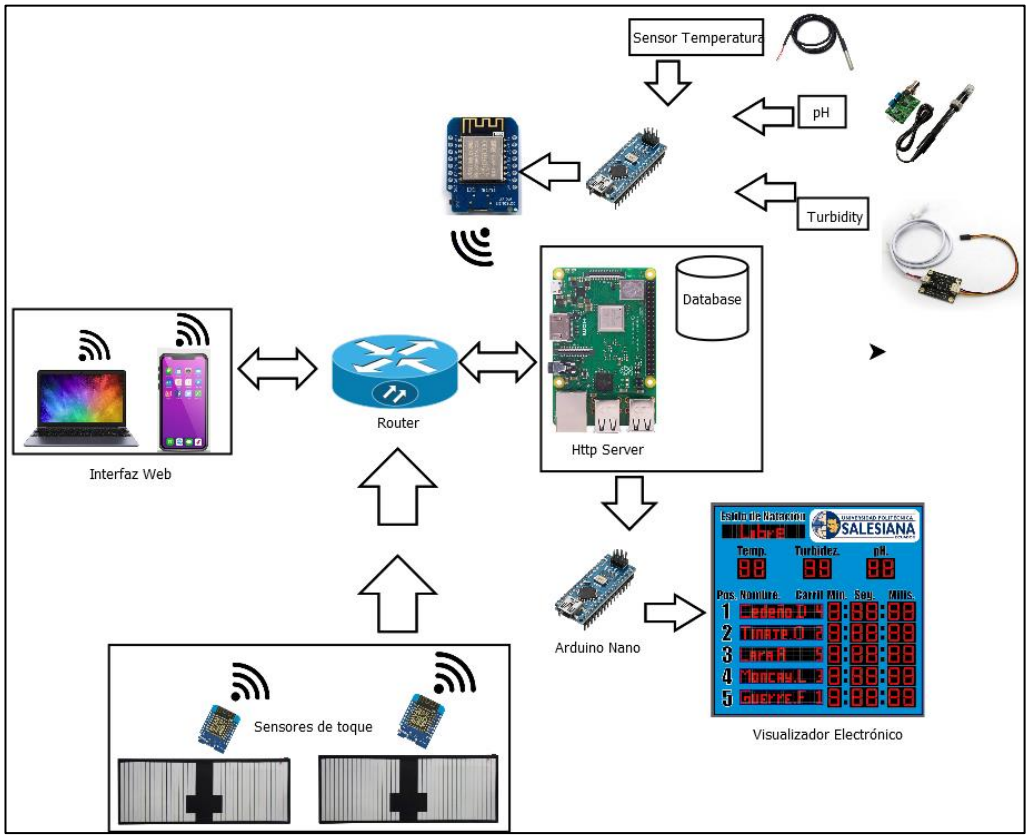


Figura 84- Diagrama de bloque del esquemático del funcionamiento del proyecto.

4.3 Armado de la estructura metálica del visualizador electrónico

Para el armado de la estructura metálica del visualizador que va alojar las 31 placas se tuvo que considerar la posición y la distribución de cada uno de los dispositivos dentro del marcador electrónico, todo se tomó en cuenta de tal manera que las placas (matrices 8x7 de diodos led y display numérico de 7segemento) tienen que encajar perfectamente para que se permita una correcta visualización y por su puesto de la lectura desde una distancia considerable con respecto a los espectadores, además de considerar el espacio que se necesita para las conexiones interna entre los elementos electrónicos.

Se diseñó previamente la estructura externa en el software AutoCAD, como se muestra a continuación en la figura tomando en cuenta las dimensiones de cada placa de diodos de alta luminosidad sea tanto para siete segmentos y matrices con respecto a la función de cada una, además de las placas electrónicas como Arduino, raspberry, etc. Realizado esto se pudo obtener el tamaño ideal de la estructura del visualizador, con todo esto se puede satisfacer los objetivos principales del proyecto, así distribuyendo las diferentes placas de diodos para indicar el estilo de la competencia en la parte superior derecha, indicar los niveles del estado físico químico del agua, nombre de competidores con su respectivo tiempo cronometrado.

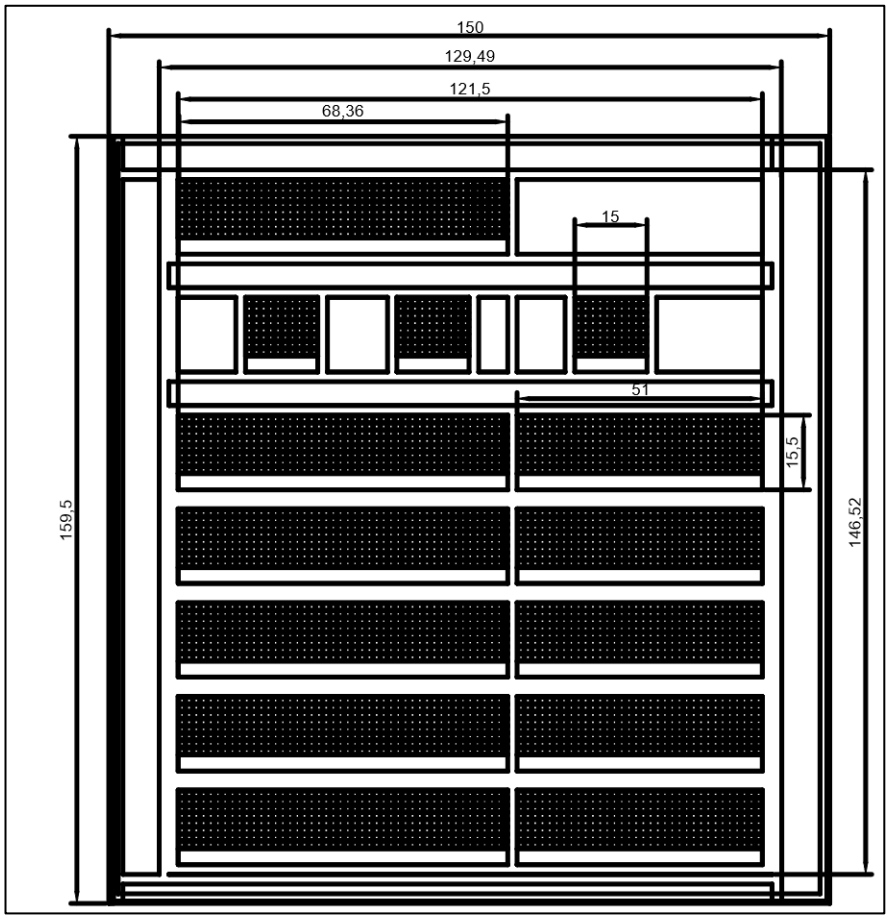


Figura 85- diseño de estructura del visualizador electrónico en AutoCad.

Figura 86- Armando estructura externa del visualizador

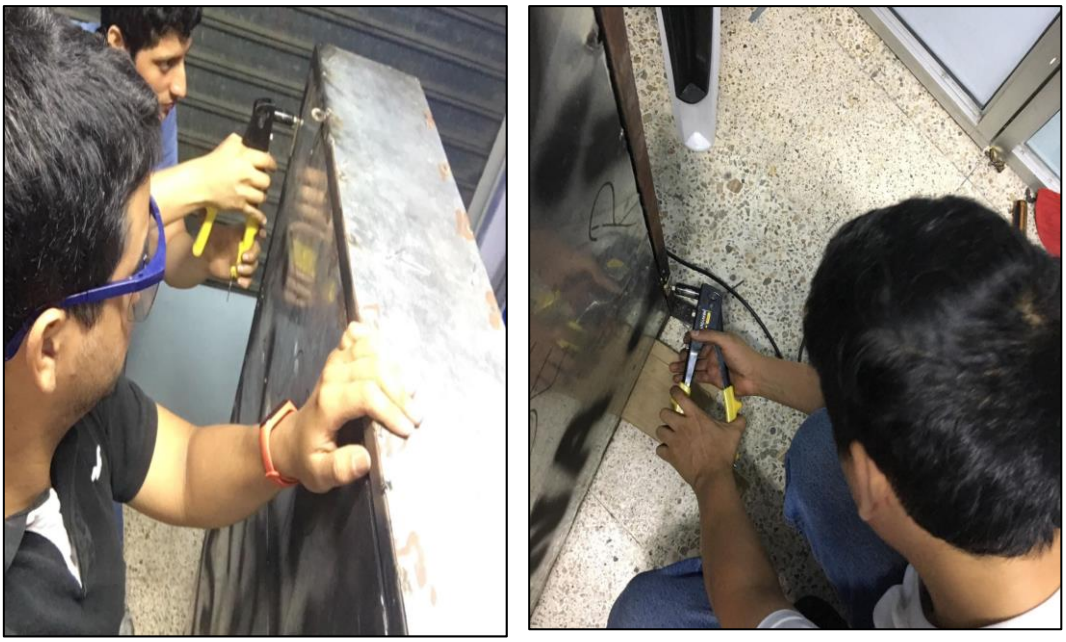


Figura 87- Ubicación e instalación de las puertas

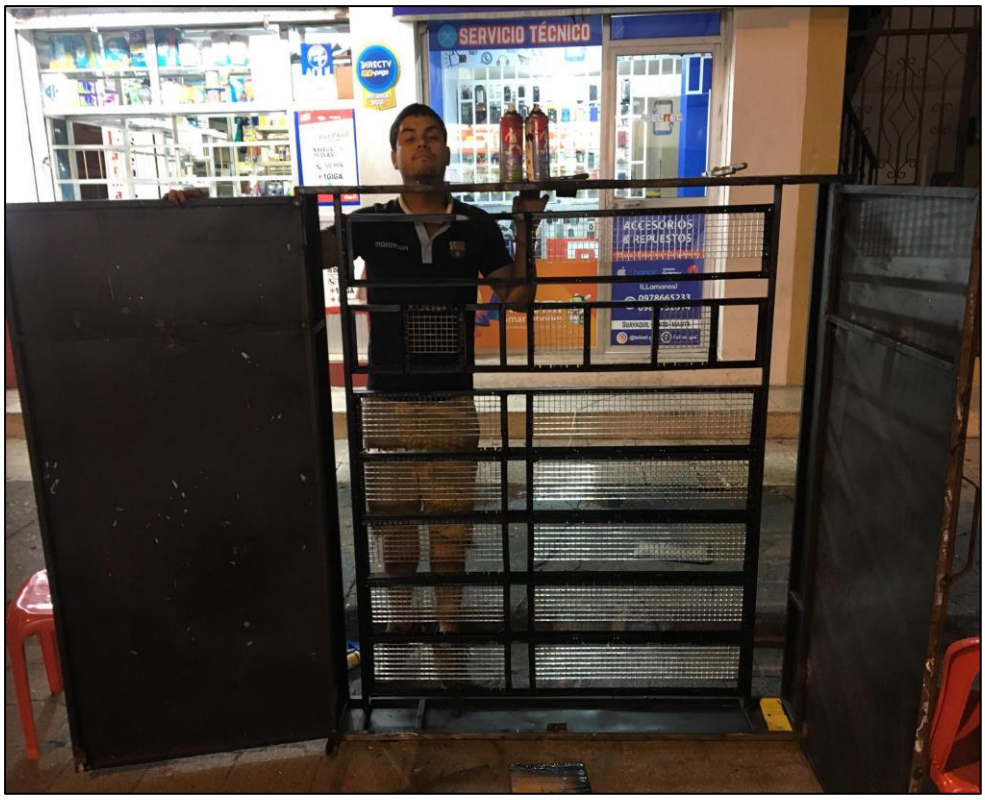


Figura 88- Estructura del visualizador

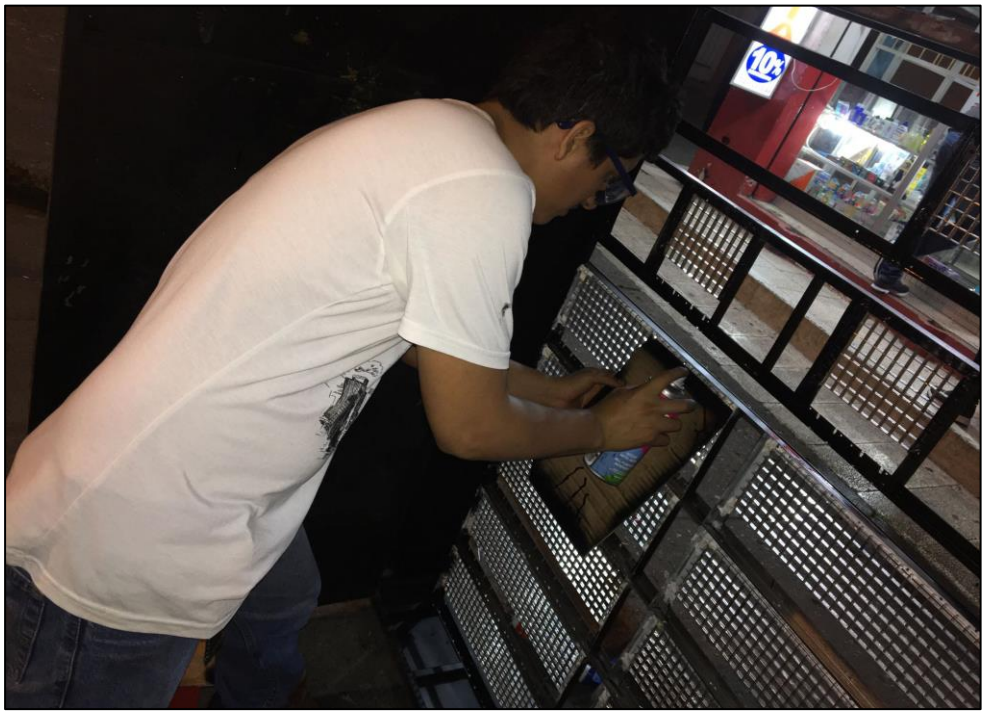


Figura 89- Pintado de la estructura del visualizador

4.4 Montaje de las placas electrónicas de diodos LED

El montaje de las 31 placas numéricas de 7 segmentos y de las 30 placas de matrices que mostraran caracteres se realizó asegurándolas con pequeños tornillos en los perfiles de metal que le corresponde a cada una de las hileras en el armazón previamente elaborado.

Figura 90- Montaje de placas electrónicas de hileras diodos led

4.5 Cableado y ubicación de las placas microcontroladoras y módulo central del proyecto.

Figura 91- Cableado y ubicación placas electrónicas de hileras diodos led

Figura 92- Cableado y ubicación de las placas microcontroladoras y módulo central.

4.6 Prueba del funcionamiento de las placas LED del visualizador

Para la primera prueba funcional de todos los elementos en conjunto para que el visualizador opere correctamente primero se debe constatar que la fuente de alimentación de 5v de corriente directa se encuentre operando, una vez cumplido lo anteriormente mencionado se procede a constatar que los arduinos que controlan las hileras de las placas LED se encuentren energizados correctamente.

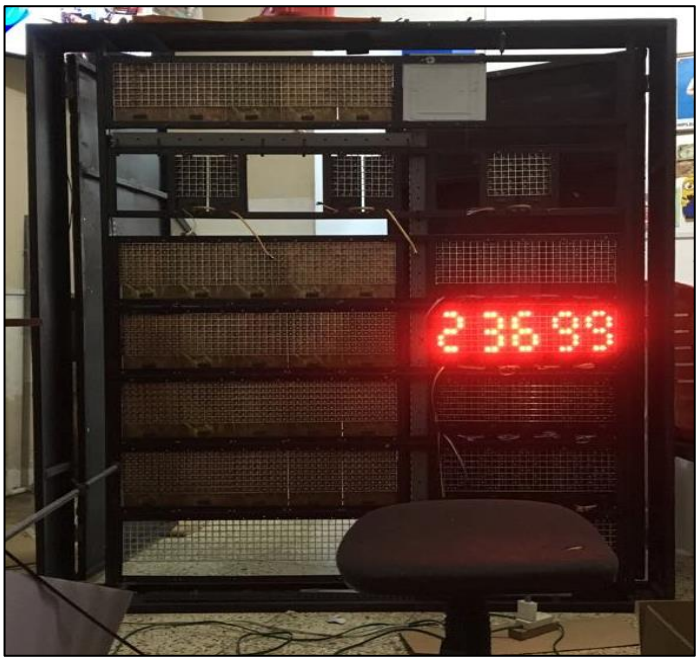


Figura 93- Cableado, encendido y prueba de las placas de display siete segmentos.

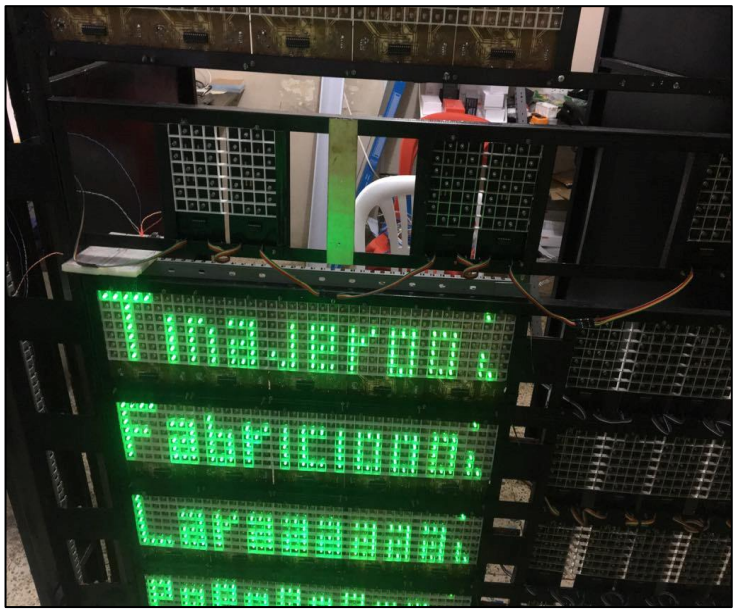


Figura 94- Encendido de las placas 8x7 que conforman el marcador electrónico.

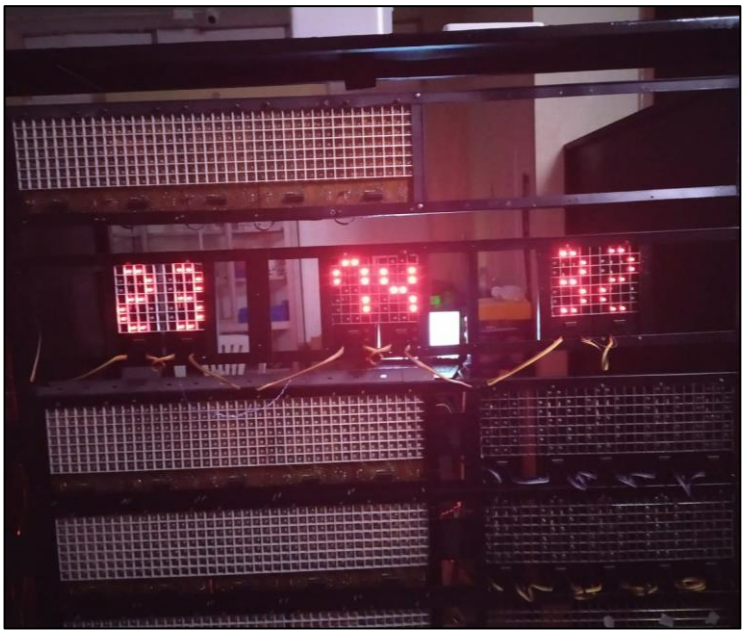


Figura 95- Placas display siete segmentos para mostrar el estado físico-químico del agua.

4.7 Análisis Experimental de la interfaz web

En la etapa experimental de prueba de posibles errores y correcciones en el funcionamiento de la plataforma (software y hardware) que forman parte del visualizador electrónico tales como el aplicativo web como parte del software y los diferentes componentes electrónicos que hacen posible que el objetivo general del proyecto se cumpla.

Para el aplicativo web en el lado del servidor o también llamado Back-end y la creación de las diferentes rutas se desarrolló la programación de manera generalizada a través de Python con su versión 3 y en específico con su framework Flask para que nuestra interfaz web pueda cumplir tareas específicas de manera dinámica.

4.8 Acceso a la interfaz web para dar inicio y administrar competencias

Cuando el computador o dispositivo electrónico móvil se encuentre dentro de la misma red en la cual se encuentra el servidor que aloja la interfaz web, una vez ahí el usuario podrá acceder a la interfaz realizando una petición al servidor mediante el (*url: 192.168.20.100/*) para que el servidor pueda responder correctamente deberá estar completamente activo y constantemente escuchar peticiones de usuarios, si se cumple lo anteriormente mencionado el servidor contestará con la página web principal de la interfaz como se muestra en la imagen a continuación.

Figura 96- Respuesta del servidor con la interfaz web vista desde un smartphone.

En caso contrario que el servidor que aloja a la interfaz envíe una respuesta de una página web con mensaje de un código 404 hay dos posibles opciones de errores en el cual el primero será que el dispositivo no se encuentre dentro de la misma red en la que se encuentra el servidor en este caso será una red interna privada con “SSID: SALESIANA2020” o en el segundo caso que puede ser más común que el servidor (Raspberry) se encuentre desconectada a la energía eléctrica o su servicio este desactivado a escuchar peticiones de usuarios, unos de los mensajes que se podrá visualizar en cualquiera de estos diferentes casos se mostrará en la siguiente imagen.

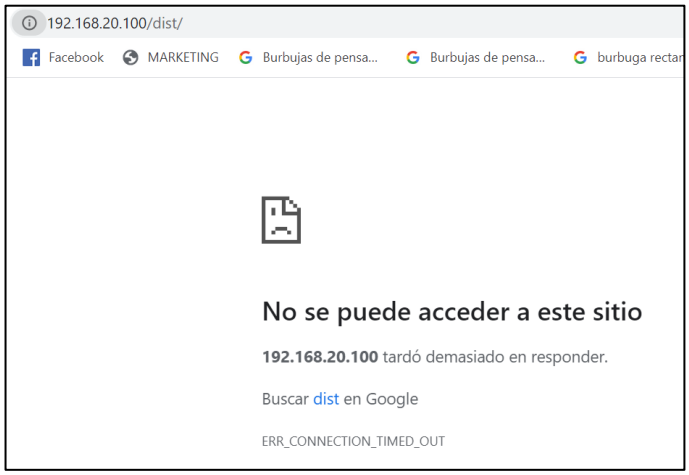


Figura 97- Mensaje de error al acceder al servidor visto desde un pc.

4.9 Prueba de ingreso de datos en la interfaz web

Para los nombres de los competidores se puedan mostrar en el visualizador se deben de registrar previamente en la interfaz web con el nombre de la competencia y su estilo.

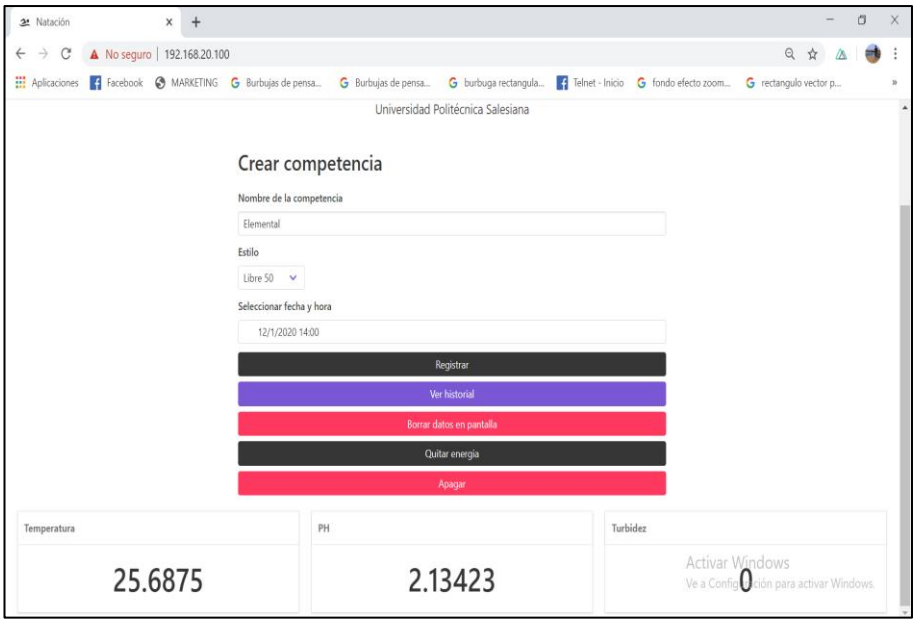


Figura 98- Registro de competencia desde un competidor.

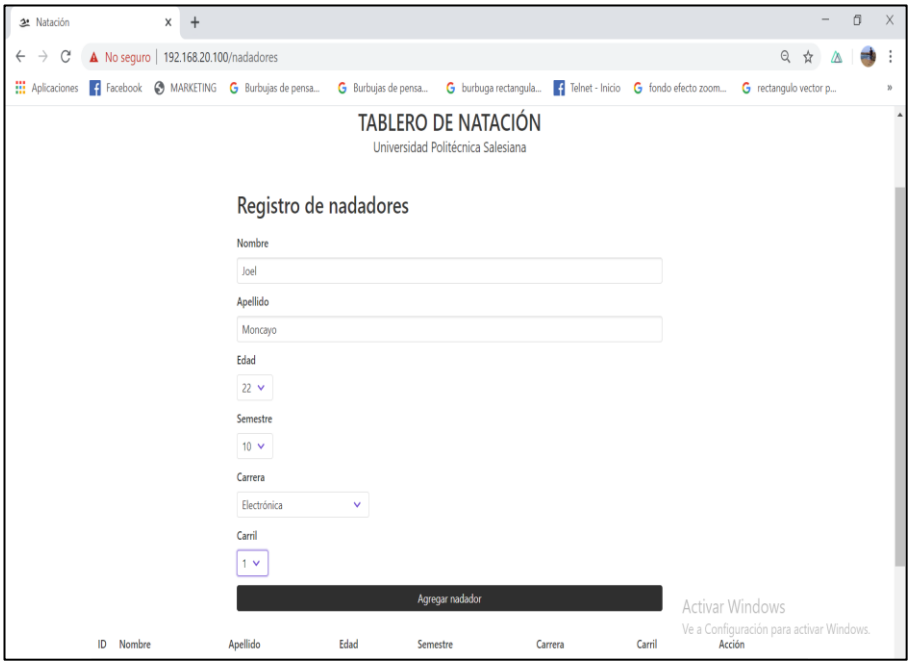


Figura 99- Registro del formulario de registro de competidores.

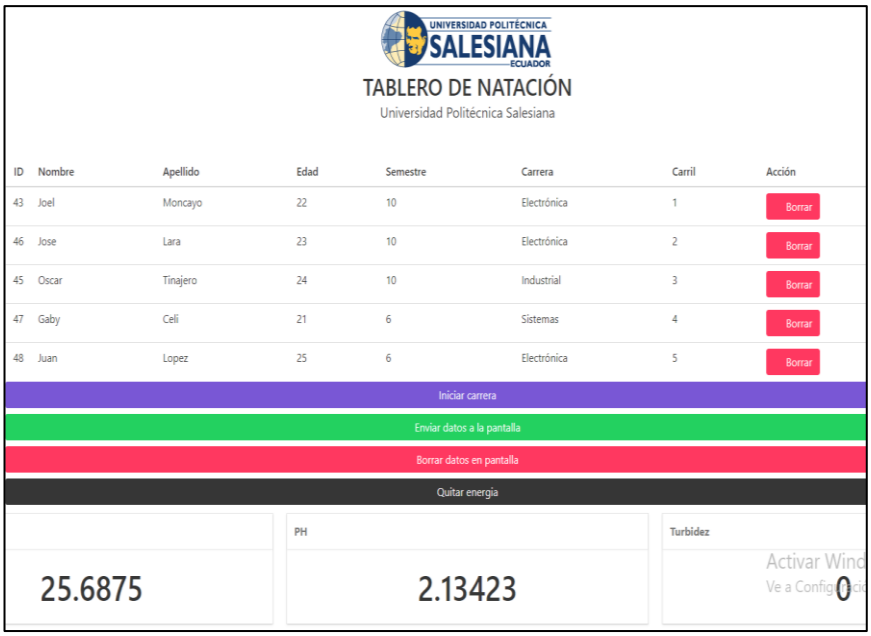


Figura 100- Imagen del registro de competidores completado.

4.10 Prueba visualización de datos enviados desde la interfaz web al visualizador electrónico

Toda la información principal de cada competencia y de cada competidor tienen que mostrarse tanto dentro de la interfaz como en el visualizador, cabe recalcar que los datos para que se muestren en el tablero electrónico tienen que ser enviados desde el aplicativo hasta cada uno de los microcontroladores arduino de cada hilera de placas de diodos led mediante el protocolo de comunicación i2c.

Para la comunicación entre la raspberry que se programó dentro del protocolo i2c como maestro y con los arduinos microcontroladores que son los esclavos, el envío de datos es muy sensible debido a algunos factores propios de un medio físico y uno de ellos es la longitud de los cables entre las conexiones, lo cual produce que eventualmente los datos se pierdan inevitablemente y que afecta de tal manera que los datos no se vean reflejados en el visualizador, para este evento fortuito se pensó y planteó una solución de un corte de energía tipo “reset” forzado con la ayuda de un dispositivo electrónico relé conectado a la alimentación de los arduinos microcontroladores para tratar de mitigar y recuperar los datos perdidos entre las placas. Garantizando así la visualización correcta de todos los datos de las competencias.

Figura 101- Datos obtenidos de la interfaz web enviados al visualizador electrónico.

4.11 Prueba de sensibilidad y respuesta de los sensores de toques.

Una vez realizada la respectiva prueba de los datos visualizados, a continuación se procede a la prueba de la señal inalámbrica enviada al Raspberry desde el dispositivo wemos en respuesta a cada sensor de toque tiene que ser exacta, puesto que es el elemento que definirá las competencias acuáticas, y para ello es indispensable realizar pruebas previas para determinar la sensibilidad ideal, para ello se logró calibrar mediante la tarjeta wemos en la sensibilidad de recepción de la señal de voltaje generada por los piezoeléctricos con el objetivo que cumpla su principal función de detener en el momento preciso el tiempo cronometrado de cada competidor. La calibración y señales wifi (interferencias) en el ambiente fueron puntos en desventaja ya que fue muchos intentos de pruebas y errores.

Luego de haber superado estas dificultades se suma una adicional que consistía que cuando los wemos se conectaban a la red satisfactoriamente esperan por señales de entrada para poder transmitir el dato inalámbricamente, en caso que no existir aquella señal, lo wemos entran en un modo llamado “hibernación” lo cual nos dimos cuenta que no emiten ninguna señal de salida por lo tanto esto afectaba que en el momento preciso que si existía señal de entrada producida por el toque de la llegada de algún competidor la señal wifi que debería emitir el wemos al raspeberry para que detenga el carril se producía con un cierto retardo no deseado afectando directamente a todos

los competidores y por su puesto a sus tiempos, entonces para tratar de eliminar el “error” se planteó que la tarjeta del wemos siempre debe encontrarse operativa enviando señal hacia el raspberry durante la competencia y todo esto con el principal objetivo que el modulo wifi no entre en la llamada “hibernación”, una vez aplicado los cambios planteados se obtuvo los resultados deseados.

4.12 Análisis Experimental de los sensores del estado físico-químico del agua

En la realización de pruebas del funcionamiento de los tres diferentes sensores que receptan y procesan los datos lo primero que se cumplió fue la calibración para lograr un eficiente rendimiento con un mínimo margen de error.

Los datos obtenidos desde los sensores son leídos por un arduino y enviados inalámbricamente por el wemos, donde podemos visualizados en el monitor serie del IDE de arduino, la prueba experimental se la realizó en tres tipos de agua donde tienen diferentes características físicas químicas para aumentar la veracidad de los sensores. El sensor de temperatura se mostrará en grados centígrados, la turbidez en ppm, y por último el pH en rangos de acides o de alcalinidad del agua.

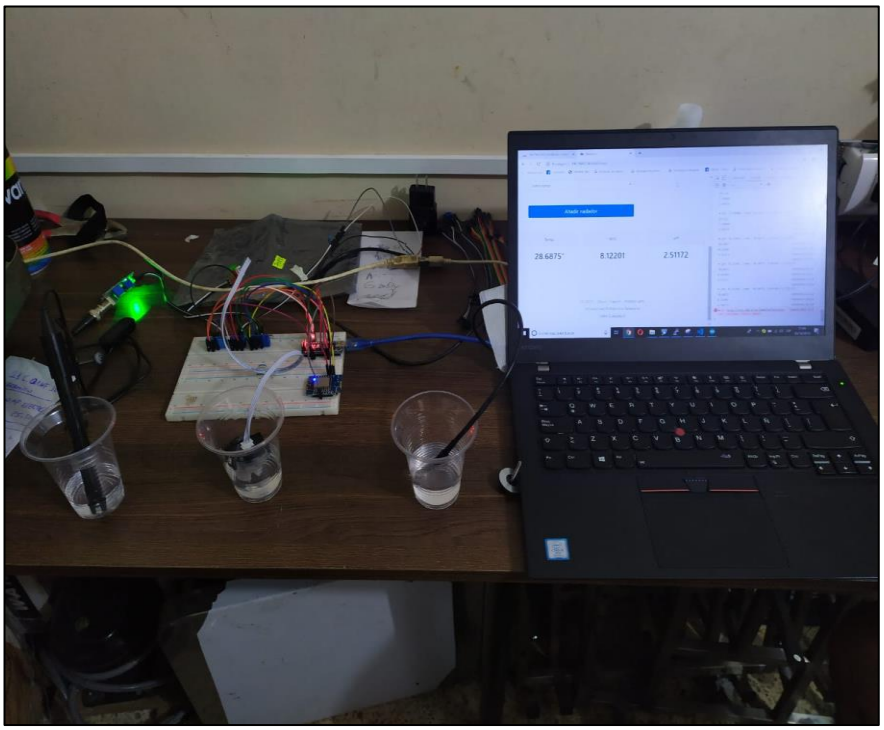


Figura 102- Prueba y calibración de los tres sensores del estado del agua.

CONCLUSIONES

Al culminar el diseño e implementación del proyecto de tesis para la obtención del título de ingeniero electrónico se pudo tener un sentimiento de haber cumplido con esfuerzo, pero con éxito la meta principal en la parte del desarrollo del visualizador electrónico ya que será un producto que estará al servicio de la universidad y de toda su comunidad que practique el deporte de la natación y que de ahora en adelante hará uso del mismo.

Se pudo observar en todas las etapas en la que se desarrolló e implementó nuestro proyecto que se pusieron en práctica todos los conocimientos y experiencias adquiridos durante el periodo de aprendizaje dentro de la carrera de Ingeniería Electrónica en la Universidad Politécnica Salesiana.

Durante la etapa destinada para la elaboración de la plataforma que involucra tanto el hardware como software del sistema de cronometraje de competencias deportivas acuáticas se han usados todos los conocimientos fundamentales de programación necesarios tanto para el diseño del sistema que brinda una interfaz interactiva con el usuario con su front-end en el lado del cliente desarrollado bajo un framework Vue.js y con su back-end desarrollado en el lado del servidor con python y con su framework flask.

En la creación de base de datos MySQL que fue implementada dentro del servidor web y accedida mediante una interfaz web llamada phpmyadmin, se tuvo que adquirir nuevos conocimientos fuera de universidad para poder cumplir con este objetivo satisfactoriamente, también se optó por implementar el servidor http de apache, y la parte de programación de los dispositivos que conforman la electrónica como los dispositivos de adquisición de datos del estado físico químico del agua, los sensores de toque, arduinos microcontroladores de placas de diodos led para la visualización y los demás dispositivos son programados bajo el IDE de arduino y con las telecomunicaciones en la configuración del router que es utilizado para crear nuestra propia red interna permitiendo a cualquier dispositivo acceder al servidor que aloja a la interfaz web y además de la transmisión y adquisición de datos de los dispositivos enlazados de manera inalámbrica.

Se concluyó como resultado que las pruebas realizadas en el desarrollo de la plataforma fueron un principal aporte para la mejora de la misma, para el desarrollo de la plataforma en el software y la culminación total de la plataforma con resultados exitosos y eficiente.

RECOMENDACIONES

La primera recomendación a tener presente para el proyecto del visualizador electrónico es que los dispositivos tanto como el Raspberry pi y los arduinos son elementos sensibles a la manipulación directa y por lo cual se sugiere que deben de estar protegidos en todo momento ósea antes, durante y después de su operatividad de tal manera que su manipulación tiene que ser con herramientas especiales, además de no estar expuesto a temperaturas muy altas ni a una alta humedad, estos son datos de seguridad que proporcionan los desarrolladores para evitar el deterioro del equipo.

La segunda recomendación es de tomar precauciones con respecto al voltaje y corriente para la alimentación de entrada sugiere adecuar una toma de energía conectada a tierra con el fin de que el visualizador con todos sus dispositivos internos se encuentre totalmente protegidos ya que en caso de generarse un cortocircuito no se pueda producir daños en los mismos.

Además, cabe recalcar que el visualizador electrónico al estar a la energía eléctrica suministrada por la universidad tiende a ser un costo adicional dentro de los servicios básicos fijos, entonces por lo consiguiente se recomienda implementar módulos para paneles solares con el fin de ser un proyecto autosustentable, amigable con el medio ambiente y con el bolsillo de la universidad.

Se recomienda que en caso de un posible daño del router que crea una red privada local al momento de un reemplazo se debe de elegir otro de la misma marca y modelo, y su configuración interna procurar que la red y su ip estática destinada al dispositivo raspberry que es el servidor central del sistema sea la misma ya que si no se cumple este requerimiento ocurriría conflictos internos al momento de intentar acceder, por lo tanto el proyecto quedaría totalmente incapacitado de realizar sus principales objetivos.

En el caso específico de la parte de las telecomunicaciones, el acceso a la red por parte del dispositivo raspberry obligatoriamente tiene que ser mediante un medio físico que en este caso es cable de dato, este cable de datos es propenso a daños de cualquier índole, y que el caso que suceda lo anteriormente mencionado se debe de reparar el mismo o reemplazarlo inmediatamente para que el proyecto siga operativo y cumpliendo con sus objetivos principales.

En el caso de posibles daños de las tarjetas microcontroladoras arduinos se recomienda reemplazarlos con otros de la misma marca y modelo ya que de esta manera se garantiza el correcto uso de los recurso y funcionalidad del proyecto.

CRONOGRAMA

	AÑO 1												AÑO 2			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1.COMPRA DE EQUIPOS																
1.1 Cotización de los elementos a usar																
1.2 Compra de equipos e implementos																
1.3 Pruebas de funcionamiento individual																
2.DISEÑO DE LAS PLACAS ELECTRONICAS LED																
2.1 Construcción de la parte electrónica																
3.DESARROLLO DEL SOFTWARE																
3.1 Programación en Raspberry																
3.2 Programación en Arduino																
3.3 Diseño del servidor web y base de datos.																
4.Construcción de estructura física																
4.1 Diseño e implementación de la estructura física de la parte del visualizador electrónico.																
4.2 Diseño e implementación de la estructura física de la parte se los sensores de toque piezoeléctrico subacuáticos.																
5. DESARROLLO DEL PROTOTIPO FINAL																
5.1 Verificación de funcionamiento																
5.2 Realización de correcciones y pruebas																

PRESUPUESTO

Descripción	Cantidad	Precio	Subtotal
Placas de baquelitas de cobre A4	20	$ 3,00	$ 60,00
Placas de baquelitas de cobre A4 doble cara	18	$ 6,00	$ 108,00
Percloruro de Hierro (1KG)	2	$ 7,00	$ 14,00
Brocas de taladro 1mm	20	$ 1,00	$ 20,00
Discos de corte DREMEL	1	$ 7,00	$ 7,00
Packet de papel fotográfico	1	$8,00	$ 8,00
Resistencias 330 ohm	300	$0,04	$ 12,00
Resistencias 4.7kohm	26	$0,04	$ 1,04
Rollo Estaño Soldar Hilo De 0.8mm	3	$6,00	$ 18,00
Borneras Triples	40	$0,10	$ 4,00
Borneras dobles	100	$0,10	$ 10,00
Cable Jumper 40 Pines 20cm M-M	6	$2,50	$ 15,00
Cable Jumper 40 Pines 20cm H-M	6	$2,50	$ 15,00
Cable Jumper 40 Pines 20cm H-H	12	$2,50	$ 30,00
Espadachin 40 Pin	20	$0,25	$ 5,00
Diodos led alta luminosidad (GREEN)	1500	$ 00,10	$ 150,00
Diodos led alta luminosidad (RED)	1000	$ 00,10	$ 100,00
I.C. 74hc595	31	$ 1,00	$ 31|,00
I.C. MAX7219	30	$ 1,50	$ 45,00
Sirena de alarma 12v	1	$ 5,00	$ 5,00
piezoeléctricos	25	$ 0,25	$ 6,25

Sensor De Ph + Tarjeta Acondicionamiento Arduino	1	$ 50,00	$ 50,00
Sensor analógico tds	1	$ 25,00	$ 25,00
Sensor sonda de Temperatura Digital Ds18b20	1	$ 5,00	$ 5,00
Módulo Rele 5v 1 Canal	2	$ 2,50	$ 5,00
Arduinos nano	14	$ 5,50	$ 77,00
Raspbery Pi 3	1	$ 70,00	$ 70,00
Wemos D1 mini	6	$ 13,00	$78,00
Router Tp-link Wr-840n 300mbps	1	$ 25,00	$25,00
Vinil negro 4x4mts	1	$ 20,00	$ 20,00
Cajas Plasticas negras	5	$ 4,00	$ 20,00
Cajas Plasticas exteriores	2	$ 7,00	$ 14,00
Fuente 5v- 5A DC	1	$ 8,00	$ 8,00
Pintura spray negro- morado	2	$ 2,00	$ 4,00
Pintura anticorrosiva negra	1	$ 8,00	$ 8,00
Estructura del visualizador electrónico	1	$ 120,00	$ 120,00
Acrílico negro, parte frontal de la estructura de visualizador electrónico	1	$ 70,00	$ 70,00
Estructura de los paneles de toque	5	$ 60,00	$ 300,00
Gastos varios	1	$200,00	$ 200,00
		Total	**$ 1763,29**

REFERENCIAS

Wallace, M. R. S. (2012). *Getting Started with Raspberry Pi*. Maker Media, Inc. Retrieved from https://books.google.es/books?hl=es&lr=&id=xYhMlilTwC4C&oi=fnd&pg=PR2&dq=raspberry+pi&ots=W46fiDnau4&sig=doTOmuD2Pbjm6DPhf87ceSTKmmk#v=onepage&q=raspberry pi&f=false

Blum, C. B. (2013). Sams Teach Yourself Python Programming for Raspberry Pi in 24 Hours.

Monk, S. (2012). Programming the Raspberry Pi: Getting Started with Python.

Banzi, M. (2008). Getting Started with Arduino.

Arduino. (2008).Arduino Nano. Retrieved from arduino.cc: https://www.arduino.cc/en/uploads/Main/ArduinoNanoManual23.pdf

https://disenowebakus.net/. (2013). Estándares Web W3C - Qué son, cómo funcionan y para qué sirven. *Diseño de Páginas Web, Sitios de Internet y Posicionamiento SEO Akus.Net*. Retrieved from https://disenowebakus.net/estandares-web.php

Alvarez, M. A. (2003). Qué es Python. Retrieved July 24, 2018, from https://desarrolloweb.com/articulos/1325.php

Grados, J. G. (n.d.). ¿Qué es JavaScript? Retrieved July 24, 2018, from https://devcode.la/blog/que-es-javascript/

Eguíluz Pérez, J. (n.d.). Introducción a JavaScript. Retrieved from www.librosweb.es

Rosa, J. M. (2017). ¿Qué es Vue.js? | OpenWebinars.net. Retrieved July 24, 2018, from https://openwebinars.net/blog/que-es-vuejs/

Macrae, C. (n.d.). *Vue.js: Up and Running: Building Accessible and Performant Web Apps.*

Fossati, M. (2014). Todo sobre MySQL: Libro ideal para ingresar en el mundo de la base de datos ... - Natsys - Google Libros. Retrieved July 24, 2018, from https://books.google.com.ec/books?id=GS3kAgAAQBAJ&printsec=frontcover&dq=mysql&hl=es&sa=X&ved=0ahUKEwjlxMPZnrncAhVRq1kKHStSB6EQ6AEIMzAC#v=onepage&q=mysql&f=false

Purdum, J. J. (2012). Beginning C for Arduino: Learn C Programming for the Arduino.

ANEXO 1

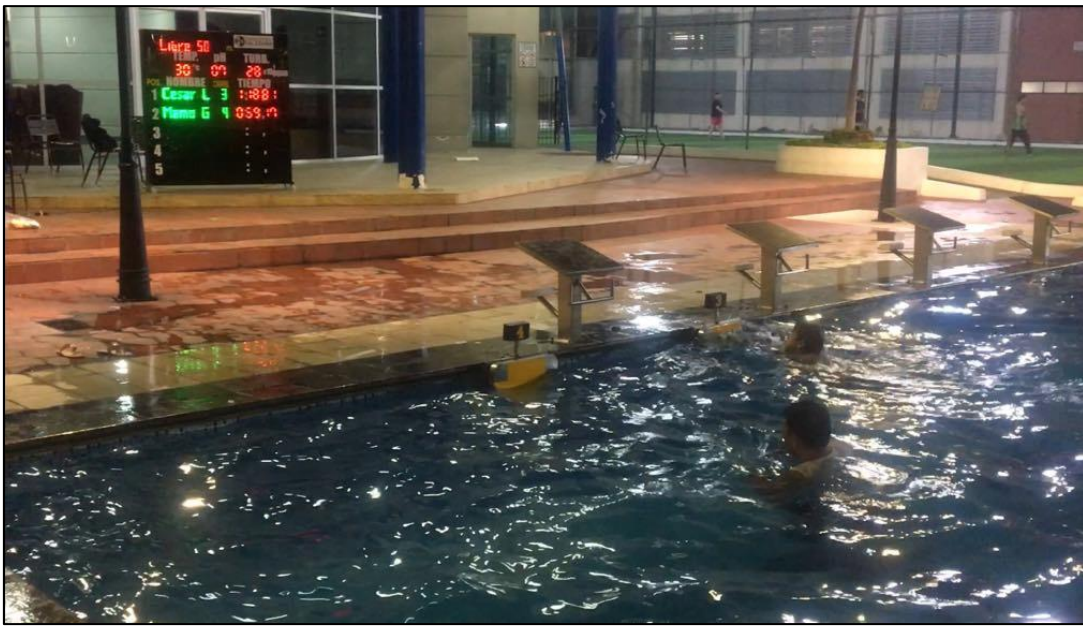
IMPLEMENTACIÓN EN LAS INSTALACIONES DE LA UNIVERSIDAD DEL VISUALIZADOR ELECTRÓNICO

Anexo 1.1- Visualizador electrónico implementado en UPS.

Anexo 1.2- Implementación y prueba de visualización del marcador.

Anexo 1.3- Implementación de los sensores para el estado del agua.

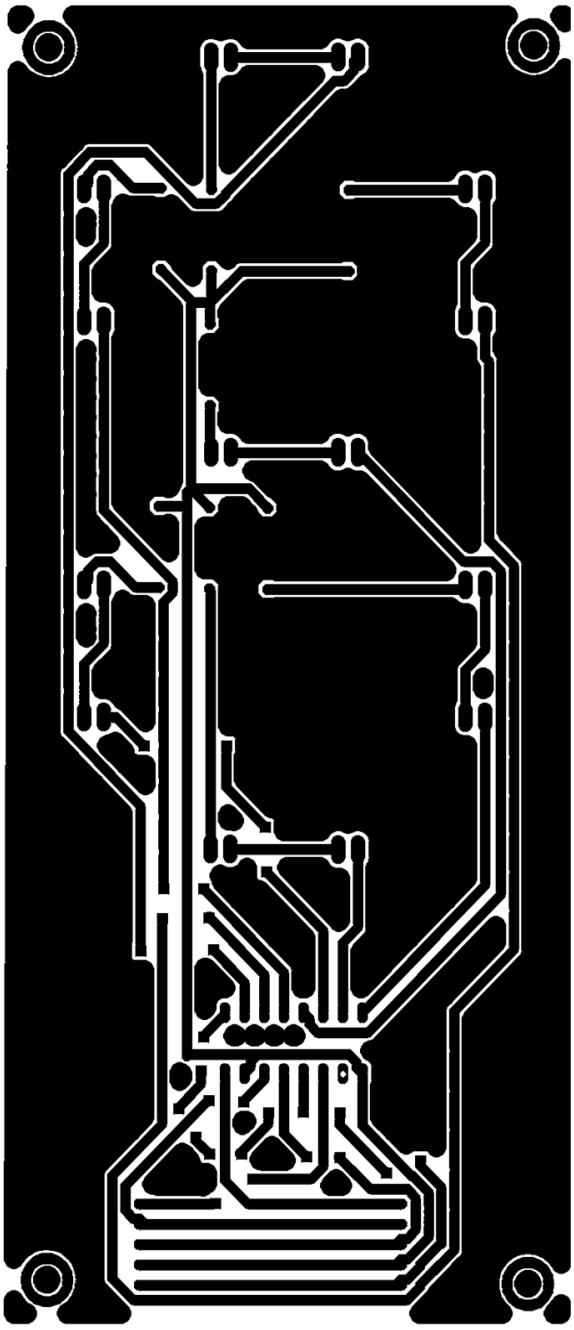
Anexo 1.4- Proyecto implementado y completamente funcional.



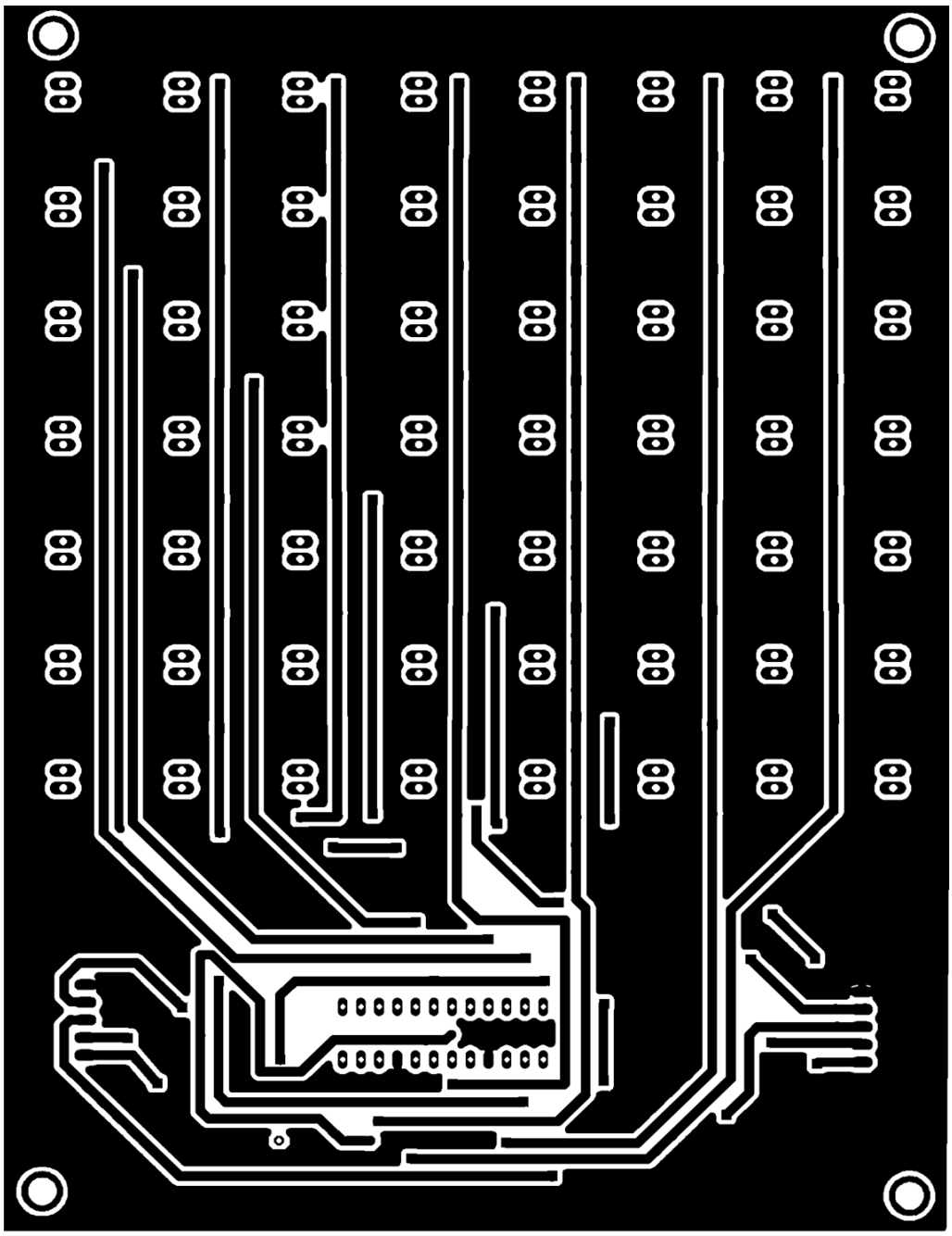
Anexo 1.5- Prueba con competidores.

ANEXO 2

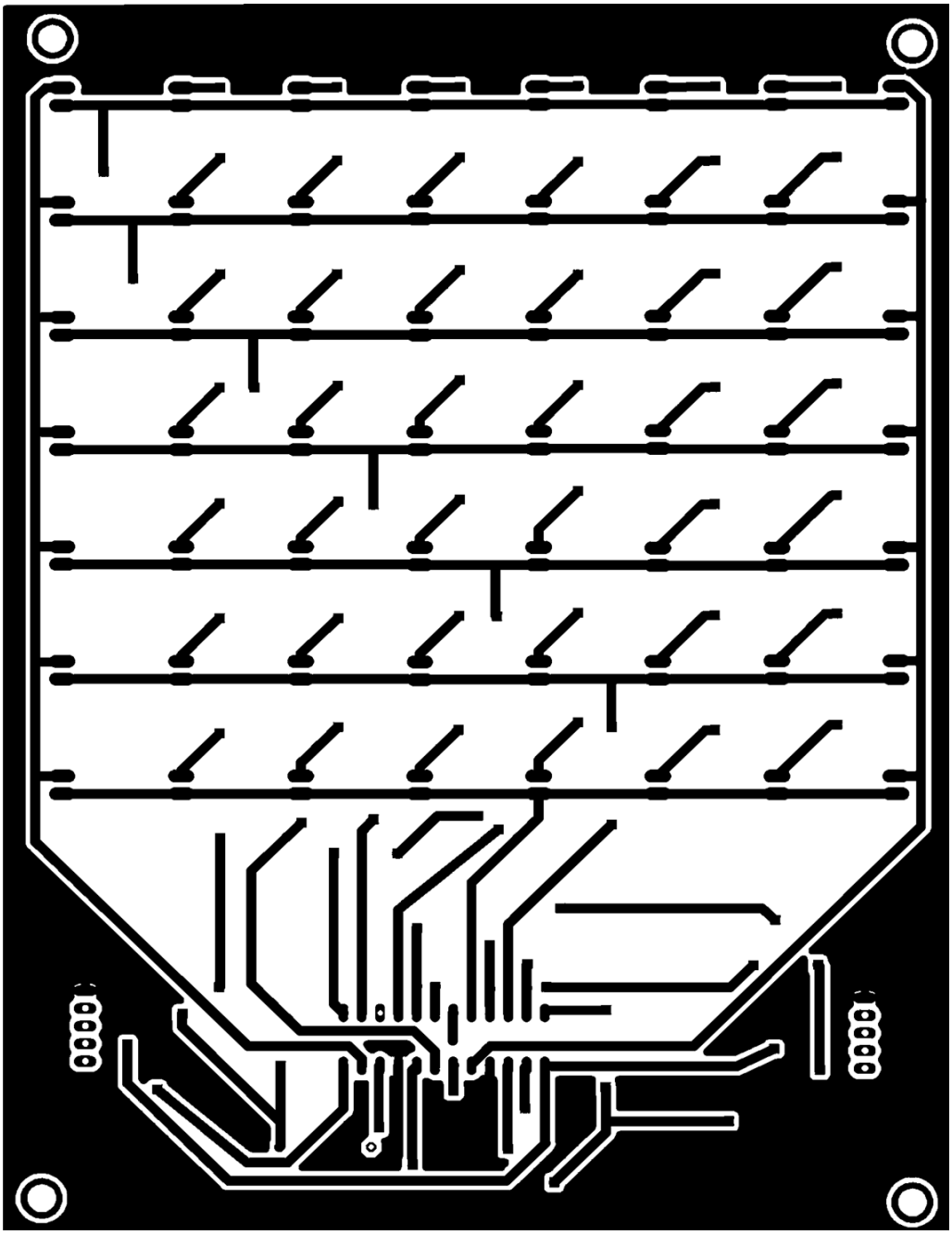
CIRCUITOS IMPRESOS PCB



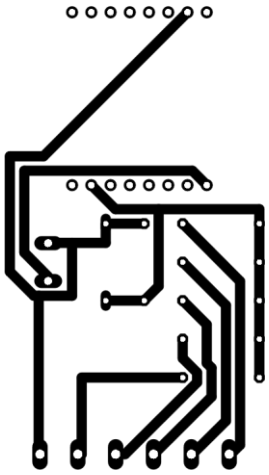
Anexo 2.1- Diseño de PCB de display de 7 segmentos de leds



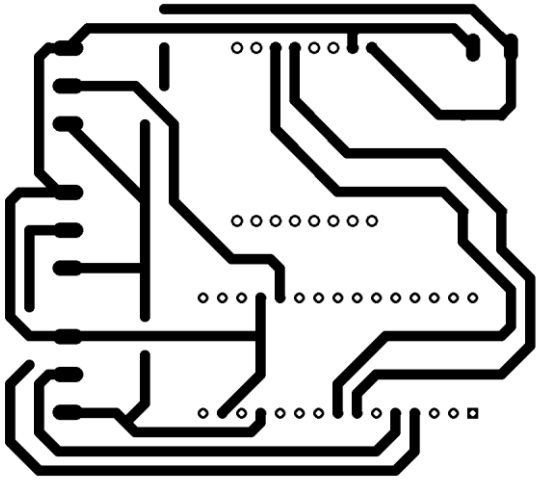
Anexo 2.2- Diseño de PCB de matriz 8x7 de leds vista frontal



Anexo 2.3- Diseño de PCB de matriz 8x7 de leds vista trasera.



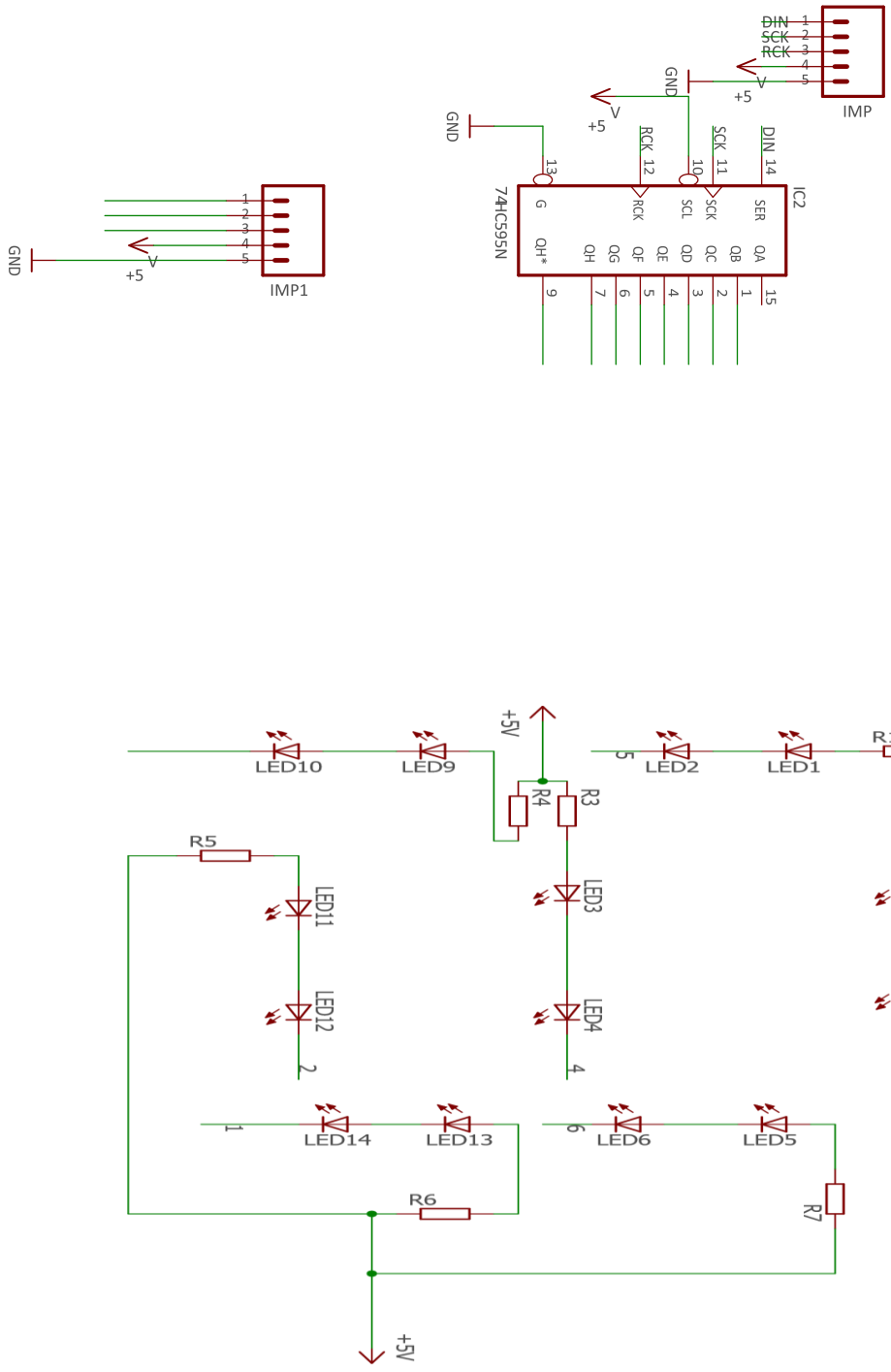
Anexo 2.4- Diseño de PCB de conexión entre piezoeléctrico y wemos.



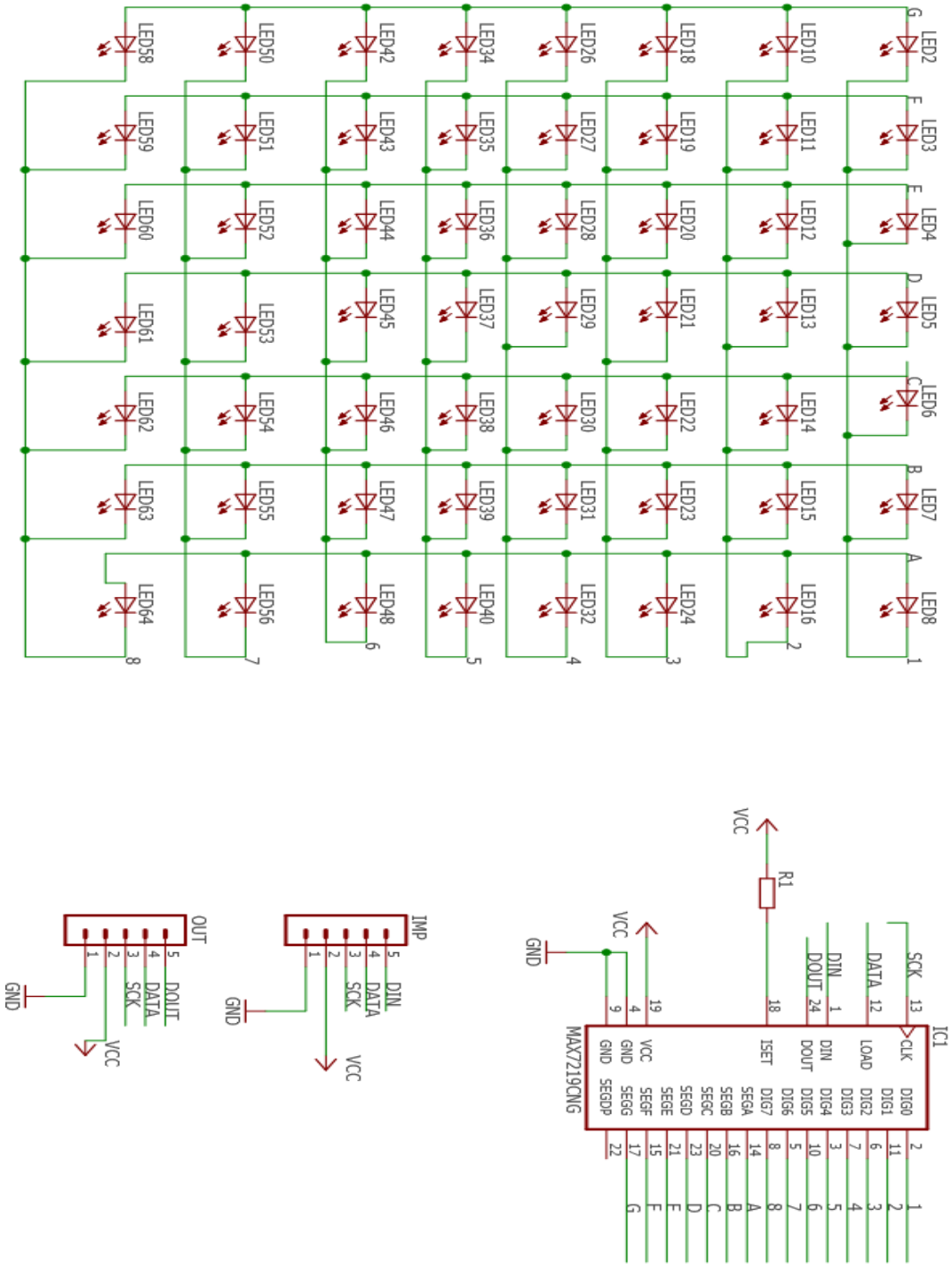
Anexo 2.5- Diseño de PCB placa para sensores del estado del agua.

ANEXO 3

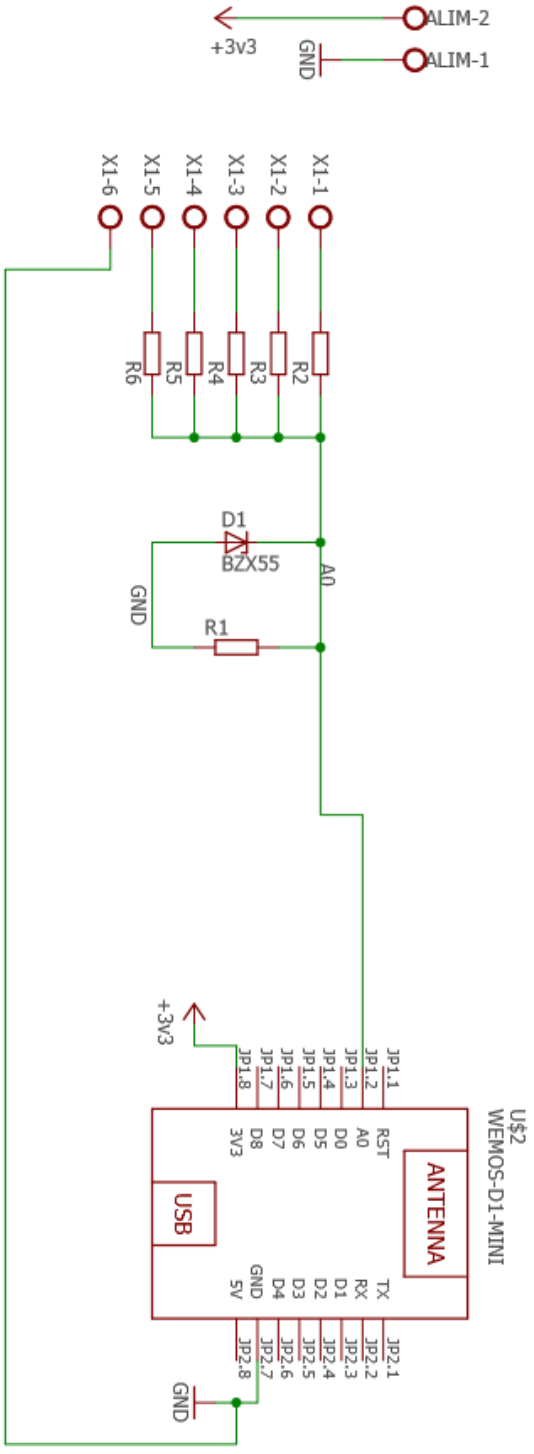
ESQUEMÁTICO DE CONEXIONES



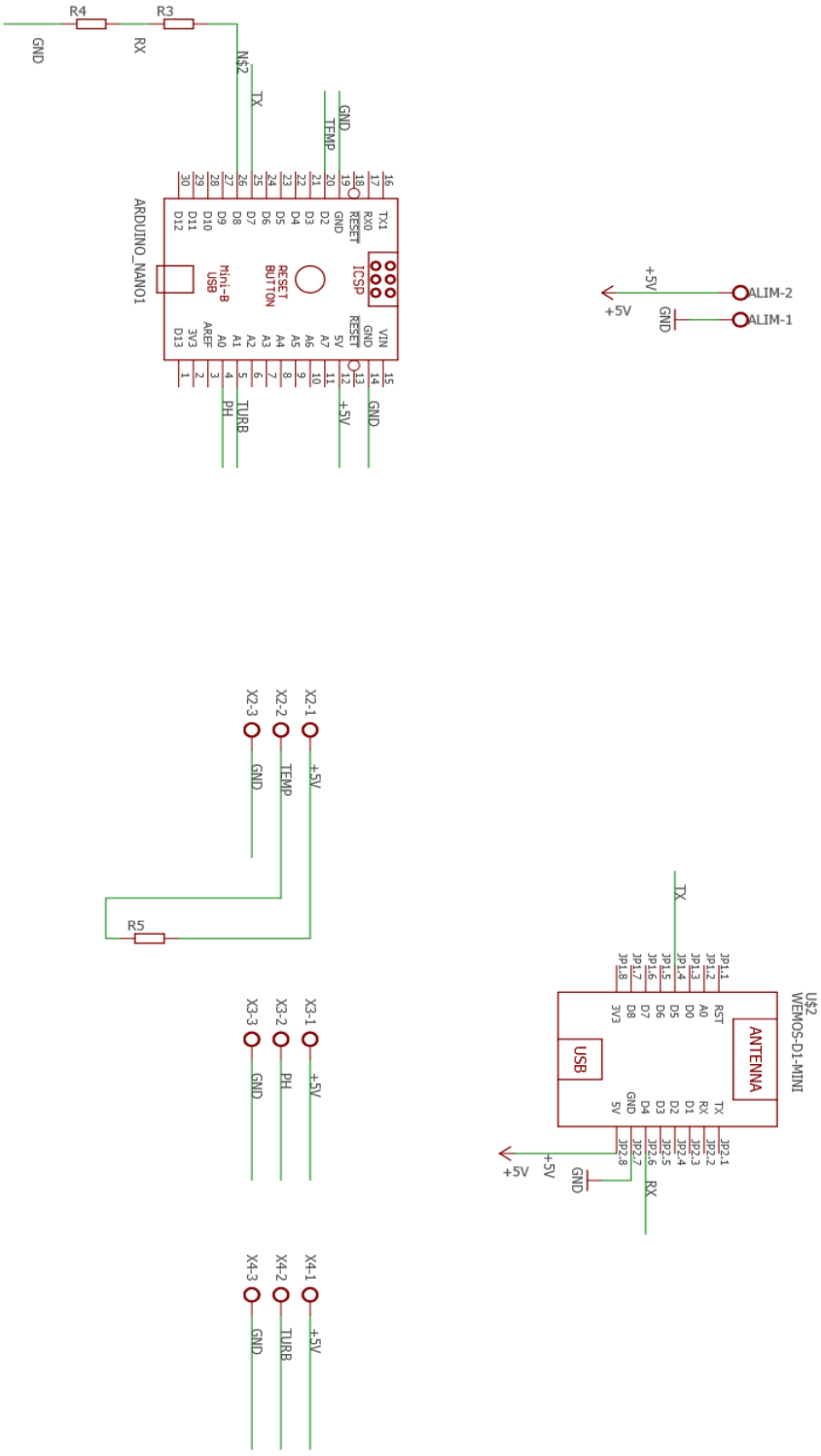
Anexo 3.1- Esquemático de conexiones del display de siete segmentos de diodos led


LED2
LED3
LED4
LED5
LED6
LED7
LED8
LED10
LED11
LED12
LED13
LED14
LED15
LED16
LED18
LED19
LED20
LED21
LED22
LED23
LED24
LED26
LED27
LED28
LED29
LED30
LED31
LED32
LED34
LED35
LED36
LED37
LED38
LED39
LED40
LED42
LED43
LED44
LED45
LED46
LED47
LED48
LED50
LED51
LED52
LED53
LED54
LED55
LED56
LED58
LED59
LED60
LED61
LED62
LED63
LED64
G
F
E
D
C
B
A
1
2
3
4
5
6
7
8
VCC
R1
GND
SCK
DATA
DIN
DOUT
LOAD
CLK
ISET
IC1
MAX7219CNG
IMP
OUT
DOUT
SCK

Anexo 3.2- Esquemático de conexiones de matriz 8x7 de diodos led.



Anexo 3.3- Esquemático de conexiones de la placa de recepción y envío de la señal de los paneles de toque



Anexo 3.4- Esquemático de conexiones de la placa de sensores del estado físico químico del agua.

ANEXO 4

DESARROLLO DEL CÓDIGO PARA LA RUTA DE LA INTERFAZ WEB DESTINADA A LA CREACIÓN DE COMPETENCIA

```
<template>
  <div class="columns">
    <div class="column is-half is-offset-one-quarter">
      <h1 class="title">Crear competencia</h1>
      <b-field label="Nombre de la competencia">
        <b-input v-
model="competencia.nombre" placeholder=" Escribir Nombre ..."></b-
input>
      </b-field>

      <b-field label="Estilo">
        <b-select v-
model="competencia.id_estilo" placeholder="Seleccione el estilo">
          <option
            v-for="estilo in estilos"
            :value="estilo.id"
            :key="estilo.id"
          >{{ estilo.nombre + ' ' + estilo.valor }}</option>
        </b-select>
      </b-field>
      <b-field label="Seleccionar fecha y hora">
        <b-datetimepicker
          v-model="competencia.fecha"
          rounded
          placeholder="Click para seleccionar ..."
          icon="calendar-today"
          :datepicker="{ showWeekNumber }"
          :timepicker="{ enableSeconds, hourFormat: format }"
        ></b-datetimepicker>
      </b-field>
      <div class="buttons">
        <b-button type="is-
dark" @click="registrar" expanded>Registrar</b-button>
        <b-button type="is-
link" @click="ver_historial" expanded>Ver historial</b-button>
        <b-button type="is-
danger" @click="borrar" expanded>Borrar datos en pantalla</b-button>
        <b-button type="is-
dark" @click="quitar_energia" expanded>Quitar energia</b-button>
        <b-button type="is-
danger" @click="apagar" expanded>Apagar Sistema</b-button>
      </div>
    </div>
  </div>
</template>
```

```
<script>
import { mapState, mapActions } from 'vuex'

export default {
  data() {
    return {
      competencia: {
        nombre: '',
        id_estilo: '',
        fecha: ''
      }
    }
  },
  computed: {
    ...mapState(['estilos'])
  },
  methods: {
    ...mapActions([
      'registrar_competencia',
      'conseguir_estilos',
      'verificar_competencia',
      'borrar_nadadores',
      'verificar_inicio_competencia',
      'apagar_raspberry',
      'cortar_energia'
    ]),
    registrar() {
      this.competencia.fecha = String(this.competencia.fecha)
      console.log(this.competencia.fecha)
      if (this.competencia.nombre === '') {
        this.$buefy.toast.open({
          message: 'Debe colocar un nombre',
          type: 'is-danger'
        })
        return
      }
      if (this.competencia.nombre.length < 5) {
        this.$buefy.toast.open({
          message: 'En nombre debe contener por lo menos 5 carácteres',
          type: 'is-danger'
        })
        return
      }
      if (this.competencia.id_estilo === '') {
        this.$buefy.toast.open({
          message: 'Debe seleccionar un estilo',
          type: 'is-danger'
```

```
        })
        return
      }
      if (this.competencia.fecha === '') {
        this.$buefy.toast.open({
          message: 'Debe colocar una fecha',
          type: 'is-danger'
        })
        return
      }
      console.log('registrar')
      console.log(this.competencia)
      this.registrar_competencia(this.competencia).then(() => {
        this.$router.push('/nadadores')
      })
    },
    ver_historial() {
      this.$router.push('/historial')
    },
    borrar() {
      this.$swal({
        title: '¿Estas seguro de borrar pantalla?',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonColor: '#3085d6',
        cancelButtonColor: '#d33',
        confirmButtonText: 'Si',
        cancelButtonText: 'Cancelar'
      }).then(result => {
        if (result.value) {
          this.borrar_nadadores()
        }
      })
    },
    quitar_energia() {
      this.$swal({
        title: '¿Estas seguro de quitar energia en placas?',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonColor: '#3085d6',
        cancelButtonColor: '#d33',
        confirmButtonText: 'Si',
        cancelButtonText: 'Cancelar'
      }).then(result => {
        if (result.value) {
          this.cortar_energia()
        }
      })
```

```
    },
    apagar() {
      this.$swal({
        title: '¿Estas seguro de apagar el equipo?',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonColor: '#3085d6',
        cancelButtonColor: '#d33',
        confirmButtonText: 'Si',
        cancelButtonText: 'Cancelar'
      }).then(result => {
        if (result.value) {
          this.apagar_raspberry()
        }
      })
    }
  },
  async created() {
    await this.verificar_competencia().then(result => {
      if (result.activo == true) {
        console.log('Ir a nadadores')
        this.$router.push('/nadadores')
      }
    })
    await this.verificar_inicio_competencia().then(result => {
      if (result.flag_competencia == true) {
        this.$router.push('/competencia')
      }
    })
    await this.conseguir_estilos()
  }
}
</script>
```

ANEXO 5

DESARROLLO DEL CÓDIGO PARA LA RUTA DESTINADA AL REGISTRO DE NADADORES EN LA INTERFAZ WEB

```
<template>
  <div>
    <div v-show="flag_registro" class="columns">
      <div class="column is-half is-offset-one-quarter">
        <h1 class="title">Registro de nadadores</h1>
        <b-field label="Nombre">
          <b-input v-model="nadador.nombre"></b-input>
        </b-field>
        <b-field label="Apellido">
          <b-input v-model="nadador.apellido"></b-input>
        </b-field>
        <b-field label="Edad">
          <b-select v-
model="nadador.edad" placeholder="Seleccione la edad">
            <option v-
for="option in edad_list" :value="option" :key="option">{{ option }}</o
ption>
          </b-select>
        </b-field>
        <b-field label="Semestre">
          <b-select v-
model="nadador.semestre" placeholder="Seleccione el semestre">
            <option v-
for="option in semestre_list" :value="option" :key="option">{{ option }
}</option>
          </b-select>
        </b-field>

        <b-field label="Carrera">
          <b-select v-
model="nadador.id_carrera" placeholder="Seleccione la carrera universit
aria">
            <option
              v-for="carrera in carreras"
              :value="carrera.id"
              :key="carrera.id"
            >{{ carrera.nombre }}</option>
          </b-select>
        </b-field>

        <b-field label="Carril">
          <b-select v-
model="nadador.carril" placeholder="Seleccione el carril">
            <option v-
for="option in carril_list" :value="option" :key="option">{{ option }}<
/option>
          </b-select>
        </b-field>
```

```
        <div class="buttons">
          <b-button type="is-
dark" expanded @click="agregar_nadador">Agregar nadador</b-button>
        </div>
      </div>
    </div>
    <div class="columns">
      <div class="column is-10 is-offset-1">
        <!-- <b-table :data="nadadores" :columns="columns"></b-table> -
->
        <b-table :data="nadadores">
          <template scope="props">
            <b-table-
column field="id" label="ID" width="40" sortable numeric>{{ props.row.i
d }}</b-table-column>

            <b-table-
column field="nombre" label="Nombre" sortable>{{ props.row.nombre }}</b
-table-column>

            <b-table-
column field="apellido" label="Apellido" sortable>{{ props.row.apellido
 }}</b-table-column>

            <b-table-
column field="edad" label="Edad" sortable>{{ props.row.edad }}</b-
table-column>

            <b-table-
column field="semestre" label="Semestre" sortable>{{ props.row.semestre
 }}</b-table-column>

            <b-table-
column field="carrera" label="Carrera" sortable>{{ props.row.carrera }}
</b-table-column>

            <b-table-
column field="carril" label="Carril" sortable>{{ props.row.carril }}</b
-table-column>
            <b-table-column field="accion" label="Acción">
              <div class="buttons">
                <b-button
                  type="is-danger"
                  icon-left="delete"
                  @click="borrar_nadador(props.row)"
                >Borrar</b-button>
              </div>
            </b-table-column>
```

```
          </template>
        </b-table>
        <div class="buttons" v-show="nadadores.length > 0">
          <b-button type="is-
link" expanded @click="iniciar_carrera">Iniciar carrera</b-button>
          <b-button type="is-
success" expanded @click="subir_nadadores">Enviar datos a la pantalla</
b-button>
          <b-button type="is-
danger" expanded @click="borrar">Borrar datos en pantalla</b-button>
          <b-button type="is-
dark" @click="quitar_energia" expanded>Quitar energia</b-button>
        </div>
      </div>
    </div>
  </div>
</template>
<script>
import { mapState, mapActions, mapMutations } from 'vuex'
export default {
  data() {
    return {
      edad_list: [],
      semestre_list: [],
      carril_list: [],
      nadador: {
        nombre: '',
        apellido: '',
        edad: '',
        semestre: '',
        id_carrera: '',
        carril: ''
      },
      flag_registro: true
    }
  },
  computed: {
    ...mapState(['carreras', 'nadadores'])
  },
  methods: {
    ...mapActions([
      'conseguir_carreras',
      'verificar_competencia',
      'registrar_nadador',
      'conseguir_nadadores',
      'eliminar_nadador',
      'iniciar_competencia',
      'enviar_nadadores',
```

```
      'borrar_nadadores',
      'verificar_inicio_competencia',
      'cortar_energia'
    ]),
    ...mapMutations(['guardar_competencia']),
    async agregar_nadador() {
      if (this.nadador.nombre === '') {
        this.$buefy.toast.open({
          message: 'Debe colocar un nombre',
          type: 'is-danger'
        })
        return
      }
      if (this.nadador.apellido === '') {
        this.$buefy.toast.open({
          message: 'Debe colocar un apellido',
          type: 'is-danger'
        })
        return
      }
      if (this.nadador.edad === '') {
        this.$buefy.toast.open({
          message: 'Debe seleccionar una edad',
          type: 'is-danger'
        })
        return
      }
      if (this.nadador.semestre === '') {
        this.$buefy.toast.open({
          message: 'Debe seleccionar un semestre',
          type: 'is-danger'
        })
        return
      }
      if (this.nadador.id_carrera === '') {
        this.$buefy.toast.open({
          message: 'Debe seleccionar una carrera',
          type: 'is-danger'
        })
        return
      }
      if (this.nadador.carril === '') {
        this.$buefy.toast.open({
          message: 'Debe seleccionar un carril',
          type: 'is-danger'
        })
        return
      }
```

```
      console.log('registrar')
      console.log(this.nadador)
      await this.registrar_nadador(this.nadador)
      await this.conseguir_nadadores()
      this.generar_carriles()
      this.nadador = {
        nombre: '',
        apellido: '',
        edad: '',
        semestre: '',
        id_carrera: '',
        carril: ''
      }
    },
        generar_carriles() {
      let carriles_activos = []
      this.nadadores.forEach(nadador => {
        carriles_activos.push(nadador.carril)
      })

      this.carril_list = []
      for (var i = 1; i <= 5; i++) {
        if (carriles_activos.indexOf(i) == -1) {
          this.carril_list.push(i)
        }
      }
      if (this.carril_list.length == 0) {
        this.flag_registro = false
      } else {
        this.flag_registro = true
      }
    },
    quitar_energia() {
      this.$swal({
        title: '¿Estas seguro de quitar energia en placas?',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonColor: '#3085d6',
        cancelButtonColor: '#d33',
        confirmButtonText: 'Si',
        cancelButtonText: 'Cancelar'
      }).then(result => {
        if (result.value) {
          this.cortar_energia()
        }
      })
    },
    async iniciar_carrera() {
```

```
      await this.conseguir_nadadores()
      if (this.nadadores.length === 1) {
        this.$buefy.toast.open({
          message: 'No se permite iniciar la competencia con un solo na
dador',
          type: 'is-danger'
        })
        return
      }
      if (this.nadadores.length > 1) {
        this.$swal({
          title:
            '¿Estas seguro de iniciar la competencia con ' +
            this.nadadores.length +
            ' nadadores?',
          icon: 'warning',
          showCancelButton: true,
          confirmButtonColor: '#3085d6',
          cancelButtonColor: '#d33',
          confirmButtonText: 'Si',
          cancelButtonText: 'Cancelar'
        }).then(async result => {
          if (result.value) {
            //await this.enviar_nadadores()
            await this.iniciar_competencia()
            this.$router.push('/competencia')
          }
        })
      }
    },
    async subir_nadadores() {
      await this.enviar_nadadores()
    },
    async borrar() {
      this.$swal({
        title: '¿Estas seguro de borrar pantalla?',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonColor: '#3085d6',
        cancelButtonColor: '#d33',
        confirmButtonText: 'Si',
        cancelButtonText: 'Cancelar'
      }).then(result => {
        if (result.value) {
          this.borrar_nadadores()
        }
      })
    },
```

```
    async borrar_nadador(nadador) {
      this.$swal({
        title: '¿Estas seguro de borrar el nadador?',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonColor: '#3085d6',
        cancelButtonColor: '#d33',
        confirmButtonText: 'Si',
        cancelButtonText: 'Cancelar'
      }).then(async result => {
        if (result.value) {
          await this.eliminar_nadador(nadador)
          await this.conseguir_nadadores()
          await this.generar_carriles()
        }
      })
    }
  },
  async created() {
    this.flag_registro = true
    await this.verificar_competencia().then(result => {
      if (result.activo == false) {
        this.$router.push('/')
      } else {
        console.log(result)
        this.guardar_competencia(result.competencia)
      }
    })
    await this.verificar_inicio_competencia().then(result => {
      if (result.flag_competencia == true) {
        this.$router.push('/competencia')
      }
    })
    await this.conseguir_carreras()
    await this.conseguir_nadadores()
    this.generar_carriles()

    this.edad_list = []
    for (let i = 13; i <= 30; i++) {
      this.edad_list.push(i)
    }
    this.semestre_list = []
    for (let i = 1; i <= 10; i++) {
      this.semestre_list.push(i)
    }
  }
}
</script>
```

ANEXO 6

DESARROLLO DEL CÓDIGO PARA LA RUTA DE LA COMPETENCIA EN CURSO DE LA INTERFAZ WEB

```
<template>
  <div class="columns">
    <div class="column is-10 is-offset-1">
      <h1 class="title">Competencia</h1>
      <b-table :data="nadadores_tiempo">
        <template scope="props">
          <b-table-
column field="id" label="Posición" width="40" sortable numeric>{{ props
.row.id }}</b-table-column>

          <b-table-column
            field="nombres"
            label="Nombres"
            sortable
          >{{ props.row.nombre + ' ' + props.row.apellido }}</b-table-
column>
          <b-table-
column field="carril" label="Carril" sortable>{{ props.row.carril }}</b
-table-column>
          <b-table-
column field="tiempo" label="Tiempo" sortable>{{ props.row.tiempo }}</b
-table-column>
        </template>
      </b-table>
      <div class="buttons">
        <b-button type="is-
link" expanded @click="terminar">Terminar</b-button>
      </div>
    </div>
  </div>
</template>

<script>
import { mapState, mapActions } from 'vuex'
export default {
  data() {
    return {}
  },
  computed: {
    ...mapState(['nadadores_tiempo'])
  },
  methods: {
    ...mapActions([
      'conseguir_nadadores_tiempo',
      'terminar_competencia',
      'verificar_inicio_competencia'
    ]),
    terminar() {
```

```
      this.$swal({
        title: '¿Estas seguro de terminar la competencia?',
        text:
          'Recordar que los nadadores que no han llegado quedaran con e
l tiempo de cuando se termine la competencia',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonColor: '#3085d6',
        cancelButtonColor: '#d33',
        confirmButtonText: 'Si',
        cancelButtonText: 'Cancelar'
      }).then(async result => {
        if (result.value) {
          await this.terminar_competencia().then(() => {
            this.$router.push('/')
          })
        }
      })
    }
  },
  async created() {
    await this.verificar_inicio_competencia().then(result => {
      if (result.flag_competencia == false) {
        this.$router.push('/nadadores')
      }
    })
    await this.conseguir_nadadores_tiempo()
    setInterval(async () => {
      await this.conseguir_nadadores_tiempo()
    }, 3000)
  }
}
</script>
```

ANEXO 7

DESARROLLO DEL CÓDIGO PARA LA RUTA DEL HISTORIAL DE COMPETENCIAS DE LA INTERFAZ WEB

```
<template>
  <div class="columns">
    <div class="column is-10 is-offset-1">
      <h1 class="title">Historial</h1>
      <!-- <b-table :data="competencias" :columns="columns"></b-
table> -->
      <b-table :data="competencias">
        <template scope="props">
          <b-table-
column field="id" label="ID" width="40" sortable numeric>
            {{
            props.row.id
            }}
          </b-table-column>

          <b-table-column field="fecha" label="Fecha" sortable>
            {{
            props.row.fecha
            }}
          </b-table-column>

          <b-table-column field="nombre" label="Nombre" sortable>
            {{
            props.row.nombre
            }}
          </b-table-column>

          <b-table-column field="estilo" label="Estilo" sortable>
            {{
            props.row.edad
            }}
          </b-table-column>

          <b-table-column field="accion" label="Acción">
            <div class="buttons">
              <b-button type="is-
primary" @click="ver_competencia(props.row)">Ver</b-button>
            </div>
          </b-table-column>
        </template>
      </b-table>
    </div>
  </div>
</template>

<script>
import { mapState, mapActions } from 'vuex'
```

```
export default {
  computed: {
    ...mapState(['competencias'])
  },
  methods: {
    ...mapActions(['conseguir_competencias', 'conseguir_nadadores_histo
rial']),
    ver_competencia(competencia) {
      this.conseguir_nadadores_historial(competencia)
      this.$router.push('/historial/nadadores')
    }
  },
  created() {
    this.conseguir_competencias()
  }
}
</script>
```

Anexo 7.1- Historial de competencias

```
<template>
  <div class="columns">
    <div class="column is-10 is-offset-1">
      <h1 class="title">Competencia - {{ competencia.estilo }}</h1>
      <b-table :data="nadadores_historial">
        <template scope="props">
          <b-table-
column field="posicion" label="Posición" width="40" sortable numeric>
            {{
            props.row.posicion
            }}
          </b-table-column>

          <b-table-column field="nombre" label="Nombre" sortable>
            {{
            props.row.nombre
            }}
          </b-table-column>

          <b-table-column field="apellido" label="Apellido" sortable>
            {{
            props.row.apellido
            }}
          </b-table-column>

          <b-table-column field="edad" label="Edad" sortable>
            {{
            props.row.edad
            }}
          </b-table-column>

          <b-table-column field="semestre" label="Semestre" sortable>
            {{
            props.row.semestre
            }}
          </b-table-column>

          <b-table-column field="carrera" label="Carrera" sortable>
            {{
            props.row.carrera
            }}
          </b-table-column>

          <b-table-column field="carril" label="Carril" sortable>
            {{
            props.row.carril
            }}
          </b-table-column>
```

```
          <b-table-column field="tiempo" label="Tiempo" sortable>
            {{
            props.row.tiempo
            }}
          </b-table-column>
        </template>
      </b-table>
      <div class="buttons">
        <b-button type="is-
success" expanded @click="subir_nadadores">Enviar datos a la pantalla</
b-button>
        <b-button type="is-
danger" expanded @click="borrar">Borrar datos en pantalla</b-button>
        <b-button type="is-
dark" @click="quitar_energia" expanded>Quitar energia</b-button>
      </div>
    </div>
  </div>
</template>

<script>
import { mapState, mapActions } from 'vuex'

export default {
  computed: {
    ...mapState(['nadadores_historial', 'competencia'])
  },
  methods: {
    ...mapActions([
      'borrar_nadadores',
      'cortar_energia',
      'mostrar_nadadores_historial'
    ]),
    quitar_energia() {
      this.$swal({
        title: '¿Estas seguro de quitar energia en placas?',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonColor: '#3085d6',
        cancelButtonColor: '#d33',
        confirmButtonText: 'Si',
        cancelButtonText: 'Cancelar'
      }).then(result => {
        if (result.value) {
          this.cortar_energia()
        }
      })
    },
```

```
    async subir_nadadores() {
      await this.mostrar_nadadores_historial()
    },
    async borrar() {
      this.$swal({
        title: '¿Estas seguro de borrar pantalla?',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonColor: '#3085d6',
        cancelButtonColor: '#d33',
        confirmButtonText: 'Si',
        cancelButtonText: 'Cancelar'
      }).then(result => {
        if (result.value) {
          this.borrar_nadadores()
        }
      })
    }
  }
}
</script>
```

Anexo 7.2- Interfaz para mostrar el historial de competidores

ANEXO 8

DESARROLLO DEL CÓDIGO EN ARDUINO PARA PANELES DE TOQUE DE LLEGADA DE COMPETIDOR

```
#include <ESP8266WiFi.h>  // Conectar wifi
#include <ESP8266HTTPClient.h> // Realizar peticiones al API( GET )

//SSID de la red LAN interna
char ssid[] = "SALESIANA2020"; //SSID of your Wi-Fi router
char pass[] = "salesiana2020"; //Password of your Wi-Fi router

int sensorPin = A0;
int sensorValue = 0;
int period = 1000; // 1 segundo = 1000 milisegundos
unsigned long time_now = 0;

void setup()
{
  Serial.begin(115200);
  delay(10);

  // Connect to Wi-Fi network
  Serial.println();
  Serial.print("Conectando a ...");
  Serial.println(ssid);
  pinMode(sensorPin, INPUT);
  pinMode(2, OUTPUT);
  for(int x=0; x<=20; x++){
    digitalWrite(2, HIGH);
    delay(50);
    digitalWrite(2, LOW);
    delay(50);
  }
  WiFi.begin(ssid, pass);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("Wi-Fi conexion exitosa");
  for(int x=0; x<=20; x++){
    digitalWrite(2, HIGH);
    delay(50);
    digitalWrite(2, LOW);
    delay(50);
  }

}
void loop () {
  sensorValue = analogRead(sensorPin);
```

```
  if (sensorValue > 200) { // cambiar sensibilidad
    Serial.println("Sensor activado");
    bool flag = true;
    //for (int i=0; i<=50; i++){
    while(flag == true) {
      if (WiFi.status() == WL_CONNECTED) { //Check WiFi connection status
        HTTPClient http;  //Declare an object of class HTTPClient
        http.begin("http://192.168.20.100:5000/competencias/nadadores/3/detener");
//Specify request destination
        int httpCode = http.GET(); //Send the request
        if (httpCode > 0) { //Check the returning code
          String payload = http.getString();   //Get the request response payload
          Serial.println(payload);                     //Print the response payload
        }
        http.end();   //Close connection
        flag = false;
      }
      delay(20);
    }
    delay(1000);
    Serial.println("Termino sensor");
  }

  while(millis() > time_now + period){
    Serial.println("Envio de peticion");
    time_now = millis();
    if (WiFi.status() == WL_CONNECTED) { //Check WiFi connection status
      HTTPClient http;  //Declare an object of class HTTPClient
      http.begin("http://192.168.20.100:5000/competencias/nadadores/6/detener");
//Specify request destination
      int httpCode = http.GET(); //Send the request
      if (httpCode > 0) { //Check the returning code
        String payload = http.getString();   //Get the request response payload
        Serial.println(payload);                     //Print the response payload
      }
      http.end();   //Close connection
    }

  }
}
```

ANEXO 9

DESARROLLO DEL CÓDIGO EN ARDUINO-WEMOS DE LOS SENSORES DEL ESTADO FÍSICO QUÍMICO DEL AGUA.

```
#include <SoftwareSerial.h>  // libreria para comunicacion serial
#include <ESP8266HTTPClient.h>
#include <ESP8266WiFi.h>

SoftwareSerial fromArduino(2, 14); //Rx, Tx  // setiAr pines diferentes a lo
preestablecidos
const char* ssid = "SALESIANA2020";
const char* password = "salesiana2020";

String inputString = "";
bool stringComplete = false;

void setup()
{
  Serial.begin(115200);
  fromArduino.begin(9600);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {

    delay(1000);
    Serial.println("Connecting..");

  }
  Serial.println("Connected");
  inputString.reserve(200);
}

void loop()
{
  if (stringComplete) {
    Serial.println(inputString);

    if (WiFi.status() == WL_CONNECTED) { //Check WiFi connection status

      HTTPClient http;  //Declare an object of class HTTPClient


http.begin(String("http://192.168.20.100:5000/sensores/registrar?sensores="+inpu
tString));  //Specify request destination
      int httpCode = http.GET();                                            //Send the
request

      if (httpCode > 0) { //Check the returning code

        String payload = http.getString();   //Get the request response payload
        Serial.println(payload);                    //Print the response payload
```

```
    }

    http.end();   //Close connection

  }

  delay(2000);    //Send a request every 30 seconds

  inputString = "";
  stringComplete = false;
 }

 while (fromArduino.available() && stringComplete == false) {
  // get the new byte:
  char inChar = (char)fromArduino.read();
  // add it to the inputString:
  inputString += inChar;
  // if the incoming character is a newline, set a flag so the main loop can
  // do something about it:
  if (inChar == '\n') {
    stringComplete = true;
  }
 }


}
```

Anexo 9.1- Código de wemos mini D1

```
#include <OneWire.h>  //sensor de temp
#include <DallasTemperature.h> //sensor de temp
#include <SoftwareSerial.h>// libreria para comunicacion serial
#include <ArduinoJson.hpp> // crear objeto json
#include <ArduinoJson.h> // crear objeto json
SoftwareSerial toESP(7, 8); //Rx, Tx
const int PhInPin = A0;
const int sensorPh = 0;
//const int TurbInPin = A1;
int sensorTurb = 0;
OneWire ourWire(2);                //Se conecta el cable de datos del sensor tde
temperatura
DallasTemperature sensors(&ourWire); //Se declara una variable u objeto para
nuestro sensor
unsigned long int avgValue;

const float b;
int buf[10],temp;
int i = 0 ;
float prom = 0;
#define TdsSensorPin A1
#define VREF 5.0      // analog reference voltage(Volt) of the ADC
#define SCOUNT  30           // sum of sample point
int analogBuffer[SCOUNT];    // store the analog value in the array, read from
ADC
int analogBufferTemp[SCOUNT];
int analogBufferIndex = 0,copyIndex = 0;
float averageVoltage = 0,tdsValue = 0,temperature = 25;

void setup() {
  Serial.begin(115200);
  toESP.begin(9600);
  sensors.begin();   //Se inicia el sensor
  pinMode(TdsSensorPin,INPUT);
}

void loop() {
  for(i=0;i<10;i++)
  {
  buf[i] = analogRead(PhInPin);
  delay(10);
  }
  for(i=0;i<9;i++)
  {
  for(int j=i+1;j<10;j++)
  {
   if(buf[i]>buf[j])
```

```
   {
    temp=buf[i];
    buf[i]=buf[j];
    buf[j]=temp;
   }
  }
  }
  avgValue=0;
  for(i=2;i<8;i++)
  avgValue+=buf[i];
  float pHVol=(float)avgValue*4.65/1024/6;
  float phValue = -5.70 * pHVol + 21.34;

    static unsigned long analogSampleTimepoint = millis();
   if(millis()-analogSampleTimepoint > 40U)     //every 40 milliseconds,read the
analog value from the ADC
   {
     analogSampleTimepoint = millis();
     analogBuffer[analogBufferIndex] = analogRead(TdsSensorPin);    //read the
analog value and store into the buffer
     analogBufferIndex++;
     if(analogBufferIndex == SCOUNT)
         analogBufferIndex = 0;
   }
   static unsigned long printTimepoint = millis();
   if(millis()-printTimepoint > 800U)
   {
      printTimepoint = millis();
      for(copyIndex=0;copyIndex<SCOUNT;copyIndex++)
        analogBufferTemp[copyIndex]= analogBuffer[copyIndex];
      averageVoltage = getMedianNum(analogBufferTemp,SCOUNT) * (float)VREF
/ 1024.0; // read the analog value more stable by the median filtering algorithm,
and convert to voltage value
      float compensationCoefficient=1.0+0.02*(temperature-25.0);    //temperature
compensation formula: fFinalResult(25^C) = fFinalResult(current)/(1.0+0.02*(fTP-
25.0));
      float compensationVolatge=averageVoltage/compensationCoefficient;
//temperature compensation

tdsValue=(133.42*compensationVolatge*compensationVolatge*compensationVol
atge - 255.86*compensationVolatge*compensationVolatge +
857.39*compensationVolatge)*0.5; //convert voltage value to tds value

   }

    sensors.requestTemperatures();   //Se envía el comando para leer la
temperatura
```

```
  float temp= sensors.getTempCByIndex(0); //Se obtiene la temperatura en ºC
  String json;
    StaticJsonDocument<300> doc;
    doc["temp"] = temp;
    doc["ph"] = phValue;
    doc["turb"] = tdsValue ;

    serializeJson(doc, json);
    toESP.println(json);
    Serial.println(json);

}

int getMedianNum(int bArray[], int iFilterLen)
{
      int bTab[iFilterLen];
      for (byte i = 0; i<iFilterLen; i++)
      bTab[i] = bArray[i];
      int i, j, bTemp;
      for (j = 0; j < iFilterLen - 1; j++)
      {
      for (i = 0; i < iFilterLen - j - 1; i++)
          {
        if (bTab[i] > bTab[i + 1])
            {
        bTemp = bTab[i];
            bTab[i] = bTab[i + 1];
        bTab[i + 1] = bTemp;
         }
      }
      }
      if ((iFilterLen & 1) > 0)
    bTemp = bTab[(iFilterLen - 1) / 2];
      else
    bTemp = (bTab[iFilterLen / 2] + bTab[iFilterLen / 2 - 1]) / 2;
      return bTemp;
}
```

Anexo 9.2- Código en arduino nano.

ANEXO 10

DESARROLLO DE CÓDIGO DE ARDUINO NANO PARA ALARMA Y RESET DEL SISTEMA

```
#include <Wire.h>

char accion[20];
char val[20];
int apagar = 10;
int rele = 9;
int resetear = 8;

void setup(){
  Serial.begin(9600);
  pinMode(rele, OUTPUT);
  pinMode(13, OUTPUT);
  pinMode(resetear, OUTPUT);
  pinMode(apagar, OUTPUT);
  digitalWrite(rele, HIGH);
  digitalWrite(resetear, HIGH);
  digitalWrite(apagar, HIGH);
  digitalWrite(13, HIGH);
  Wire.begin(7);                // join i2c bus with address #8
  Wire.onReceive(receiveEvent); // register event
}

void loop(){
  if (val[0] == '0' && val[1] == '0' && val[2] == '0') {
    Serial.println("Parlante");
    digitalWrite(rele, LOW);
    delay(100);
    digitalWrite(rele, HIGH);
    val[0] = ' ';
    val[1] = ' ';
    val[2] = ' ';
  }
  if (val[0] == '0' && val[1] == '0' && val[2] == '1') {
    Serial.println("Reset");
    digitalWrite(resetear, LOW);
    val[0] = ' ';
    val[1] = ' ';
    val[2] = ' ';
    delay(500);
    digitalWrite(resetear, HIGH);
  }
  if (val[0] == '0' && val[1] == '0' && val[2] == '2') {
    Serial.println("Cortar energia");
    digitalWrite(apagar, LOW);
    delay(1000);
    digitalWrite(apagar, HIGH);
    val[0] = ' ';
```

```
    val[1] = ' ';
    val[2] = ' ';
  }
}

void receiveEvent(int howMany) {
  Serial.println("Llego");
  Serial.println(howMany);
  for (int i = 0; i < howMany; i++){
    accion[i] = Wire.read();
    accion[i + 1] = '\0';
  }

  for (int i = 0; i < howMany; ++i){
    val[i] = accion[i + 1];
    //temp[i] = temp[i + 1];
  }
  Serial.println(val);
}
```

ANEXO 11

DESARROLLO DE CÓDIGO DE ARDUINO NANO PARA LAS PLACAS DE DIODOS DE 7 SEGMENTOS

```
#define LATCH 4
#define CLK 3
#define DATA 2
#include <Wire.h>  // libreria comunicacio i2c

byte digitTwo[11]= {0xEE, 0x42, 0xDC, 0xD6, 0x72, 0xB6, 0xBE, 0xE2, 0xFE,
0xF6, 0x00};
                  //0123456789null
char numero[7];

int numero_mostrar[6];

void setup(){
  Serial.begin(9600);
  pinMode(LATCH, OUTPUT);
  pinMode(CLK, OUTPUT);
  pinMode(DATA, OUTPUT);

  for (int i = 0; i < 6; i++){
    numero[i] = ':';
    numero[i + 1] = '\0';
  }
  for(int i = 0; i<6; i++){
    Serial.println(int(numero[i]-48));
    numero_mostrar[i] = int(numero[i]-48);
  }
  Serial.println(numero);

  Wire.begin(6);                // join i2c bus with address #8
  Wire.onReceive(receiveEvent); // register event
}


void loop(){
  // muestra el dato
    digitalWrite(LATCH, LOW);
    for (int i=5; i>=0; i--) {
      //Serial.println(numero_mostrar[i]);
      shiftOut(DATA, CLK, MSBFIRST, ~digitTwo[numero_mostrar[i]]); // digitTwo
    }
    digitalWrite(LATCH, HIGH);
}

void receiveEvent(int howMany)
{
  // resive el dato
```

```
for (int i = 0; i < howMany; i++){
    numero[i] = Wire.read();
    numero[i + 1] = '\0';
  }

  for(int i = 1; i<howMany; i++){
    numero_mostrar[i-1] = int(numero[i]-48);
    Serial.println(numero_mostrar[i-1]);

  }

    Serial.println("Termino");

}
```

ANEXO 12

DESARROLLO DE CÓDIGO DE LOS ARDUINOS NANO PARA LAS MATRICES DE PLACAS DE DIODOS LED

```
#include <MaxMatrix.h>
#include <avr/pgmspace.h>
#include <Wire.h>


int data = 8;    // DIN pin of MAX7219 module
int load = 9;    // CS pin of MAX7219 module
int clock = 10;  // CLK pin of MAX7219 module

int maxInUse = 5;    //change this variable to set how many MAX7219's you'll use

const byte letra[63][6] = {
  {0x3F,0x48,0x48,0x3F,0x00}, // A0
  {0x7F,0x49,0x49,0x3E,0x00}, // B1
  {0x7F,0x41,0x41,0x41,0x00}, // C2
  {0x7F,0x41,0x41,0x3E,0x00}, // D3
  {0x7F,0x49,0x49,0x41,0x00}, // E4
  {0x7F,0x48,0x48,0x40,0x00}, // F5
  {0x7F,0x41,0x49,0x4F,0x00}, // G6
  {0x7F,0x08,0x08,0x7F,0x00}, // H7
  {0x41,0x7F,0x41,0x00},//I 3 8
  {0x02,0x01,0x01,0x7E,0x00},// J 4 9
  {0x7F,0x18,0x24,0x43,0x00},//K 4 10
  {0x7F,0x01,0x01,0x01,0x00},//L 4 11
  {0x7F,0x20,0x10,0x20,0x7F,0x00},//M 5 12
  {0x7F,0x20,0x10,0x08,0x7F,0x00},//N 5 13
  {0x3E,0x41,0x41,0x41,0x3E,0x00},//O 5 14
  {0x7F,0x48,0x48,0x30,0x00},//P 4 15
  {0x3E,0x41,0x45,0x43,0x3F,0x00},//Q 5 16
  {0x7F,0x4C,0x4A,0x31,0x00},// R 4 17
  {0x31,0x49,0x49,0x46,0x00},// S 4 18
  {0x40,0x40,0x7F,0x40,0x40,0x00},// T 5 19
  {0x7E,0x01,0x01,0x01,0x7E,0x00},// U 5 20
  {0x7C,0x02,0x01,0x02,0x7C,0x00},// V 5 21
  {0x7F,0x02,0x04,0x02,0x7F,0x00},// W 5 22
  {0x63,0x14,0x08,0x14,0x63,0x00},// X 5 23

  {0x60,0x10,0x0F,0x10,0x60,0x00},// Y 5 24

  {0x43,0x45,0x49,0x51,0x61,0x00},// Z 5 25
  {0x17,0x15,0x1F,0x00},//a 3 26
  {0x3F,0x09,0x0F,0x00},//b  3 27
  {0x1F,0x11,0x11,0x00},//c  3 28
  {0x0F,0x09,0x3F,0x00},//d  3 29
  {0x1F,0x15,0x1D,0x00},//e  3 30
  {0x1F,0x28,0x28,0x00},//f  3 31
  {0x1D,0x15,0x1F,0x00},//g  3 32
```

```
  {0x3F,0x08,0x07,0x00},//h  3 33
  {0x1F,0x00},//i  1 34
  {0x02,0x01,0x1F,0x00},//j  3 35
  {0x3F,0x0C,0x13,0x00},//k  3 36
  {0x3F,0x00},//l  1 37
  {0x1F,0x10,0x1F,0x10,0x0F,0x00},// m 5 38
  {0x1F,0x10,0x0F,0x00},//n  3 39
  {0x1F,0x11,0x1F,0x00},//o  3 40
  {0x3F,0x28,0x38,0x00},//p  3 41
  {0x1C,0x14,0x1F,0x00},//q  3 42
  {0x1F,0x08,0x10,0x00},//r  3 43
  {0x09,0x15,0x12,0x00},//s  3 44
  {0x08,0x1F,0x09,0x00},//t  3 45
  {0x1F,0x01,0x1F,0x00},//u  3 46
  {0x1E,0x01,0x1E,0x00},//v  3 47
  {0x1E,0x01,0x1E,0x01,0x1E,0x00},//w 5 48
  {0x1B,0x04,0x1B,0x00},// x  3 49
  {0x18,0x07,0x18,0x00},//y  3 50
  {0x13,0x15,0x19,0x00},//z  3 51
  {0x7F,0x41,0x7F,0x00},//0  3 52
  {0x21,0x7F,0x01,0x00},//1  3 53
  {0x4F,0x49,0x79,0x00},//2  3 54
  {0x49,0x49,0x7F,0x00},//3  3 55
  {0x78,0x08,0x7F,0x00},//4  3 56
  {0x79,0x49,0x4F,0x00},//5  3 57
  {0x7F,0x49,0x4F,0x00},//6  3 58
  {0x40,0x40,0x7F,0x00},//7  3 59
  {0x40,0x40,0x7F,0x00},//8  3 60
  {0x40,0x40,0x7F,0x00},//9  3 61
  {0x00,0x00,0x00,0x00}//Espacio  3 62


};

int letra_size[63] = {
  4,4,4,4,4,4,4,4,3,4,4,4,5,5,5,4,5,4,4,5,5,5,5,5,5,5,3,
  3,3,3,3,3,3,3,1,3,3,1,5,3,3,3,3,3,3,3,3,3,5,3,3,3,3,3,3,3,3,3,3,3,3,3,3
};

char temp[32];
String command;
char texto[32] = "                    ";
char carril = ' ';
int intensidad = 7;

MaxMatrix m(data, load, clock, maxInUse); // define module
```

```
byte buffer[10];

void setup()
{
  m.init(); // module initialize
  m.setIntensity(intensidad); // dot matix intensity 0-15
  Serial.begin(9600); // serial communication initialize
  Wire.begin(8);                // join i2c bus with address #8
  Wire.onReceive(receiveEvent); // register event
}

void loop(){

  byte texto_hex[200] = {};

  int contador_texto = 0;
  for (int x=0; x<=sizeof(texto); x++) {

    int letra_int = 0;
    if (int(texto[x]) >= 65 &&  int(texto[x]) <= 90){
      letra_int = int(texto[x]) - 65;
    }
    if (int(texto[x]) >= 97 &&  int(texto[x]) <= 122){
      letra_int = int(texto[x]) - 97 + 26;
    }
    if (int(texto[x]) >= 48 &&  int(texto[x]) <= 57){
      letra_int = int(texto[x]) - 48 + 52;
    }

    if (int(texto[x]) == 32){
      letra_int = int(texto[x]) - 32 + 62;
    }


    for (int j=0; j<=letra_size[letra_int]; j++) {

      texto_hex[contador_texto] = letra[letra_int][j];
      contador_texto++;
    }

  }

  long contador = 0;

  for (int x=32; x<=39; x++) {
    contador++;
    m.setColumn(byte(x), texto_hex[x-32]);
```

```
    delay(20);
  }

  for (int x=24; x<=31; x++) {
    contador++;
    m.setColumn(byte(x), texto_hex[x-24+8]);
    delay(20);
  }

  for (int x=16; x<=23; x++) {
    contador++;
    m.setColumn(byte(x), texto_hex[x-16+8*2]);
    delay(20);
  }

  for (int x=8; x<=15; x++) {
    contador++;
    m.setColumn(byte(x), texto_hex[x-8+8*3]);
    delay(20);
  }

  for (int x=0; x<=3; x++) {
    contador++;
    m.setColumn(byte(x), texto_hex[x+8*4]);
    delay(20);
  }
  String peli;
  Serial.println(carril+4);
  if (carril == ' ') {
    m.setColumn(0x04, 0x00);
    m.setColumn(0x05, 0x00);
    m.setColumn(0x06, 0x00);
    m.setColumn(0x07, 0x00);
  } else {
    m.setColumn(0x04, 0x00);
    m.setColumn(0x05, letra[carril+4][0]);
    m.setColumn(0x06, letra[carril+4][1]);
    m.setColumn(0x07, letra[carril+4][2]);

  }

  Serial.println(texto);
}

void receiveEvent(int howMany) {

  for (int i = 0; i < howMany; i++) {
```

```
    temp[i] = Wire.read();
    temp[i + 1] = '\0'; //add null after ea. char
  }

  //RPi first byte is cmd byte so shift everything to the left 1 pos so temp contains
our string
  for (int i = 0; i < howMany; ++i){
    texto[i] = temp[i + 1];
    //temp[i] = temp[i + 1];
  }
  carril = temp[howMany-1];
  intensidad = int(temp[howMany-2]);
  m.setIntensity(intensidad);

}
```

Anexo 12.1**-** Código de letras de nombres y carril.

```
#include <MaxMatrix.h> // libreria neceseria para el inegrado o generar la matrix
#include <avr/pgmspace.h>
#include <Wire.h>



int data = 8;    // DIN pin of MAX7219 module
int load = 9;    // CS pin of MAX7219 module
int clock = 10;  // CLK pin of MAX7219 module

int maxInUse = 5;    //change this variable to set how many MAX7219's you'll use

//abecedario y numeros
const byte letra[63][6] = {
  {0x3F,0x48,0x48,0x3F,0x00}, // A0
  {0x7F,0x49,0x49,0x3E,0x00}, // B1
  {0x7F,0x41,0x41,0x41,0x00}, // C2
  {0x7F,0x41,0x41,0x3E,0x00}, // D3
  {0x7F,0x49,0x49,0x41,0x00}, // E4
  {0x7F,0x48,0x48,0x40,0x00}, // F5
  {0x7F,0x41,0x49,0x4F,0x00}, // G6
  {0x7F,0x08,0x08,0x7F,0x00}, // H7
  {0x41,0x7F,0x41,0x00},//I 3 8
  {0x02,0x01,0x01,0x7E,0x00},// J 4 9
  {0x7F,0x18,0x24,0x43,0x00},//K 4 10
  {0x7F,0x01,0x01,0x01,0x00},//L 4 11
  {0x7F,0x20,0x10,0x20,0x7F,0x00},//M 5 12
  {0x7F,0x20,0x10,0x08,0x7F,0x00},//N 5 13
  {0x3E,0x41,0x41,0x41,0x3E,0x00},//O 5 14
```

```
{0x7F,0x48,0x48,0x30,0x00},//P 4 15
{0x3E,0x41,0x45,0x43,0x3F,0x00},//Q 5 16
{0x7F,0x4C,0x4A,0x31,0x00},// R 4 17
{0x31,0x49,0x49,0x46,0x00},// S 4 18
{0x40,0x40,0x7F,0x40,0x40,0x00},// T 5 19
{0x7E,0x01,0x01,0x01,0x7E,0x00},// U 5 20
{0x7C,0x02,0x01,0x02,0x7C,0x00},// V 5 21
{0x7F,0x02,0x04,0x02,0x7F,0x00},// W 5 22
{0x63,0x14,0x08,0x14,0x63,0x00},// X 5 23

{0x60,0x10,0x0F,0x10,0x60,0x00},// Y 5 24

{0x43,0x45,0x49,0x51,0x61,0x00},// Z 5 25
{0x17,0x15,0x1F,0x00},//a 3 26
{0x3F,0x09,0x0F,0x00},//b  3 27
{0x1F,0x11,0x11,0x00},//c  3 28
{0x0F,0x09,0x3F,0x00},//d  3 29
{0x1F,0x15,0x1D,0x00},//e  3 30
{0x1F,0x28,0x28,0x00},//f  3 31
{0x1D,0x15,0x1F,0x00},//g  3 32
{0x3F,0x08,0x07,0x00},//h  3 33
{0x1F,0x00},//i  1 34
{0x02,0x01,0x1F,0x00},//j  3 35
{0x3F,0x0C,0x13,0x00},//k  3 36
{0x3F,0x00},//l  1 37
{0x1F,0x10,0x1F,0x10,0x0F,0x00},// m 5 38
{0x1F,0x10,0x0F,0x00},//n  3 39
{0x1F,0x11,0x1F,0x00},//o  3 40
{0x3F,0x28,0x38,0x00},//p  3 41
{0x1C,0x14,0x1F,0x00},//q  3 42
{0x1F,0x08,0x10,0x00},//r  3 43
{0x09,0x15,0x12,0x00},//s  3 44
{0x08,0x1F,0x09,0x00},//t  3 45
{0x1F,0x01,0x1F,0x00},//u  3 46
{0x1E,0x01,0x1E,0x00},//v  3 47
{0x1E,0x01,0x1E,0x01,0x1E,0x00},//w 5 48
{0x1B,0x04,0x1B,0x00},// x  3 49
{0x18,0x07,0x18,0x00},//y  3 50
{0x13,0x15,0x19,0x00},//z  3 51
{0x7F,0x41,0x7F,0x00},//0  3 52
{0x21,0x7F,0x01,0x00},//1  3 53
{0x4F,0x49,0x79,0x00},//2  3 54
{0x49,0x49,0x7F,0x00},//3  3 55
{0x78,0x08,0x7F,0x00},//4  3 56
{0x79,0x49,0x4F,0x00},//5  3 57
{0x7F,0x49,0x4F,0x00},//6  3 58
{0x40,0x40,0x7F,0x00},//7  3 59
```

```
  {0x40,0x40,0x7F,0x00},//8  3 60
  {0x40,0x40,0x7F,0x00},//9  3 61
  {0x00,0x00,0x00,0x00}//Espacio  3 62


};

int letra_size[63] = {
  4,4,4,4,4,4,4,4,3,4,4,4,5,5,5,4,5,4,4,5,5,5,5,5,5,5,3,
  3,3,3,3,3,3,3,1,3,3,1,5,3,3,3,3,3,3,3,3,3,5,3,3,3,3,3,3,3,3,3,3,3,3,3,3
};

char temp[32];
String command;
char texto[32] = "                    ";
char carril = ' ';
int intensidad = 7;

MaxMatrix m(data, load, clock, maxInUse); // define module

byte buffer[10];


void setup()
{
  m.init(); // module initialize
  m.setIntensity(intensidad); // dot matix intensity 0-15
  Serial.begin(9600); // serial communication initialize
  Wire.begin(13);               // join i2c bus with address #8
  Wire.onReceive(receiveEvent); // register event
}

void loop(){
  //delay(5000);

  byte texto_hex[200] = {};

  int contador_texto = 0;
  for (int x=0; x<=sizeof(texto); x++) {

    int letra_int = 0;
    if (int(texto[x]) >= 65 &&  int(texto[x]) <= 90){
      letra_int = int(texto[x]) - 65;
    }
    if (int(texto[x]) >= 97 &&  int(texto[x]) <= 122){
      letra_int = int(texto[x]) - 97 + 26;
    }
```

```
    if (int(texto[x]) >= 48 &&  int(texto[x]) <= 57){
      letra_int = int(texto[x]) - 48 + 52;
    }

    if (int(texto[x]) == 32){
      letra_int = int(texto[x]) - 32 + 62;
    }


    for (int j=0; j<=letra_size[letra_int]; j++) {

      texto_hex[contador_texto] = letra[letra_int][j];
      contador_texto++;
    }


  }

  long contador = 0;

  for (int x=32; x<=39; x++)
{
    contador++;
    m.setColumn(byte(x), texto_hex[x-32]);

  }

  for (int x=24; x<=31; x++) {
    contador++;
    m.setColumn(byte(x), texto_hex[x-24+8]);

  }

  for (int x=16; x<=23; x++) {
    contador++;
    m.setColumn(byte(x), texto_hex[x-16+8*2]);

  }

  for (int x=8; x<=15; x++) {
    contador++;
    m.setColumn(byte(x), texto_hex[x-8+8*3]);

 }

  for (int x=0; x<=7; x++) {
    contador++;
```

```
    m.setColumn(byte(x), texto_hex[x+8*4]);

  }
  String peli;

  Serial.println(texto);
}

void receiveEvent(int howMany) {

  for (int i = 0; i < howMany; i++) {
    temp[i] = Wire.read();
    temp[i + 1] = '\0'; //add null after ea. char
  }

  //RPi first byte is cmd byte so shift everything to the left 1 pos so temp contains
our string
  for (int i = 0; i < howMany; ++i){
    texto[i] = temp[i + 1];

  }

  carril = temp[howMany-1];
  intensidad = int(temp[howMany-2]);
  m.setIntensity(intensidad);

  }
```

Anexo 12.2- Código de letras de estilo de natación