

UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:
INGENIERÍA ELECTRÓNICA

Trabajo de titulación previo a la obtención del título de:
INGENIEROS ELECTRÓNICOS

TEMA:
DESARROLLO DE UN DISPOSITIVO IOT CON
CLOUD PARA EL CRIADERO DE TRUCHAS LA
MERCED

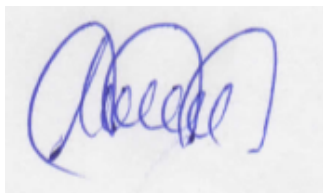
AUTORES:
CHANATAXI GUALOTUÑA CARLOS EDUARDO
PILCA SILVA HENRY HERNÁN

TUTOR:
OÑATE CADENA LUIS GERMÁN

Quito, septiembre 2021

CESIÓN DE DERECHOS DE AUTOR

Nosotros, Chanataxi Gualotuña Carlos Eduardo Y Pilca Silva Henry Hernán, con documentos de identificación N° 172173660-9 y N° 172199017-2 manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de titulación intitulado: **DESARROLLO DE UN DISPOSITIVO IOT CON CLOUD PARA EL CRIADERO DE TRUCHAS LA MERCED**, mismo que ha sido desarrollado para optar por el título de: Ingenieros Electrónicos, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente. En aplicación a lo determinado en la Ley de Propiedad Intelectual, en nuestra condición de autores nos reservamos los derechos morales de la obra antes citada. En concordancia, suscribimos este documento en el momento que hacemos entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.



Chanataxi Gualotuña Carlos Eduardo
C.I. 172173660-9



Pilca Silva Henry Hernán
C.I. 172199017-2

Quito, septiembre 2021

DECLARATORIA DE COAUTORÍA DEL DOCENTE TUTOR

Yo declaro que bajo mi dirección y asesoría fue desarrollado el Proyecto Técnico, DESARROLLO DE UN DISPOSITIVO IOT CON CLOUD PARA EL CRIADERO DE TRUCHAS LA MERCED, realizado por Chanataxi Gualotuña Carlos Eduardo Y Pilca Silva Henry Hernán, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana para ser considerado como trabajo final de titulación.

Quito, septiembre del 2021



Oñate Cadena Luis Germán

CI: 1712157401

DEDICATORIA

El presente trabajo lo dedico a mis padres por su esfuerzo y apoyo incondicional en el proceso universitario, a mis familiares que me motivaron a seguir adelante frente a los obstáculos que se presentaron en el camino. A mis amigos con los cuales compartimos momentos agradables y a Henry Pilca con quien compartimos el proceso de titulación.

Chanataxi Carlos

Este trabajo de titulación se lo dedico a mi madre que su esfuerzo y apoyo incondicional fue un pilar fundamental para poder culminar mis estudios, a mis hermanos Paulina y Sergio que con su cariño nunca me dejaron solo a ningún momento, a Diego por haber sido un padre y un ejemplo en mi vida, y de manera especial dedico este trabajo a mi hijo Ricardo que es el pilar fundamental en mi vida.

Pilca Henry

AGRADECIMIENTO

A la Universidad Politécnica Salesiana por brindarnos la oportunidad de formarnos académicamente y a su vez inculcarnos valores que nos servirán en nuestra vida profesional. De la misma manera a los docentes que formaron parte de nuestro crecimiento académico ya que, su conocimiento fue pilar fundamental en nuestra formación como profesionales.

Al Ingeniero Oñate Cadena Luis Germán, MSc. por su apertura y tiempo para solventar nuestras inquietudes y dudas en el transcurso del desarrollo del este trabajo de titulación.

Gracias.

ÍNDICE GENERAL

CESIÓN DE DERECHOS DE AUTOR.....	i
DECLARATORIA DE COAUTORÍA DEL DOCENTE TUTOR.....	ii
DEDICATORIA	iii
AGRADECIMIENTO.....	iv
ÍNDICE GENERAL.....	v
ÍNDICE DE FIGURAS.....	vii
ÍNDICE DE TABLAS	x
RESUMEN.....	xi
ABSTRACT.....	xii
INTRODUCCIÓN	xiii
CAPÍTULO 1 ANTECEDENTES	1
1.1 Planteamiento del problema	1
1.2 Justificación.....	1
1.3 Objetivos.....	2
1.3.1 Objetivo general.....	2
1.3.2 Objetivos específicos	2
1.4 Metodología.....	2
CAPÍTULO 2 FUNDAMENTACIÓN TEÓRICA.....	3
2.1. IoT	3
2.2. CLOUD	3
2.3. NODE MCU-32S	4
2.4. SENSOR DE OXÍGENO DISUELTO.....	6
2.5. SENSOR DE POTENCIAL DE HIDRÓGENO (pH).....	6
2.6. SENSOR DE TEMPERATURA	7
2.7. ThingSpeak.....	8

2.8. Thingview.....	8
2.9. IFTTT	9
2.10. WEB HOCKS.....	9
CAPÍTULO 3 DISEÑO E IMPLEMENTACIÓN	10
3.1. Parámetros establecidos para control de la calidad del agua.....	10
3.2. Diagramas de bloques del equipo.....	11
3.3. Diagramas esquemáticos	22
3.4. Lectura de datos en la plataforma ThingSpeak	24
3.5. Generar alertas por medio de ThingSpeak	25
3.6. Aplicación para el monitoreo de datos	25
3.7. Sistema de activación de una bomba por medio de Telegram.	27
CAPÍTULO 4 PRUEBAS Y RESULTADOS	29
4.1. Latencia	29
4.2 Retardos.....	34
4.3 Disponibilidad de la red.....	40
4.4 Pruebas de envío y recepción de datos entre el NodeMCU-32s y el usuario... 41	
4.5 Error y Precisión de los sensores.....	43
4.5.1 Error y precisión del sensor de pH.....	43
4.5.2 Error y precisión del sensor de temperatura	44
4.5.3 Error y precisión del sensor de oxígeno disuelto.....	44
4.6 Encuesta para verificar la factibilidad del equipo	44
CONCLUSIONES	49
RECOMENDACIONES	51
BIBLIOGRAFÍA.....	52
ANEXOS.....	54

ÍNDICE DE FIGURAS

Figura 2.1. Conexión de dispositivos IoT	3
Figura 2.2. Cloud.....	3
Figura 2.3. Microcontrolador NODE MCU ESP-32S	4
Figura 2.4. Sensor de oxígeno disuelto	6
Figura 2.5. Sensor de pH.....	7
Figura 2.6. Sensor de temperatura.....	7
Figura 2.7. Plataforma de ThingSpeak.....	8
Figura 2.8. Interfaz de ThingView	9
Figura 2.9. Trama de IFTTT	9
Figura 3.1. Piscina del criadero de truchas La Merced	10
Figura 3.2. Diagrama de bloques de la comunicación y control IoT	12
Figura 3.3. Diagrama de bloques de la comunicación de la nube con el NODE MCU-32S.....	12
Figura 3.4. Diagrama de flujo de adquisición y envío de datos del NODE MCU-32S	13
Figura 3.5. Diagrama de flujo de lectura y escalamiento de los datos de los sensores.	14
Figura 3.6. Diagrama de flujo de comunicación y envío de datos a la nube.	15
Figura 3.7. Diagrama de flujo de observación de los datos almacenados en la nube	16
Figura 3.8. Diagrama de flujo del servicio IFTTT	17
Figura 3.9. Diagrama de flujo de alerta por valores excedidos	18
Figura 3.10. Diagrama de flujo de alerta vía Telegram	19
Figura 3.11. Diagrama de flujo de la visualización de datos en la aplicación móvil.	20
Figura 3.12. Diagrama de flujo para la producción de alertas	21
Figura 3.13. Diagrama de las conexiones de los dispositivos con el NODE MCU-32S	22
Figura 3.14. Diagrama esquemático realizado para poder elaborar la placa PBC	23

Figura 3.15. Diseño e implementación de la placa PCB para su impresión en cobre	23
.....	23
Figura 3.16. Interfaz de histogramas en ThingSpeak.....	24
Figura 3.17. Evento creado en el servicio IFTTT	25
Figura 3.18. Origen del ID y API KEY	26
Figura 3.19. Confirmación de los canales a exhibirse en la aplicación de ThingView	
.....	26
Figura 3.20. Vista de históricos en ThingView.....	27
Figura 3.21. Menú para el control el del sistema en el criadero de truchas La Merced	
.....	28
Figura 4.1. Datos de latencia obtenidos a las 10:00 am	30
Figura 4.2. Datos de latencia obtenidos a las 12:00 am.....	31
Figura 4.3. Datos obtenidos a las 14:00 pm	32
Figura 4.4. Datos obtenidos a las 16:00 pm	32
Figura 4.5. Datos obtenidos a las 18:00 pm	33
Figura 4.6. Datos obtenidos a las 20:00 pm	34
Figura 4.7. Datos obtenidos a las 10:00 am	35
Figura 4.8. Datos obtenidos a las 12:00 am	36
Figura 4.9. Datos obtenidos a las 14:00 pm	37
Figura 4.10. Datos obtenidos a las 16:00 pm	38
Figura 4.11. Datos obtenidos a las 18:00 pm	39
Figura 4.12. Datos obtenidos a las 20:00 pm	40
Figura 4.13. Disponibilidad de la red.....	41
Figura 4.14. Interfaz del chat de la red social Telegram.....	42
Figura 4.15. Pregunta 1 de la encuesta.....	44
Figura 4.16. Pregunta 2 de la encuesta.....	45
Figura 4.17. Pregunta 3 de la encuesta.....	46
Figura 4.18. Pregunta 4 de la encuesta.....	46

Figura 4.19. Pregunta 5 de la encuesta.....	47
Figura 4.20. Pregunta 6 de la encuesta.....	48

ÍNDICE DE TABLAS

Tabla 2.1. Comparativa de módulos para trabajar con IoT	5
Tabla 4.1. Porcentaje de respuestas de la pregunta 1	45
Tabla 4.2. Porcentaje de respuestas de la pregunta 2	45
Tabla 4.3. Porcentaje de respuestas de la pregunta 3	46
Tabla 4.4. Porcentaje de respuestas de la pregunta 4	47
Tabla 4.5. Porcentaje de respuestas de la pregunta 5	47
Tabla 4.6. Porcentaje de respuestas de la pregunta 6	48

RESUMEN

La tasa de mortalidad en los criaderos artesanales de truchas especie arcoíris es elevada debido a que las piscinas carecen de las condiciones necesarias para monitorear los parámetros de oxígeno disuelto, PH y temperatura ya que estos influyen directamente en el crecimiento y desarrollo adecuado de los alevines. El criadero de truchas la Merced no dispone de sensores para medir, registrar y controlar en tiempo real las variables las cuales afectan de manera directa al crecimiento de las truchas sufriendo pérdidas de crecimiento y producción. Se utilizó una aplicación para smartphone y una página web en los cuales se puede inspeccionar de manera local y remota los datos adquiridos por los sensores colocados en la piscina del criadero de truchas la Merced, de la misma manera los datos obtenidos permitirán la activación de un actuador el cual controlará una electrobomba permitiendo mantener el nivel adecuado de los parámetros antes mencionados. Con el análisis de los datos se desarrolló un dispositivo IoT con Cloud el cual permite controlar los parámetros de PH, temperatura y oxígeno disuelto para la piscina del criadero de truchas La Merced, disminuyendo la tasa de mortalidad de las truchas.

ABSTRACT

The mortality rate in artisanal rainbow trout farms is high because the pools lack the necessary conditions to monitor dissolved oxygen parameters, pH and temperature as these directly influence the growth and proper development of the fry. La Merced trout farm does not have sensors to measure, record and control in real time the variables that directly affect the growth of trout suffering growth and production losses. A smartphone application and a website were used in which the data acquired by the sensors placed in the pool of the La Merced trout farm can be remotely and locally inspected, in the same way the data obtained will allow the activation of an actuator which will control an electropump allowing to maintain the appropriate level of the aforementioned parameters. With the analysis of the data an IoT device was developed with Cloud which allows to control the parameters of pH, temperature and dissolved oxygen for the pool of the trout hatchery La Merced, decreasing the mortality rate of trout.

INTRODUCCIÓN

El criadero de truchas La Merced tiene un método de crianza artesanal por lo cual carecen de un dispositivo que permita el monitoreo y control de la calidad del agua, aquello ocasiona pérdidas económicas además de problemas en el desarrollo óptimo en la crianza de truchas especie arcoíris.

El no poder discernir los niveles de pH, temperatura y oxígeno disuelto en el agua o su vez no poder monitorearla local o remotamente provoca pérdidas de producción y crecimiento de las truchas dificultando la capacidad de sacar un mayor rendimiento en la calidad de cada uno de los alevines que se encuentran en las piscinas.

En el presente proyecto técnico se desarrolló un dispositivo IoT con Cloud para el monitoreo de la calidad del agua en una de las piscinas del criadero de truchas La Merced, el monitoreo constante se lo realizó por medio de los siguientes sensores: temperatura, pH y oxígeno disuelto debido a que las truchas tienen exigencias elevadas en dichos parámetros, los sensores antes mencionados envían información a la nube por medio de conectividad WiFi gracias al dispositivo NODE MCU-32S, la falta de supervisión de estos puede provocar enfermedades o incluso la muerte de las truchas. Para poder estabilizar los parámetros establecidos para la calidad del agua se implementó un sistema de bombeo el cual será controlado por la persona encargada de supervisar la piscina, para facilidad del usuario se generó un menú en la red social Telegram para la activación y desactivación de la misma.

El presente proyecto consta de cinco capítulos y un apartado de anexos los cuales se especifican a continuación:

En el Capítulo 1 se muestra el planteamiento del problema, justificación, detalle de objetivos general y específicos y metodología a utilizar.

En el Capítulo 2 se presentan los conceptos utilizados para el desarrollo del dispositivo IoT con Cloud, así como también los sensores empleados para el monitoreo de la calidad del agua.

En el Capítulo 3 se define el diseño e implementación del dispositivo IoT y su vez el software generado para cumplir los requerimientos de monitoreo.

En el Capítulo 4 se muestran las pruebas y resultados sobre latencia, retardos y disponibilidad de la red obtenidos en la adquisición de datos, de la misma forma se indica la interacción del dispositivo con la red social Telegram. Para finalizar este capítulo se realizó una encuesta para poder verificar el grado de utilidad del dispositivo IoT.

En el capítulo 5 se indican las respectivas conclusiones y recomendaciones.

En el apartado de anexos se muestra el código de programación del dispositivo NODEMCU-32S además de la implementación del dispositivo IoT en una de las piscinas del criadero de truchas la merced.

CAPÍTULO 1

ANTECEDENTES

1.1 Planteamiento del problema

El criadero de truchas la merced carece de supervisión constante y controlada, lo que provoca que existan pérdidas en la producción, tanto económicas, como de tiempo.

El nivel de oxígeno disuelto en el agua es uno de los parámetros que más influencia tiene en la supervisión de las piscinas de truchas debido a que, el oxígeno disuelto junto con otras variables permiten un mejor desarrollo de las truchas especie arcoíris para alcanzar un tamaño esperado en un tiempo determinado y que, por el contrario su crecimiento se vea afectado por no tener los niveles adecuados de oxígeno disuelto temperatura y PH que ayudan a mantener mejores condiciones de vida en la piscina.

El PH en el agua controla el nivel de acidez que tiene y por otro lado también se puede determinar la contaminación que tenga la piscina, teniendo en cuenta estas variables es de vital importancia la inspección de estos datos en la piscina de truchas La Merced, para no tener pérdidas en su producción.

1.2 Justificación

El desarrollo de este dispositivo permitirá monitorear y controlar en tiempo real los parámetros de oxígeno disuelto, PH y temperatura en los criaderos de truchas arcoíris por medio de un sistema IOT con Cloud logrando que las truchas tengan un crecimiento adecuado y su producción se eleve con respecto a otros criaderos artesanales los cuales siguen trabajando con un método artesanal de crianza.

El dispositivo tiene como principal ventaja la adquisición de datos, los cuales proporcionarán el estado específico de la calidad del agua en el criadero de truchas La Merced, de la misma manera se pretende establecer una nueva forma de supervisión y cultivo para que las truchas se puedan alimentar bien y a su vez prevenir enfermedades que puedan afectar su crecimiento el cual está ligado directamente con la calidad de agua en las piscinas de truchas especie arcoíris.

1.3 Objetivos

1.3.1 Objetivo general

- Desarrollar un dispositivo IOT con CLOUD en un criadero de truchas de especie arcoíris para el monitoreo del oxígeno disuelto, PH y temperatura del agua.

1.3.2 Objetivos específicos

- Analizar los diferentes sensores y dispositivos para la toma de muestras del oxígeno disuelto, PH y temperatura del agua en criaderos de truchas.
- Diseñar un dispositivo IOT en cloud para el monitoreo de los parámetros del agua en el criadero de truchas La Merced.
- Implementar un dispositivo IOT en cloud para el monitoreo y la supervisión de las condiciones adecuadas de la calidad de agua de la piscina de las truchas arco iris
- Realizar las pruebas de latencia, retardo, disponibilidad del dispositivo IOT para la comprobación del correcto funcionamiento del mismo.

1.4 Metodología

Metodología Descriptiva.

Se utilizará esta metodología para establecer las características de los módulos de Arduino e IoT con Cloud, estableciendo la relación entre ellos.

Metodología Analítica

Se efectuará un análisis de la eficiencia del proyecto en base a los costos del servicio propuesto, con los datos que se tenía anteriormente versus los datos resultados que se tenga luego.

Metodología Experimental

Mediante la experimentación se comprobará el funcionamiento del sistema en el cual va a monitorear las variables en tiempo real de la piscina de truchas arcoíris.

CAPÍTULO 2

FUNDAMENTACIÓN TEÓRICA

2.1. IoT

Es una de las tecnologías que más se encuentran en auge tanto a nivel doméstico como industrial como se puede apreciar en la Figura 2.1. Porque gracias a la interconexión de dispositivos que se encuentren en un mismo lugar, se puede dar soluciones innovadoras para que dicho lugar funcione automáticamente generando una base de datos, en donde pueda considerar las necesidades que hacen falta y que componentes o dispositivos aún se encuentran con suficientes recursos o autonomía respectivamente. (Vishwakarma, Upadhyaya, Kumari, & Mishra, 2019)

Figura 2.1. Conexión de dispositivos IoT



El sector IoT se está implementando con mayor demanda, desde la pandemia del COVID-19, y con ello aparecen novedosos diseños y nuevas formas de comunicación. Fuente (Valdeolmillos, 2020)

2.2. CLOUD

Considerado comúnmente desde principios de su aparición como nube, que en sí es un paradigma que se lo desarrolla cada vez más con mejoras con las cuales se pueden ampliar las fronteras del conocimiento informático y digital como se aprecia en la Figura 2.2. (Hossain, Khan, Noor, & Hasan, 2016)

Figura 2.2. Cloud



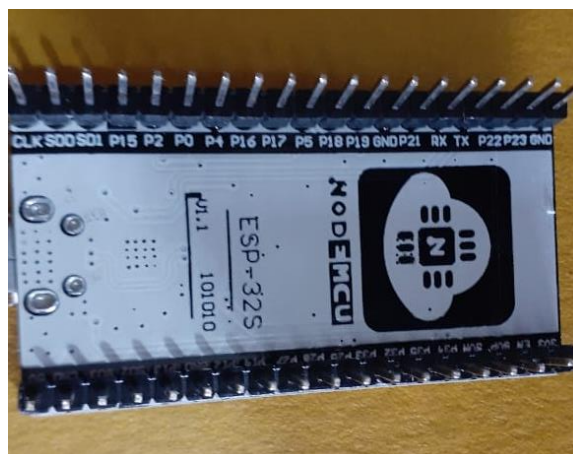
La computación que está basada en la internet, es lo que se conoce comúnmente como Cloud. Fuente: (Ealde, 2019)

Ya que oferta servicios computacionales en tiempo real, que, dicho de otra manera, nos ayuda con el trabajo informático con conexión a la red (tradicionalmente internet), que nos facilita el trabajo desde otros lugares y no como se trabajaba desde y solamente del ordenador personal. Ahora con este nuevo servicio, se abren muchos recursos, porque podemos seguir trabajando con nuestros archivos, proyectos, etc. Desde el lugar que necesitemos.

2.3. NODE MCU-32S

El NODE MCU-32S es un dispositivo compacto como se muestra en la Figura 2.3 además de poseer baja potencia que va tomando bastante interés en el mercado para la elaboración de proyectos IoT de alta fiabilidad capaz de funcionar de forma eficaz en entornos industriales. Formado por circuitos de calibración avanzados el NODE MCU-32S está en la capacidad de eliminar dinámicamente los ruidos del circuito externo y adaptarse a los cambios en las condiciones externas que se presenten. Se puede ahorrar y manejar la energía consumida gracias a su nuevo sistema de consumo ultra bajo, con las condiciones que se le den mediante software, para que cuando el módulo no se encuentre transmitiendo datos o no haga uso de todas sus funciones, se mantenga en un modo de operación básica para que tenga un consumo de energía parcial. La integración de Wi-Fi, Bluetooth y Bluetooth LE asegura que se pueda supervisar una extensa gama de aplicaciones. (Espressif, 2021)

Figura 2.3. Microcontrolador NODE MCU ESP-32S



El microcontrolador del ESP 32 es uno de los más actuales con mayores prestaciones para realizar trabajos con conexión a la internet. Elaborado por: Chanataxi Carlos, Pilca Henry

En la Tabla 2.1 se muestra las características del NODE MCU-32S respecto a otros dispositivos utilizados en el mercado para aplicaciones IoT. Los sensores de oxígeno disuelto y pH necesitan trabajar con una gran velocidad de transmisión de datos por lo que se elige el módulo antes mencionado, debido a que viene con mayor resolución para la adquisición de datos, así como también cuenta con una frecuencia de transmisión de 160 Mhz hasta 240 Mhz permitiendo obtener una menor pérdida de datos en el tiempo de transmisión. Los puertos ADC son muy importantes al momento de escoger el módulo de trabajo porque los sensores de pH y oxígeno disuelto entregan un dato de forma analógica, necesitando de manera obligatoria más de dos puertos ADC, el NODE MCU-32S cuenta con un mayor número de puertos ADC respecto a los demás.

Tabla 2.1. Comparativa de módulos para trabajar con IoT

Características	ESP8266	Arduino 2560	NODE MCU-32S
Resolución	10 bits	10 bits	12 bits
ADC	1	16	18
DAC	No	54	2 de 8 bits
SRAM	50kB	8kB	448 kB
Memoria FLASH	4Mbytes	256kB	4Mbytes (+)
GPIO	17	54	36
Voltaje alimentación	2.5 V – 3.6V	5V – 12V	2.5 V – 3.6V
Voltaje de entrega	3.3V	5V	1V, 2V, 3.3V
Frecuencia	80 Mhz	16 Mhz	160 Mhz - 240 Mhz
WIFI	HT20	No	HT40
Bluetooth	No	No	4.2 y BLE
Dimensiones	57 x 30,4 mm	101 x 53 mm	57 x 30,4 mm
Precio	8\$	22 \$	8\$

Tabla de comparación con las respectivas características de varios elementos utilizados en aplicaciones IoT. Elaborado por: Chanataxi Carlos, Pilca Henry

El voltaje de alimentación de los 3 módulos comparados varía entre 2.5V a 12V, en el caso del módulo utilizado para el presente proyecto trabaja con un voltaje de 3.6V entregado por su regulador de voltaje incorporado a comparación de los otros dos dispositivos. Los módulos para aplicaciones IoT trabajan con un voltaje de entrega que oscila entre 3.3V a 5V, los sensores utilizados no trabajan con un voltaje similar entre ellos por lo que es necesario poder variar este parámetro, el único dispositivo que nos permite realizar esta variación es el NODE MCU-32S, el cual tiene un voltaje de entrega variable entre 1V y 3.3V. Los módulos ESP 8266 y el ESP 32 cuentan con un

módulo WiFi integrado y en el caso de este último cuenta también con Bluetooth 4.2 y BLE.

Las especificaciones antes mencionadas permitieron escoger el módulo NODE MCU-32S sobre los otros elementos comparados debido a su relación costo-beneficio puesto que sus prestaciones de acuerdo con nuestras necesidades es la mejor opción.

2.4. SENSOR DE OXÍGENO DISUELTO

Este sensor de oxígeno disuelto está en la capacidad de medir la saturación de oxígeno en el agua continuamente y gracias a esta información es posible determinar la calidad de la misma, es clasificado como un elemento pasivo el cual genera pequeñas cantidades de voltaje, tomando en cuenta si se encuentra en un estanque o en un reservorio con mayores dimensiones y dependiendo del grado de saturación de oxígeno que esté presente en los terminales de la membrana que se presenta en el encapsulado de la Figura 2.4 la cual es sensible y envía la información hacia el transductor encargado de procesar la información. (Pérez, 2013)

Figura 2.4. Sensor de oxígeno disuelto



Sonda del sensor de oxígeno disuelto con su respectivo conector BNC. Elaborado por: Chanataxi Carlos, Pilca Henry

2.5. SENSOR DE POTENCIAL DE HIDRÓGENO (pH)

El pH permite cuantificar cuán ácida o básica (alcalina) es una solución. Es un índice catalogado como logarítmico de la concentración de los iones de hidrógeno en una solución acuosa. La escala se debe leer de forma inversa, de manera que los valores de pH disminuyen cuando aumentan los niveles de iones hidrógeno y viceversa. El agua limpia o pura tiene un nivel de pH de 7, los valores que se encuentren por debajo de

dicho rango se consideran ácidos y los valores localizados por arriba como básicos. Normalmente el rango de la escala de pH se encuentra definido entre 0 y 14. (Artero, Nogueras, & Mànuel, 2012)

En la Figura 2.5 se muestra el encapsulado de una sonda galvánica la cual ayudará a medir el rango de pH en la piscina de truchas La Merced.

Figura 2.5. Sensor de pH



Sonda del sensor de pH con su líquido regulador y su conector BNC. Elaborado por: Chanataxi Carlos, Pilca Henry

2.6. SENSOR DE TEMPERATURA

El sensor de temperatura DS18B20 es un dispositivo que proporciona mediciones de 9 a 12 bits para que el usuario use esos datos de la forma que sea necesaria. Está dotado por 3 cables que son: datos, alimentación y tierra como se observa en la Figura 2.6.

Figura 2.6. Sensor de temperatura



Conectores del sensor de temperatura y su encapsulado protector del sensor para que pueda ser introducido en el agua sin problemas. Elaborado por: Chanataxi Carlos, Pilca Henry.

2.7. THINGSPEAK

Es una plataforma online utilizada para trabajar aplicaciones IoT ya que brinda facilidades para proyectos pequeños, así como también para proyectos robustos en el cual admite la recolección y almacenamiento de datos tomados por los sensores del proyecto hacia la nube, de tal forma que se pueda construir un abanico de aplicaciones IoT gracias a su interfaz web de fácil manejo como se muestra en la Figura 2.7. A su vez Thingspeak está en la capacidad de ofrecer aplicaciones que permiten analizar y visualizar los datos recolectados en el software MATLAB y manejar los datos a nuestra conveniencia. (MathWorks, 2021)

Figura 2.7. Plataforma de ThingSpeak

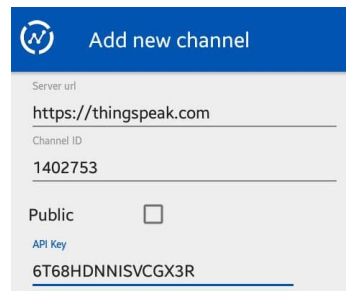


Interfaz de la página principal de la plataforma de ThingSpeak para poder crear los histogramas.
Fuente: (MathWorks, 2021)

2.8. THINGVIEW

Esta aplicación móvil que nos permite mirar todos los canales creados previamente en ThingSpeak, solamente ingresando el ID de nuestro canal de forma rápida y fácil como se muestra en la Figura 2.8. La App respeta la configuración de Windows en lo que respecta a canales públicos, como son las siguientes características: color, escala de tiempo, tipo de gráfico y el número de resultados; en tanto que, la versión actual es compatible con los gráficos de líneas y columnas, y las tablas de spline se encuentran como gráficos de líneas. (Google Play, 2020)

Figura 2.8. Interfaz de ThingView



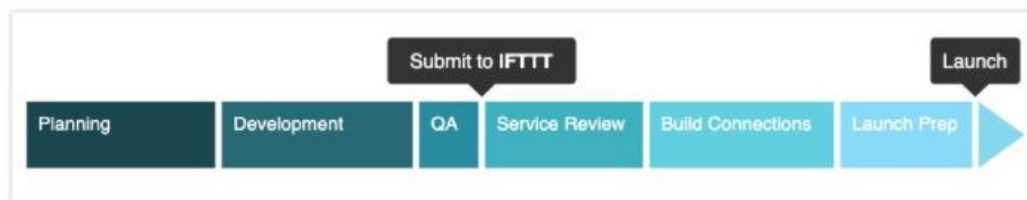
The image shows a web form titled 'Add new channel' with the ThingSpeak logo. It contains the following fields: 'Server url' with the value 'https://thingspeak.com', 'Channel ID' with the value '1402753', a 'Public' checkbox which is unchecked, and 'API Key' with the value '6T68HDNNISVCGX3R'.

Aplicación móvil para poder visualizar los histogramas de ThingSpeak. Fuente: (DMCA, 2020)

2.9. IFTTT

Funciona como una pasarela para la conexión o comunicación de dispositivos con aplicaciones móviles que se usan actualmente para hacer más llevadera la administración de dispositivos IoT como se puede ver en la Figura 2.9. Esta plataforma abre una amplia gama de funciones que podemos sacar provecho para proyectos, ya sea desde los más básicos, hasta los más complejos, todo depende de cómo se le de uso. (Linden Tibbets, 2010)

Figura 2.9. Trama de IFTTT



Protocolo de acción de IFTTT mediante el cual actúa ante un evento para que trabaje un activador.
Fuente: (Inc, 2019)

2.10. WEBHOOK

Webhook es un concepto de API muy útil para aplicaciones como se evidencia en la Figura 2.10 debido a que proporciona otras utilidades con información en tiempo real. Webhook también es una URL enlace que se agrega a la aplicación para que los datos enviados se puedan recibir directamente al mismo tiempo que el Enlace URL que se ha determinado. Esto permite a los desarrolladores enviar mensajes a un servidor y recibir respuestas basadas en eventos sin tener que pedir al servidor una respuesta. (NS, 2020)

CAPÍTULO 3

DISEÑO E IMPLEMENTACIÓN

En el presente capítulo, damos paso a la descripción del software desarrollado y del hardware usado en el proyecto, en el cual se muestra la estructura para censar la temperatura, nivel de oxígeno disuelto y PH en la piscina del criadero de truchas La Merced.

3.1. Parámetros establecidos para control de la calidad del agua

El criadero de truchas La Merced consta con una piscina de 5 x 1,50 metros en la cual existen problemas con la calidad del agua debido a la falta de oxigenación de la misma, como se visualiza en la Figura 3.1 la caída del agua no es la adecuada por lo que es necesario controlar de manera adecuada los parámetros de esta piscina, debido a lo antes mencionado el crecimiento de las truchas de especie arcoíris se ha visto afectado de manera directa.

Figura 3.1. Piscina del criadero de truchas La Merced



Pisca del criadero de truchas La Merced, la cual necesita el control de la calidad del agua. Elaborado por: Chanataxi Carlos, Pilca Henry

De acuerdo a los parámetros establecidos para un óptimo crecimiento se dice que la temperatura debería estar entre 13 y 18 °C para que el crecimiento de las truchas sea rápido, pero existe un problema debido a que el crecimiento de las truchas a temperaturas altas depende del nivel del oxígeno disuelto en el agua, por lo que si tenemos una temperatura elevada esta debe ser proporcional el oxígeno disuelto para evitar pérdidas en la producción. Por esa razón la temperatura debe de ser promedio para evitar toda clase de problemas.

Con respecto al pH del agua, es un parámetro que se debe encontrar en un rango de 6,5 a 8,5 ya que, acorde a este podremos saber si el agua de la piscina necesita una limpieza, debido a que con el pasar del tiempo suelen alojarse algas en los costados de la piscina y combinadas con las pequeñas porciones de heces que quedan de las truchas en el agua hacen que el pH se eleve con el pasar del tiempo, por lo que no permite el correcto crecimiento de las truchas. En resumen, con ese dato podemos realizar una limpieza oportuna y evitar pérdidas o enfermedades venideras en las truchas.

El oxígeno disuelto en el agua debe de mantenerse en un rango 7,5 a 12 mg/L ya que, si descende de dichos valores las truchas podrían tener enfermedades o incluso morir. Esto puede suceder por una pérdida de caudal en la fuente, lo cual puede ser señal de obstrucción de la tubería o que existe fugas en el trayecto de transporte de agua, entonces este parámetro puede alertarnos cuando se produzcan este tipo de fallos.

3.2. Diagramas de bloques del equipo

En la Figura 3.2. Se puede apreciar los bloques de los que básicamente está compuesto el diseño, el bloque de sensores se encuentra conformado por los sensores de PH, temperatura y oxígeno disuelto, que tienen como función principal la recolección de los datos de la piscina de truchas, mismas que permitirán verificar los valores que contiene la piscina de cada una de ellas, haciendo una relación con los datos obtenidos para optimizar la piscina de truchas en el criadero La Merced.

El bloque de Alimentación tiene la responsabilidad de proporcionar la energía o voltaje hacia la placa o PCB, que prácticamente es o comúnmente se usa un adaptador de 5 Voltios y 2 Amperios, del cual van a encontrarse alimentados los sensores y el módulo de comunicación.

El bloque del módulo NODE MCU-32S es el encargado de obtener y procesar los datos de los sensores, además de ser el controlador principal que se encuentra en comunicación con la nube para que los datos antes mencionados permanezcan reportándose constantemente.

Figura 3.2. Diagrama de bloques de la comunicación y control IoT

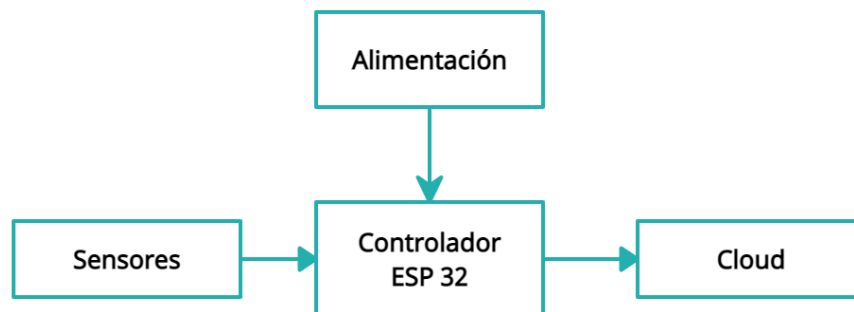


Diagrama de bloques con los respectivos elementos utilizados en el presente proyecto. Elaborado por: Chanataxi Carlos, Pilca Henry

En la Figura 3.3. Se puede observar la arquitectura para realizar el envío de datos hacia la nube (Cloud) por medio del módulo NODE MCU-32S, lo cual facilitará subir los resultados o datos monitoreados de la piscina de truchas. Y con todo esto, poder supervisar los parámetros en tiempo real desde cualquier lugar que se encuentre el usuario o persona encargada.

Figura 3.3. Diagrama de bloques de la comunicación de la nube con el NODE MCU-32S

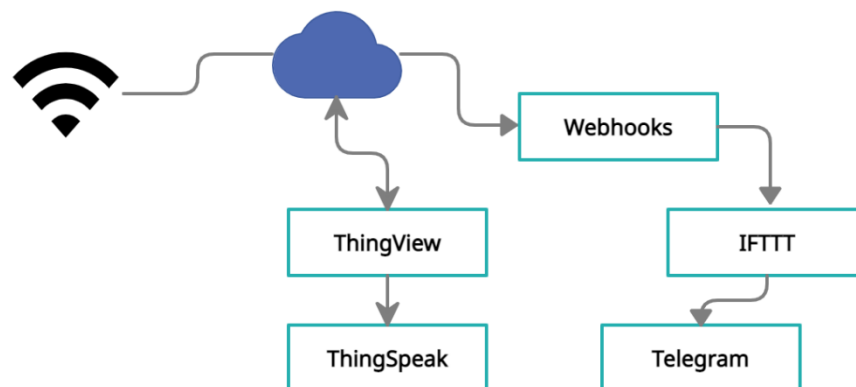


Diagrama de flujo para la comunicación de los elementos con el servicio de Cloud. Elaborado por: Chanataxi Carlos, Pilca Henry

Ahora bien, el desarrollo del algoritmo para la adquisición de datos con el módulo NODE MCU-32S tal como se visualiza en la Figura 3.4 se da inicio con la implementación de las librerías necesarias para todo el proceso, así como la declaración de las variables. Y para hacerlo posible, se procede a la habilitación de los puertos o pines analógicos del módulo, para luego mediante código en el software pasar por un respectivo escalamiento (dependiendo de la resolución del módulo) para cada uno de los sensores. Con dicho escalamiento se procede a la conversión de los valores analógicos de los sensores, en cadenas de caracteres entendibles para el microprocesador, y hacer con ellos lo que nos convenga.

Figura 3.4. Diagrama de flujo de adquisición y envío de datos del NODE MCU-32S

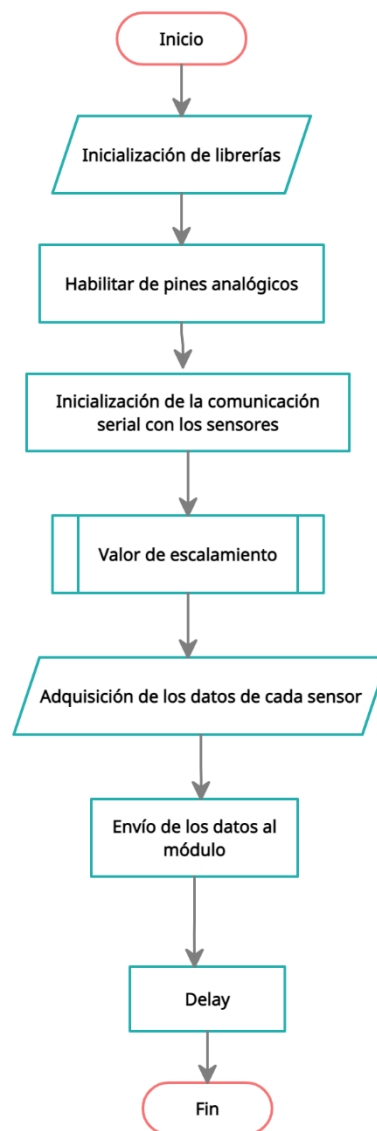


Diagrama de flujo para la lectura y envío de datos por medio del módulo NODE MCU-32S

En la Figura 3.5 se aprecia el diagrama de flujo de la lectura y obtención de los datos por parte de NODE MCU-32S partiendo de la resolución de lectura, ya que dicho microcontrolador trabaja con 10 y 12 bits por lo que se debe seleccionar una de las dos antes de empezar. Para luego continuar con el escalamiento de los valores para que puedan ser leídos sin ningún problema, y enviarlos hacia la nube.

Figura 3.5. Diagrama de flujo de lectura y escalamiento de los datos de los sensores.

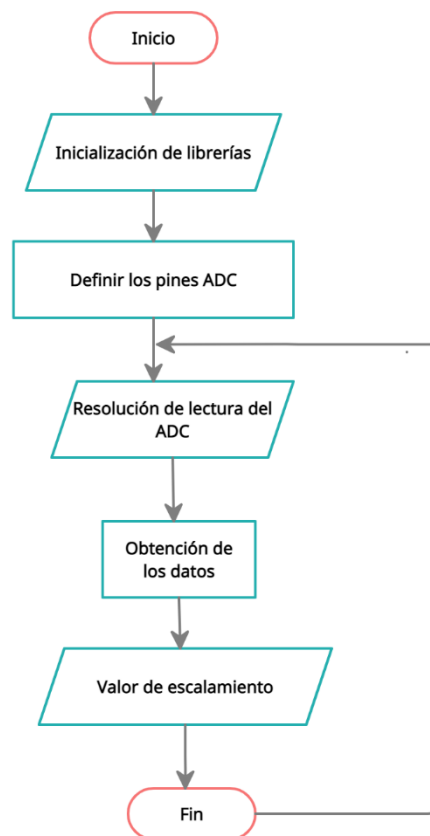
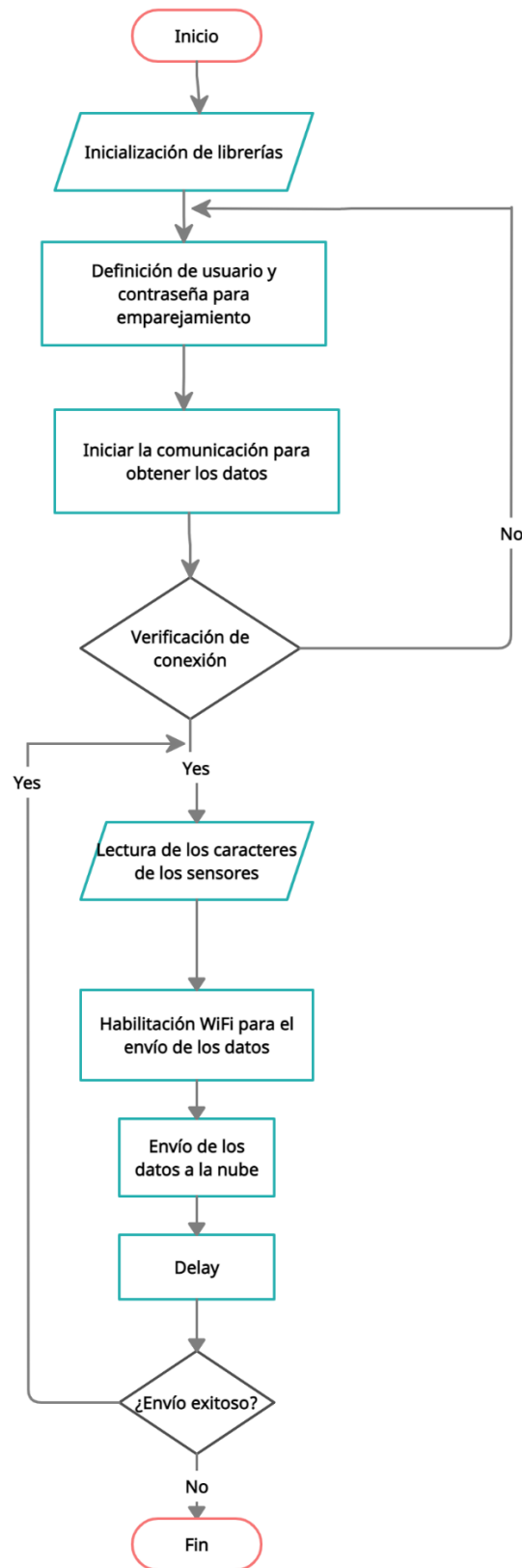


Diagrama de flujo para la lectura correcta de los datos analógicos utilizados. Elaborado por: Chanataxi Carlos, Pilca Henry.

Como se puede apreciar en la Figura 3.6 se encuentra la lógica del algoritmo de recepción de datos del NODE MCU-32S, tomando en cuenta las condiciones revisadas previamente en la hoja de datos del mismo. Se ve que, del bloque de decisión del envío de datos, se encuentra condicionado para que los datos se transmitan constantemente hacia la nube y así, poder garantizar un flujo de datos cada cierto tiempo y poder realizar el análisis de los mismos, si así fuese el caso.

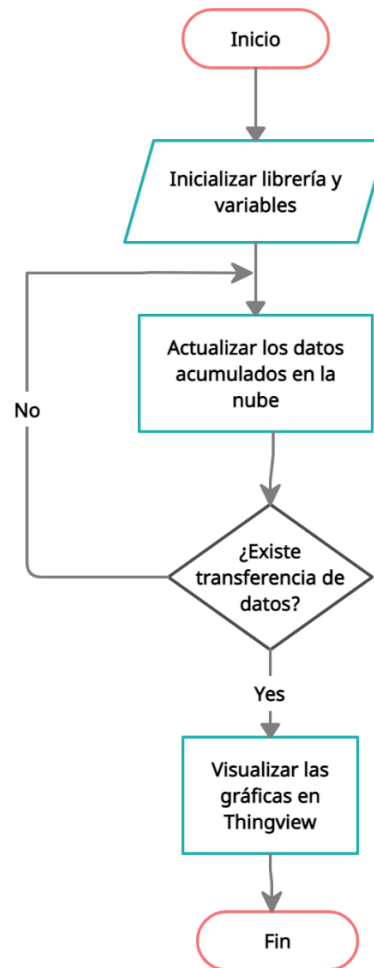
Figura 3.6. Diagrama de flujo de comunicación y envío de datos a la nube.



Establecimiento de la comunicación con la nube y su respectivo envío de datos para su posterior monitoreo. Elaborado por: Chanataxi Carlos, Pilca Henry.

En la Figura 3.7 se observa el diagrama de flujo para la visualización de los datos almacenados en la nube. Para lo cual se necesita autorización del administrador para que no pueda tener acceso cualquier persona, sino que solamente pueden tener acceso los encargados del lugar y poder visualizar los datos de pH, temperatura y oxígeno disuelto en el agua.

Figura 3.7. Diagrama de flujo de observación de los datos almacenados en la nube



La visualización de los datos almacenados en la nube se realiza por medio de la actualización de datos en los histogramas como muestra el diagrama de flujo. Elaborado por: Chanataxi Carlos, Pilca Henry.

En la siguiente Figura 3.8 se puede apreciar la petición de servicio IFTTT, con la cual podemos alertar cuando los valores de pH, temperatura y nivel de oxígeno disuelto sobrepasen de los valores óptimos. Es decir que, cuando esto suceda, el servicio de IFTTT envía un mensaje de alerta hacia el ThingSpeak, para que pueda ser visualizado en los registros que tenemos creados en la nube, y con eso poder tomar acciones ante eso.

Figura 3.8. Diagrama de flujo del servicio IFTTT

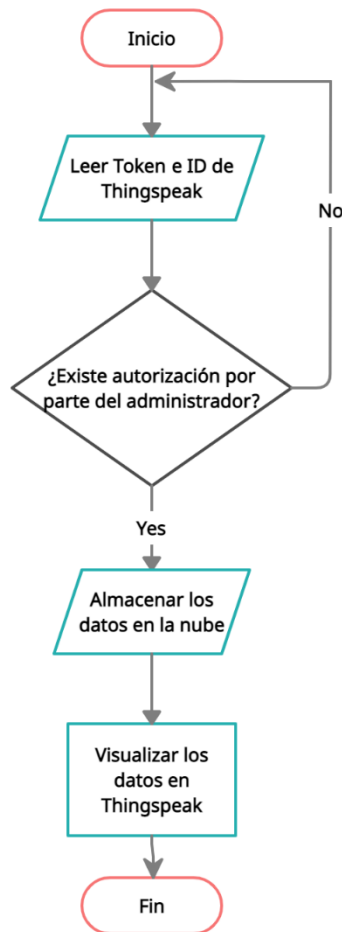


Diagrama de flujo que indica los requisitos que debe de tener el servicio IFTTT para poder realizar la acción requerida. Elaborado por: Chanataxi Carlos, Pilca Henry.

En la Figura 3.9 vemos el diagrama de flujo que nos ayuda con la alerta hacia nuestra cuenta de Telegram, cuando necesitamos ser alertados por exceso de los valores que habíamos considerado. Básicamente lo que se hace, es leer los datos de la nube constantemente, y cuando uno de dichos valores sobrepase o se exceda, automáticamente se produce una petición en Webhooks la cual es un servicio de IFTTT. Una vez obtenida la petición se difunde el mensaje en Telegram, claro está que, el mensaje de alerta fue creado en la aplicación en donde se escribió el mensaje que queremos que sea transmitido mediante Telegram.

Figura 3.9. Diagram de flujo de alerta por valores excedidos



Diagrama de flujo para tener conocimiento de las alertas que se puedan generar en el sistema.
Elaborado por: Chanataxi Carlos, Pilca Henry.

Para las alertas se puede ver en la Figura 3.10 la arquitectura del diagrama de flujo el cual, empieza con la lectura de los datos de la nube para luego verificarlos con los valores establecidos como óptimos para la piscina, si se excede de los valores predefinidos se genera una solicitud http por parte del Webhooks para poder enviar un mensaje de alerta a la red social Telegram, y con ello logramos mantener en constante monitoreo a la persona encargada de la supervisión de las piscinas de truchas.

Figura 3.10. Diagrama de flujo de alerta vía Telegram

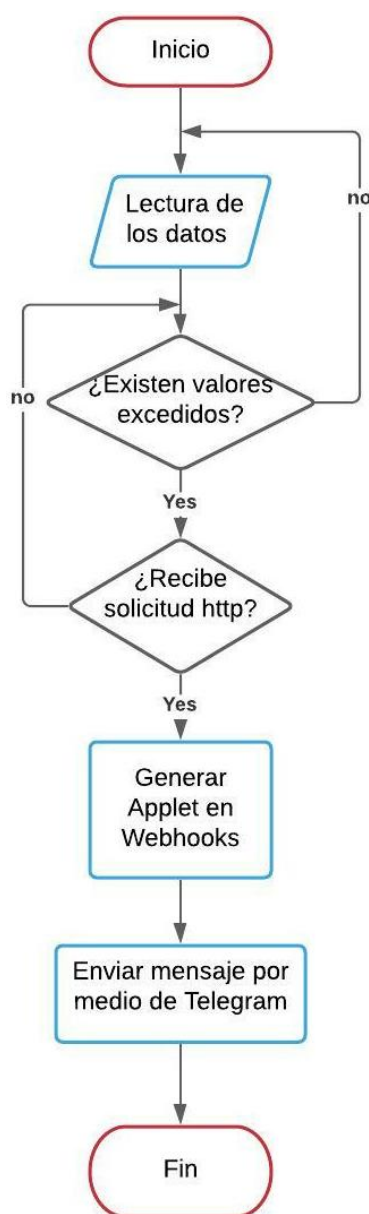
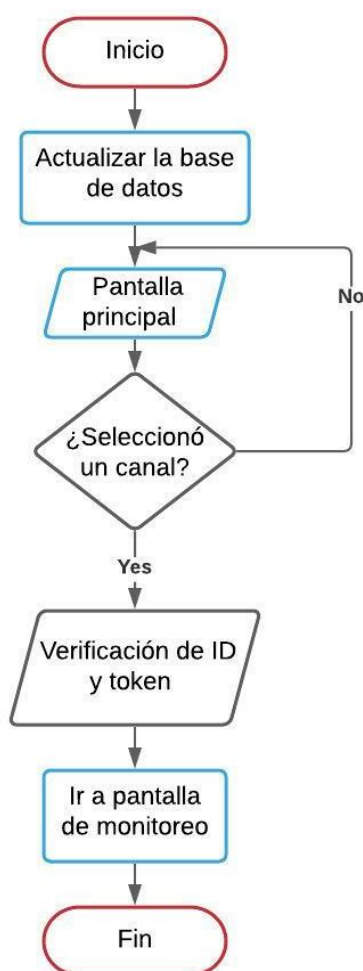


Diagrama de flujo en el cual Telegram sirve como gestor de avisos ante la petición del servicio http.
Elaborado por: Chanataxi Carlos, Pilca Henry

Para la visualización en la aplicación móvil, se puede examinar la Figura 3.11 en donde explica claramente su función. Cuando se tiene descargada la aplicación en nuestro móvil, ingresamos en la aplicación y una vez que piquemos en el canal que deseamos visualizar, la App procede a la verificación del ID y token, los cuales son proporcionados por la plataforma de ThingSpeak. Una vez que se ha verificado dichos datos, la App procede a transferirnos a la pantalla de monitoreo, en donde se encuentra nuestro canal con los históricos en tiempo real, de los datos de pH, temperatura y nivel de oxígeno disuelto.

Figura 3.11. Diagrama de flujo de la visualización de datos en la aplicación móvil



Una vez actualizado los datos se procede con la selección de un canal el cual permite el monitoreo de los mismos con el fin de visualizar los histogramas en el dispositivo móvil. Elaborado por: Chanataxi Carlos, Pilca Henry

Como se puede examinar en la Figura 3.12, se hallan los pasos seguidos para las alertas que puede generar la aplicación móvil. Empezamos con la lectura de los datos almacenados en la nube, para luego notar los datos de pH, temperatura y nivel de oxígeno disuelto, tanto en ThingSpeak como también en la aplicación móvil. Seguidamente se realiza la comparación de los valores adquiridos con los rangos propuestos y que son los óptimos para la piscina, entonces si uno de ellos rebasa de los valores propuestos como óptimos, se produce el mensaje de alerta hacia el administrador o usuario.

Figura 3.12. Diagrama de flujo para la producción de alertas

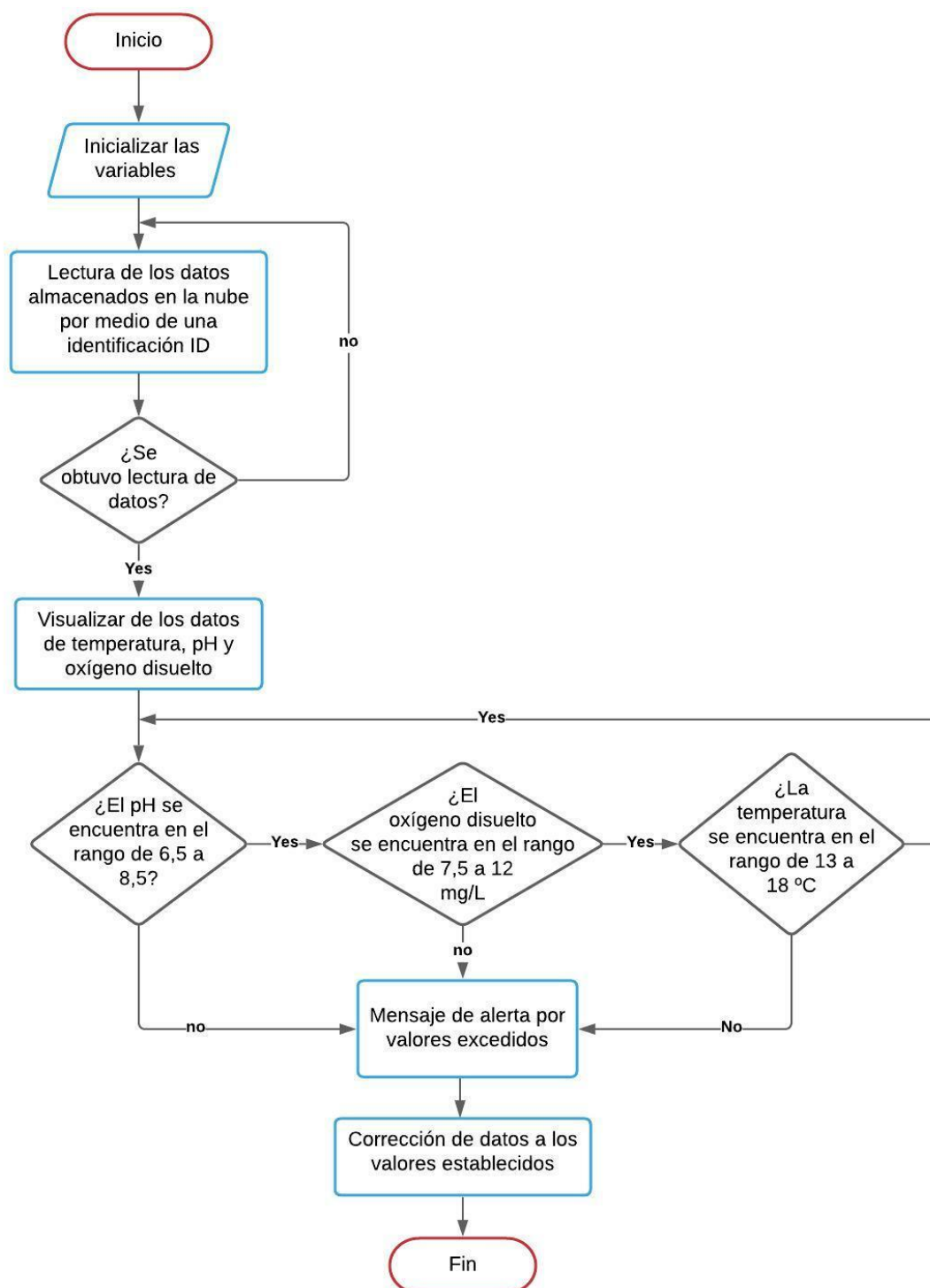
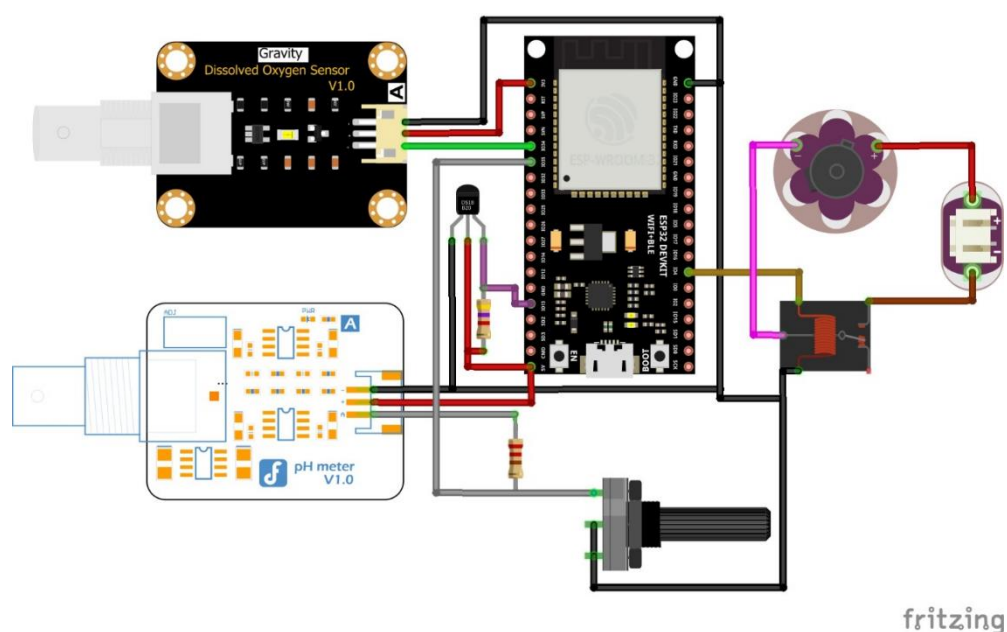


Diagrama de flujo para el cual los límites establecidos generarán una alerta permitiendo al usuario conocer el estado de la calidad del agua. Elaborado por: Chanataxi Carlos, Pilca Henry

3.3. Diagramas esquemáticos

A continuación, en la Figura 3.13 se puede observar la representación 3D de cada uno de los elementos utilizados en el circuito electrónico.

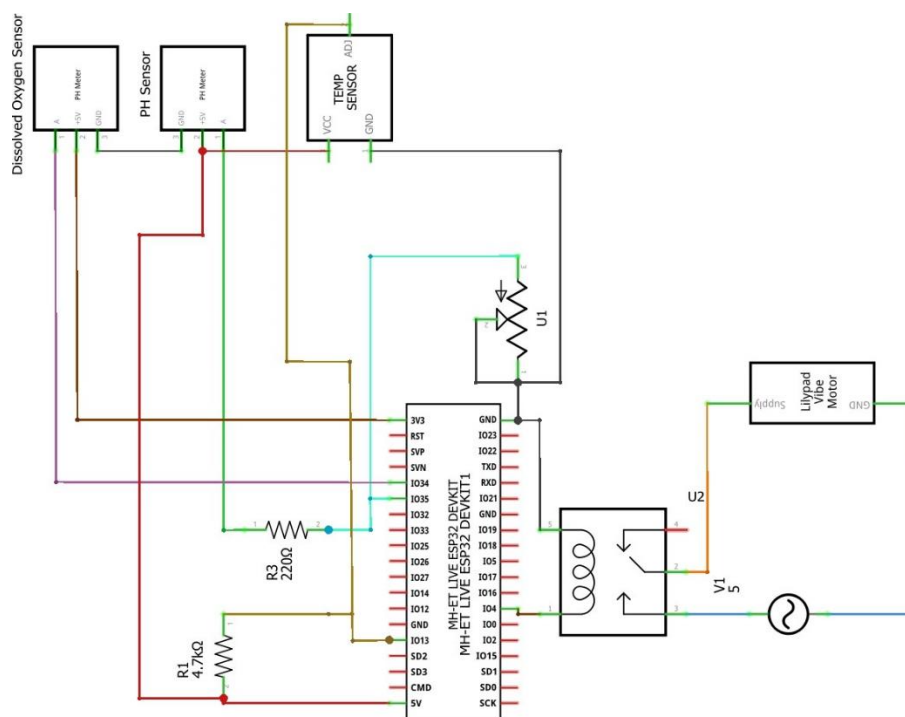
Figura 3.13. Diagrama de las conexiones de los dispositivos con el NODE MCU-32S



Vista 3D de la conexión del módulo NODE MCU-32S y los respectivos sensores utilizados (Temperatura, pH, oxígeno disuelto). Elaborado por: Chanataxi Carlos, Pilca Henry

En el siguiente diagrama esquemático de la Figura 3.14 se presenta la conexión realizada para el envío de datos, por medio del módulo Node MCU32-S y los respectivos sensores utilizados (pH, temperatura, oxígeno disuelto) y el actuador a accionarse cuando los valores predeterminados se sobrepasen.

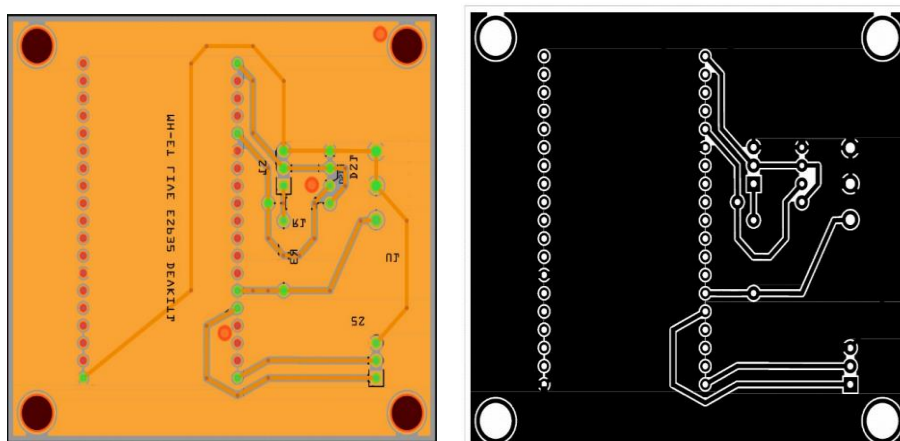
Figura 3.14. Diagrama esquemático realizado para poder elaborar la placa PBC



Vista previa de la conexión esquemática de los sensores con el módulo NODE MCU-32S. Elaborado por: Chanataxi Carlos, Pilca Henry.

Para conectar los elementos antes mencionados se elaboró una placa PCB como el de la Figura 3.15, la cual consta con las respectivas normativas de diseño, el software utilizado para esta placa fue FRITZING.

Figura 3.15. Diseño e implementación de la placa PCB para su impresión en cobre



PCB creada en el software Fritzing para la correcta ubicación de los elementos. Elaborado por:
Chanataxi Carlos, Pilca Henry

3.4. Lectura de datos en la plataforma ThingSpeak

ThingSpeak es una plataforma la cual permite la lectura de datos por medio de gráficas o históricos, como se puede visualizar en la Figura 3.16, los datos obtenidos son almacenados directamente en la nube o Cloud. La actualización de datos se realiza cada 14 segundos, por lo que la comunicación entre el Node MCU32-S y este servicio web debe establecer este retraso en el código de programación.

Figura 3.16. Interfaz de histogramas en ThingSpeak



Página principal de histogramas creados para la visualización y supervisión de los parámetros de la piscina de truchas La Merced. Elaborado por: Chanataxi Carlos, Pilca Henry

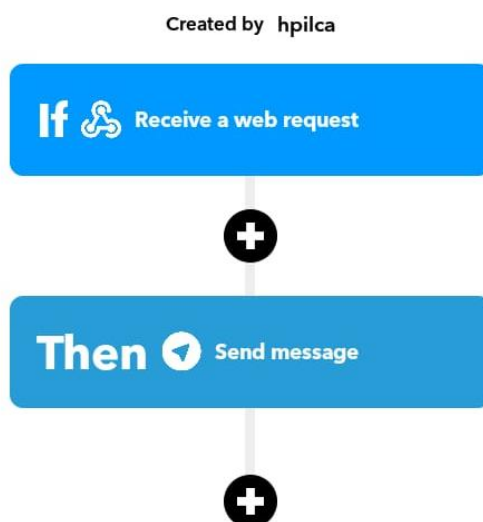
Tenemos cuatro históricos de los cuales, tres esta dedicados para inspeccionar los datos obtenidos de los sensores, y el cuarto es para que se pueda saber en dónde se encuentra ubicada la piscina, en caso de que exista otra persona encargada del mantenimiento de esta, así cuando se produzcan las alertas dicha persona pueda acercarse al lugar preciso de manera rápida para realizar el control de la calidad del agua.

3.5. Generar alertas por medio de ThingSpeak

Una vez creados los campos (PH, Temperatura y Oxígeno Disuelto) estos datos serán almacenados en la nube permitiendo que el usuario pueda revisarlos de manera periódica de tal forma que se pueda establecer un tiempo adecuado para el control de la calidad de agua y así mantener los rangos establecidos.

En el caso de exceder los parámetros establecidos la plataforma Thingspeak puede generar alertas para el control de estos. En este caso se generó una alerta por medio de la red social Telegram, para poder solicitar este tipo de alerta se necesita un servicio IFTTT el cual realiza una petición HTTP a la plataforma Webhooks y si esta acepta dicha petición entonces se genera un mensaje de alerta asignado como se muestra en la Figura 3.17.

Figura 3.17. Evento creado en el servicio IFTTT



Establecimiento de la alerta generada por el servicio IFTTT por medio de plataforma Webhooks y Telegram. Elaborado por: Chanataxi Carlos, Pilca Henry

3.6. Aplicación para el monitoreo de datos

Los datos obtenidos se pueden visualizar de manera continua por medio de la aplicación ThingView. La aplicación utilizada es una función preestablecida por la plataforma ThingSpeak por lo que es necesario generar el ID y API KEY del canal creado como se muestra en la figura 3.18.

Figura 3.18. Origen del ID y API KEY

Monitoreo de Sensores

Channel ID: **1402753** | Sensor de PH, Temperatura
Author: mwa0000022685246
Access: Private

[Private View](#) [Public View](#) [Channel Settings](#) [Sharing](#) [API Keys](#)

Write API Key

Key

[Generate New Write API Key](#)


Read API Keys

Key

Formación de los códigos de acceso (ID y API KEY) para la vinculación con la aplicación ThingView. Elaborado por: Chanataxi Carlos, Pilca Henry

Con los datos de acceso generados se procedió a vincular la aplicación con el servidor que en este caso es <https://thingspeak.com>. La aplicación generará una solicitud de confirmación que se muestra en la figura 3.19 permitiendo controlar los campos que se desea visualizar y de la misma forma mostrará estos datos por medio de los históricos creados en el servidor Thingspeak.

Figura 3.19. Confirmación de los canales a exhibirse en la aplicación de ThingView

 Confirm channel

Channel

Monitoreo de Sensores

Description

Sensor de PH, Temperatura y Oxígeno disuelto

Fields

PH

Temperatura

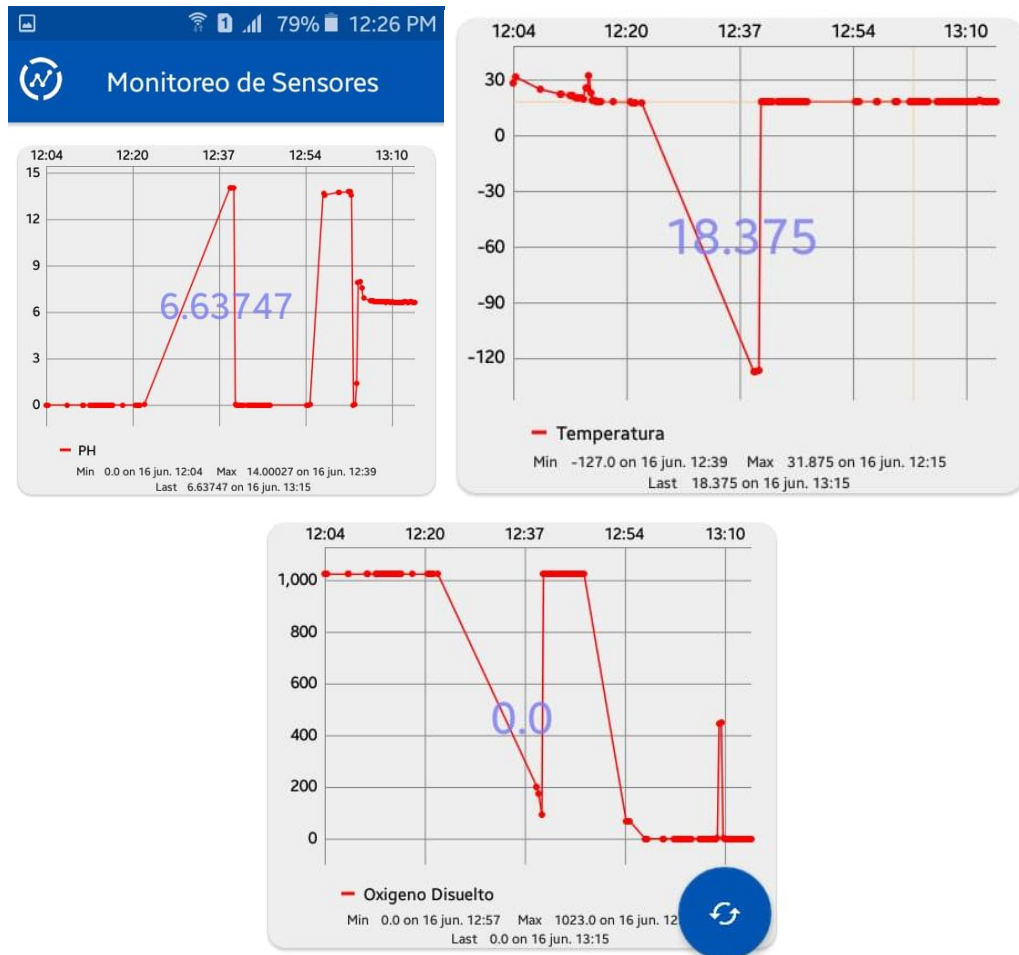
Oxígeno Disuelto

Done

Validación del canal a exhibirse en la aplicación móvil ThingView. Elaborado por: Chanataxi Carlos, Pilca Henry

A continuación, en la figura 3.20 se muestra la pantalla del dispositivo móvil en donde se puede visualizar los datos obtenidos por el monitoreo de los sensores utilizados.

Figura 3.20. Vista de históricos en ThingView

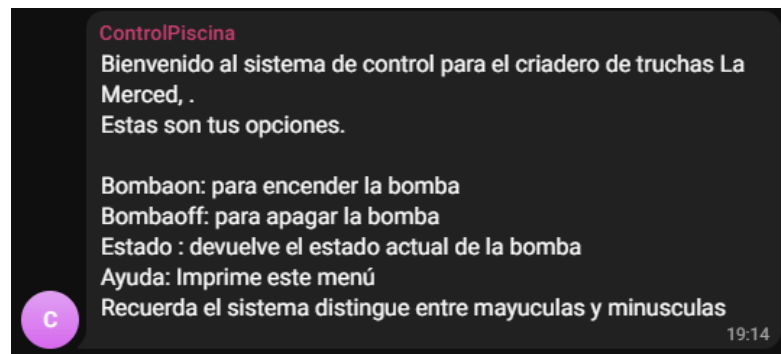


Lectura de datos de los sensores de temperatura, pH y oxígeno disuelto en la aplicación móvil ThingView. Elaborado por: Chanataxi Carlos, Pilca Henry.

3.7. Sistema de activación de una bomba por medio de Telegram.

La red social Telegram gracias a sus prestaciones nos brinda una buena compatibilidad con aplicaciones IoT para poder enlazar varias funciones por medio de comandos de texto establecidos de acuerdo a la necesidad del usuario, para el control de la bomba ubicada en la piscina de truchas especie arcoíris se generó un menú en donde se señala los comandos establecidos para el control del sistema como se muestra en la Figura 3.21.

Figura 3.21. Menú para el control el del sistema en el criadero de truchas La Merced



Menú generado por medio del sistema para el control de la bomba ubicada en la piscina de truchas especia arcoíris. Elaborado por: Chanataxi Carlos, Pilca Henry.

CAPÍTULO 4

PRUEBAS Y RESULTADOS

En este capítulo se hace la descripción general de las pruebas realizadas al presente proyecto que como consecuencia muestran los resultados de latencia, retardo y disponibilidad de la red que se presentan a continuación.

De la misma manera se presentan las pruebas generadas para el envío y recepción de comandos por parte de la red social Telegram que nos permite activar un actuador que permite controlar una bomba la cual por medio del flujo de agua ayuda a mejorar los parámetros establecidos para la calidad de agua en la piscina de truchas especie arcoíris.

4.1. Latencia

La latencia ha sido desde hace mucho tiempo la piedra angular de una red debido a que la mayoría de los diseñadores se centran en otros objetivos antes que en la latencia como tal al momento de implementar algo nuevo, lo antes mencionado ha generado conflictos en la transmisión de datos por la pérdida de paquetes. (Stephen M. Rumble, 2011)

En términos generales la latencia es el tiempo que le toma a un paquete en ser enviado hasta cuando llega a su destino, en donde se puede sufrir pérdidas por diferentes circunstancias, principalmente se pierde un paquete en el camino por el crecimiento a gran escala del internet y los diversos puntos por los que tiene que pasar dicho paquete. (Angos, 2017)

La medida de la latencia se la realiza en milisegundos (ms) dando a entender que si el número es pequeño la latencia es baja, caso contrario si el número es grande la latencia es alta y con ello el sistema se vuelve lento independientemente del computador u ordenador que se esté utilizando. Dicho esto, se realizó las pruebas de latencia en diferentes horarios para obtener los tiempos de transferencia o verificar si existe pérdida de paquetes en las horas que más se congestiona la red, así como también cuando la red no está saturada.

Para las pruebas de latencia se obtuvo las direcciones ip del servidor y el sistema, en el caso del servidor se estableció una dirección ip asignada por defecto por thingspeak.com, en el segundo caso respectivamente la dirección ip fue generada por el router de internet al cual está conectado el sistema, a continuación, se muestran las direcciones ip utilizadas.

ip-servidor: 184.106.153.149

ip-sistema: 192.168.1.20

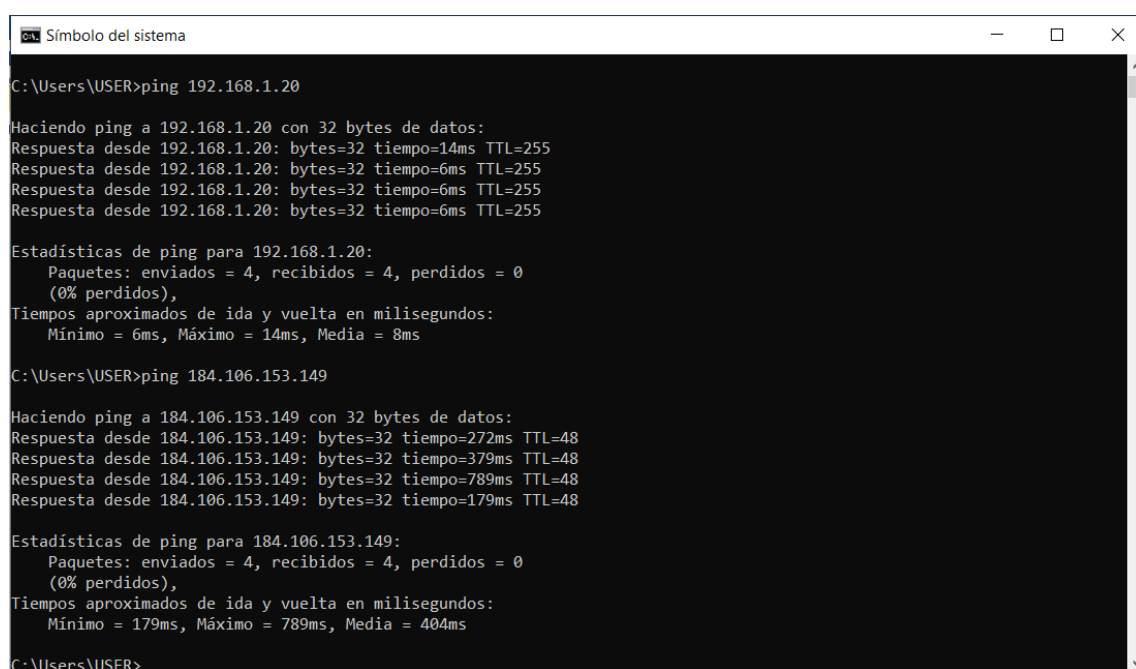
En la figura 4.1 se presenta las características de los pings realizados tanto al servidor como al dispositivo IoT, para realizar las medidas de latencia mencionadas se procedió a utilizar el intérprete de comandos de Windows o CMD en el cual se utilizó los siguientes comandos: ping *ip-servidor*, ping *ip-sistema*

Los tiempos de latencia obtenidos en la para el envío y recepción de cuatro paquetes a las 10:00 am fueron los siguientes:

Servidor: Mínimo= 179ms, Máximo= 789ms, Media= 404ms.

Sistema: Mínimo= 6ms, Máximo= 14ms, Media= 8ms.

Figura 4.1. Datos de latencia obtenidos a las 10:00 am



```
Símbolo del sistema
C:\Users\USER>ping 192.168.1.20

Haciendo ping a 192.168.1.20 con 32 bytes de datos:
Respuesta desde 192.168.1.20: bytes=32 tiempo=14ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=6ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=6ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=6ms TTL=255

Estadísticas de ping para 192.168.1.20:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
        (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 6ms, Máximo = 14ms, Media = 8ms

C:\Users\USER>ping 184.106.153.149

Haciendo ping a 184.106.153.149 con 32 bytes de datos:
Respuesta desde 184.106.153.149: bytes=32 tiempo=272ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=379ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=789ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=179ms TTL=48

Estadísticas de ping para 184.106.153.149:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
        (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 179ms, Máximo = 789ms, Media = 404ms

C:\Users\USER>
```

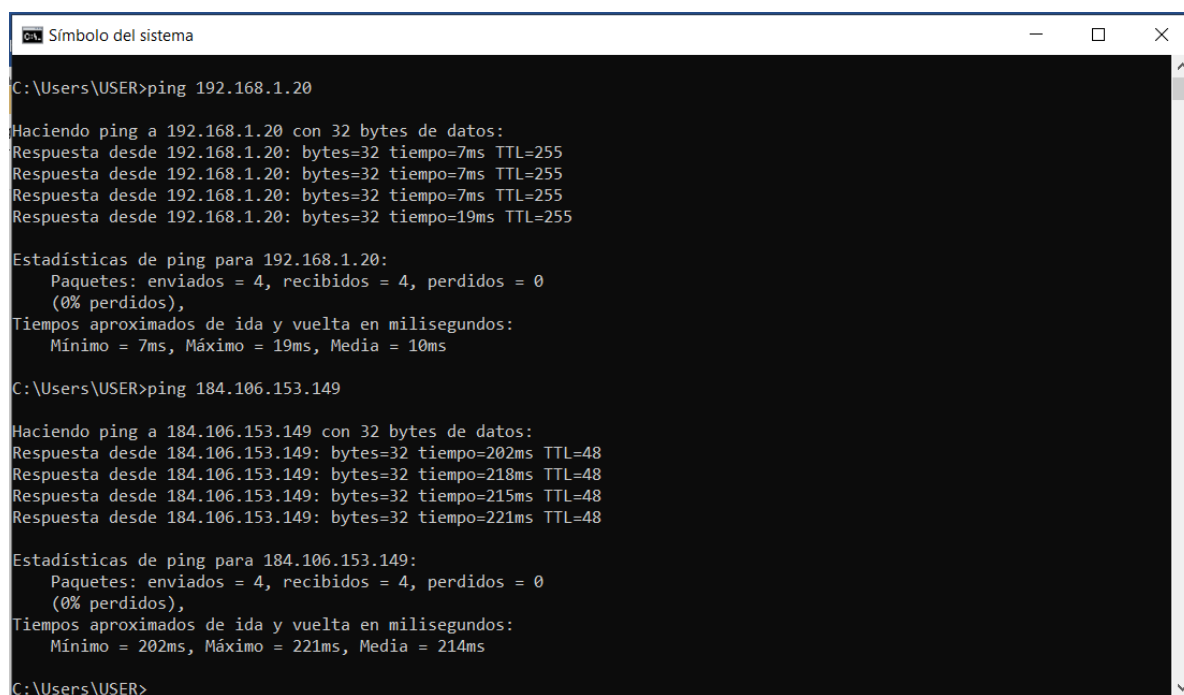
Elaborado por: Chanataxi Carlos, Pilca Henry

Los tiempos obtenidos en la Figura 4.2 fueron tomados a las 12:00 am, estos tiempos varían poco que los obtenidos anteriormente, este se debe generalmente a que a tempranas horas de la mañana existen muchos usuarios conectados a la red por lo que esta se vuelve lenta.

Servidor: Mínimo= 202ms, Máximo= 221ms, Media= 214ms.

Sistema: Mínimo= 7ms, Máximo= 19ms, Media= 10ms.

Figura 4.2. Datos de latencia obtenidos a las 12:00 am



```
C:\Users\USER>ping 192.168.1.20

Haciendo ping a 192.168.1.20 con 32 bytes de datos:
Respuesta desde 192.168.1.20: bytes=32 tiempo=7ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=7ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=7ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=19ms TTL=255

Estadísticas de ping para 192.168.1.20:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
        (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 7ms, Máximo = 19ms, Media = 10ms

C:\Users\USER>ping 184.106.153.149

Haciendo ping a 184.106.153.149 con 32 bytes de datos:
Respuesta desde 184.106.153.149: bytes=32 tiempo=202ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=218ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=215ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=221ms TTL=48

Estadísticas de ping para 184.106.153.149:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
        (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 202ms, Máximo = 221ms, Media = 214ms

C:\Users\USER>
```

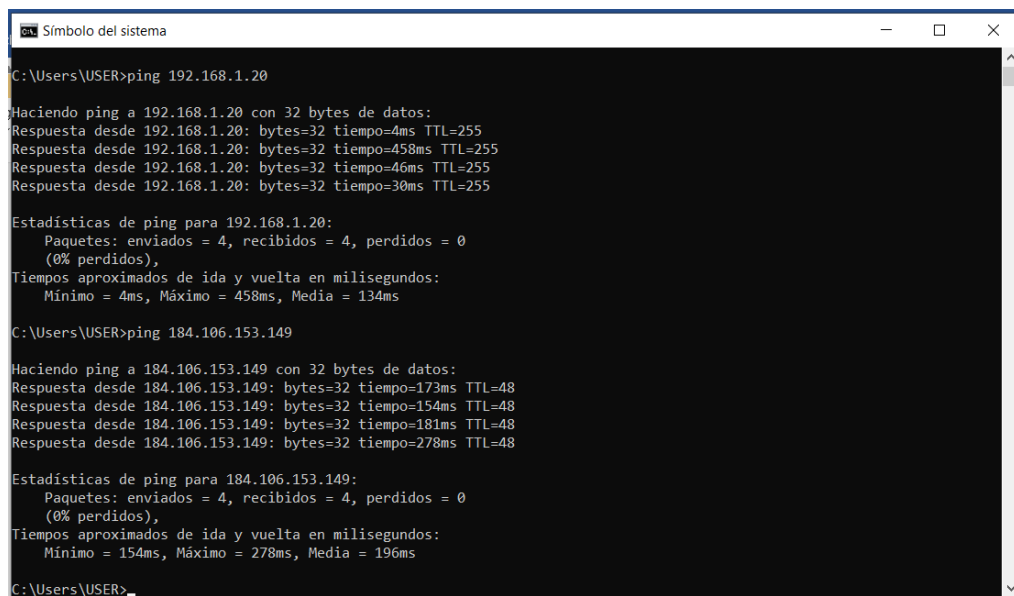
Elaborado por: Chanataxi Carlos, Pilca Henry

Se realizaron las respectivas pruebas a las 14:00 pm como se observa en la Figura 4.3 para seguir identificando la latencia que existe en la red, de lo que se obtuvo los siguientes valores:

Servidor: Mínimo= 154ms, Máximo= 278ms, Media= 196ms.

Sistema: Mínimo= 4ms, Máximo= 458ms, Media= 134ms.

Figura 4.3. Datos obtenidos a las 14:00 pm



```
Símbolo del sistema
C:\Users\USER>ping 192.168.1.20

Haciendo ping a 192.168.1.20 con 32 bytes de datos:
Respuesta desde 192.168.1.20: bytes=32 tiempo=4ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=458ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=46ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=30ms TTL=255

Estadísticas de ping para 192.168.1.20:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
              (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 4ms, Máximo = 458ms, Media = 134ms

C:\Users\USER>ping 184.106.153.149

Haciendo ping a 184.106.153.149 con 32 bytes de datos:
Respuesta desde 184.106.153.149: bytes=32 tiempo=173ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=154ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=181ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=278ms TTL=48

Estadísticas de ping para 184.106.153.149:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
              (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 154ms, Máximo = 278ms, Media = 196ms

C:\Users\USER>
```

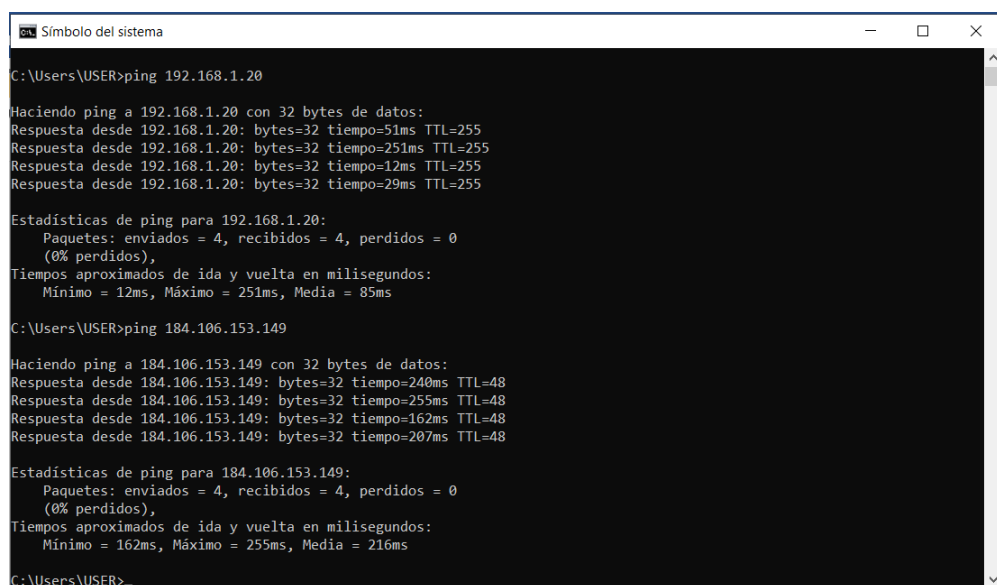
Elaborado por: Chanataxi Carlos, Pilca Henry

Continuando con el análisis de los tiempos de latencia, se realizó pruebas en la tarde donde existe una demanda de datos alta, para verificar el estado de la red como se indica en la Figura 4.4 se consiguieron los tiempos:

Servidor: Mínimo= 162ms, Máximo= 255ms, Media= 216ms.

Sistema: Mínimo= 12ms, Máximo= 251ms, Media= 85ms.

Figura 4.4. Datos obtenidos a las 16:00 pm



```
Símbolo del sistema
C:\Users\USER>ping 192.168.1.20

Haciendo ping a 192.168.1.20 con 32 bytes de datos:
Respuesta desde 192.168.1.20: bytes=32 tiempo=51ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=251ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=12ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=29ms TTL=255

Estadísticas de ping para 192.168.1.20:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
              (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 12ms, Máximo = 251ms, Media = 85ms

C:\Users\USER>ping 184.106.153.149

Haciendo ping a 184.106.153.149 con 32 bytes de datos:
Respuesta desde 184.106.153.149: bytes=32 tiempo=240ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=255ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=162ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=207ms TTL=48

Estadísticas de ping para 184.106.153.149:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
              (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 162ms, Máximo = 255ms, Media = 216ms

C:\Users\USER>
```

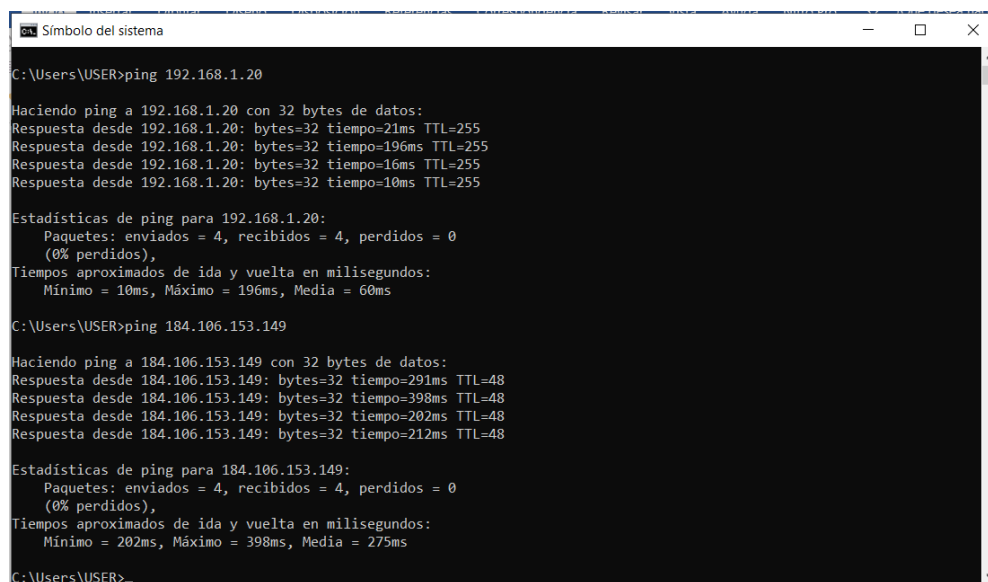
Elaborado por: Chanataxi Carlos, Pilca Henry

Se procedió con el análisis en horas de la tarde para constatar el envío y recepción de paquetes obteniendo los valores de la Figura 4.5.

Servidor: Mínimo= 202ms, Máximo= 398ms, Media= 275ms.

Sistema: Mínimo= 10ms, Máximo= 196ms, Media= 60ms.

Figura 4.5. Datos obtenidos a las 18:00 pm



```
Símbolo del sistema
C:\Users\USER>ping 192.168.1.20

Haciendo ping a 192.168.1.20 con 32 bytes de datos:
Respuesta desde 192.168.1.20: bytes=32 tiempo=21ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=196ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=16ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=10ms TTL=255

Estadísticas de ping para 192.168.1.20:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
        (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 10ms, Máximo = 196ms, Media = 60ms

C:\Users\USER>ping 184.106.153.149

Haciendo ping a 184.106.153.149 con 32 bytes de datos:
Respuesta desde 184.106.153.149: bytes=32 tiempo=291ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=398ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=202ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=212ms TTL=48

Estadísticas de ping para 184.106.153.149:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
        (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 202ms, Máximo = 398ms, Media = 275ms

C:\Users\USER>
```

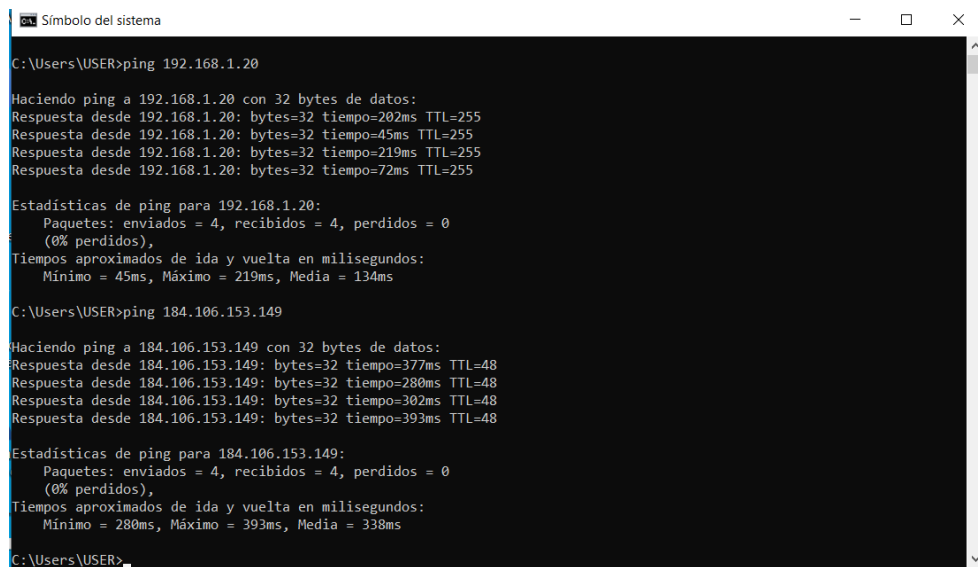
Elaborado por: Chanataxi Carlos, Pilca Henry

Para finalizar el análisis de latencia se realizó un ping a las 20:00 pm como se muestra en la Figura 4.6 tanto del servidor como al dispositivo para terminar con el seguimiento de las latencias realizadas en distintos horarios.

Servidor: Mínimo= 280ms, Máximo= 393ms, Media= 338ms.

Sistema: Mínimo= 45ms, Máximo= 219ms, Media= 134ms.

Figura 4.6. Datos obtenidos a las 20:00 pm



```
C:\Users\USER>ping 192.168.1.20

Haciendo ping a 192.168.1.20 con 32 bytes de datos:
Respuesta desde 192.168.1.20: bytes=32 tiempo=202ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=45ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=219ms TTL=255
Respuesta desde 192.168.1.20: bytes=32 tiempo=72ms TTL=255

Estadísticas de ping para 192.168.1.20:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
            (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 45ms, Máximo = 219ms, Media = 134ms

C:\Users\USER>ping 184.106.153.149

Haciendo ping a 184.106.153.149 con 32 bytes de datos:
Respuesta desde 184.106.153.149: bytes=32 tiempo=377ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=280ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=302ms TTL=48
Respuesta desde 184.106.153.149: bytes=32 tiempo=393ms TTL=48

Estadísticas de ping para 184.106.153.149:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
            (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 280ms, Máximo = 393ms, Media = 338ms

C:\Users\USER>
```

Elaborado por: Chanataxi Carlos, Pilca Henry

Los valores obtenidos son aproximadamente de 150 y 80 ms para el servidor y el sistema respectivamente, manteniéndose en un rango aceptable para ejecutar los servicios de envío y transferencia de datos sin que estos se pierdan en el camino, garantizando confiabilidad del sistema para el monitoreo y control constante de la piscina de truchas.

4.2 Retardos

Los retardos son el tiempo de extremo a extremo el cual inicia el instante en que el servidor envía un paquete de datos o mensaje y finaliza luego de que recibe la señal de reconocimiento. Para las redes WiFi las mediciones de retardo sirven para establecer la información básica de retroalimentación que el servidor principal puede hacer uso para inducir el estado de conexión de la red con el servidor y los dispositivos remotos. (C. Lozoya, 2018)

Haciendo uso del software WinMTR se realizó el análisis de retardos al enviar y recibir paquetes desde nuestro dispositivo hasta el servidor thingspeak.com.

La Figura 4.7 está compuesta por una columna de Host en la cual se encuentran las ip de los hosts que tiene que atravesar un paquete hasta llegar al destino. En la segunda columna se encuentra el porcentaje de tiempo que se tarda nuestro dispositivo en

transmitir los paquetes, a continuación, en las columnas 3 y 4 se muestra el envío y recepción de datos, en las siguientes 4 columnas se indican el mejor, peor, ultimo y el promedio de los tiempos en milisegundos (ms) que se demoran los paquetes en ser transmitidos.

Figura 4.7. Datos obtenidos a las 10:00 am

WinMTR statistics							
Host	- %	Sent	Recv	Best	Avrg	Wrst	Last
192.168.1.1	- 0	118	118	1	4	39	1
10.52.0.1	- 2	110	108	9	66	1170	143
173.4.46.186.static.anycast.cnt-grms.ec	- 2	109	107	8	78	2352	82
190.152.252.153	- 2	110	108	11	69	1152	146
10.9.3.17	- 2	108	106	96	167	2508	236
10.9.3.17	- 1	111	110	101	192	3790	244
ae-11.a01.nycmny17.us.bb.gin.ntt.net	- 2	110	108	94	153	1242	238
ae-5.r21.nwrknj03.us.bb.gin.ntt.net	- 2	110	108	106	161	1252	247
ae-3.r22.chcgil09.us.bb.gin.ntt.net	- 1	111	110	108	200	3795	261
ae-30.a00.chcgil09.us.bb.gin.ntt.net	- 2	110	108	115	174	1278	280
ae-0.rackspace.chcgil09.us.bb.gin.ntt.net	- 2	108	106	117	187	2535	255
dcpe1-coreb.ord1.rackspace.net	- 1	111	110	110	198	3776	260
coreb-core6.ord1.rackspace.net	- 1	111	110	110	199	3779	260
core6-aggr1501a-7.ord1.rackspace.net	- 3	106	103	120	164	307	263
thingspeak.com	- 1	112	111	108	191	2507	248

Elaborado por: Chanataxi Carlos, Pilca Henry

Los valores que rigen en la Figura 4.8 son los que se realizaron en horas de la mañana y se obtuvo un tiempo de retardo aproximado de 155ms.

Figura 4.8. Datos obtenidos a las 12:00 am

WinMTR statistics								
Host	- %	Sent	Recv	Best	Avrg	Wrst	Last	
192.168.1.1	- 0	138	138	1	2	16	1	
10.52.0.1	- 3	127	124	9	61	313	10	
173.4.46.186.static.anycast.cnt-grms.ec	- 3	127	124	9	60	300	13	
190.152.252.153	- 2	128	126	8	93	4058	10	
10.9.3.17	- 3	127	124	97	145	387	105	
10.9.3.17	- 1	131	130	100	190	4154	136	
ae-11.a01.nycmny17.us.bb.gin.ntt.net	- 3	127	124	94	141	388	98	
ae-5.r21.nwrknj03.us.bb.gin.ntt.net	- 6	115	109	101	150	381	110	
ae-3.r22.chcgil09.us.bb.gin.ntt.net	- 4	123	119	106	158	411	109	
ae-30.a00.chcgil09.us.bb.gin.ntt.net	- 3	127	124	114	161	388	121	
ae-0.rackspace.chcgil09.us.bb.gin.ntt.net	- 3	122	119	117	220	4259	159	
No response from host	- 100	28	0	0	0	0	0	
dcpe1-coreb.ord1.rackspace.net	- 1	131	130	107	202	4159	137	
coreb-core6.ord1.rackspace.net	- 3	127	124	110	159	405	131	
core6-aggr1501a-7.ord1.rackspace.net	- 4	123	119	118	167	521	120	
thingspeak.com	- 4	123	119	109	158	416	113	

Elaborado por: Chanataxi Carlos, Pilca Henry

En la Figura 4.9 tenemos los tiempos ejercidos para la transmisión de los paquetes en el horario de la tarde, del cual se obtuvo un tiempo de retardo promedio aproximado de 185ms.

Figura 4.9. Datos obtenidos a las 14:00 pm

WinMTR statistics								
Host	- %	Sent	Recv	Best	Avg	Wrst	Last	
192.168.1.1	- 0	147	147	1	2	22	4	
10.52.0.1	- 1	143	142	8	71	739	25	
173.4.46.186.static.anycast.cnt-grms.ec	- 2	139	137	9	63	794	10	
190.152.252.153	- 1	143	142	9	66	869	44	
10.9.3.17	- 2	139	137	97	152	877	113	
10.9.3.17	- 2	139	137	100	157	889	144	
ae-11.a01.nycmny17.us.bb.gin.ntt.net	- 2	139	137	94	156	871	105	
ae-5.r21.nwrknj03.us.bb.gin.ntt.net	- 1	143	142	104	167	962	149	
ae-3.r22.chcgil09.us.bb.gin.ntt.net	- 2	139	137	107	164	889	155	
ae-30.a00.chcgil09.us.bb.gin.ntt.net	- 2	139	137	115	168	966	115	
ae-0.rackspace.chcgil09.us.bb.gin.ntt.net	- 2	139	137	117	177	873	119	
No response from host	- 100	30	0	0	0	0	0	
dcpe1-coreb.ord1.rackspace.net	- 2	139	137	107	169	966	107	
coreb-core6.ord1.rackspace.net	- 1	143	142	110	166	966	155	
core6-aggr1501a-7.ord1.rackspace.net	- 1	143	142	117	171	966	131	
thingspeak.com	- 1	143	142	108	169	883	128	

Elaborado por: Chanataxi Carlos, Pilca Henry

Los tiempos que se observan en la Figura 4.10 se los realizó en la noche para verificar cuánto es el retardo de los paquetes a medida que transcurre el día, de los cuales se tiene un promedio aproximado del tiempo de 150 ms hasta llegar al destino.

Figura 4.10. Datos obtenidos a las 16:00 pm

WinMTR statistics							
Host	- %	Sent	Recv	Best	Avrg	Wrst	Last
192.168.1.1	- 0	275	275	1	3	54	1
10.52.0.1	- 2	255	250	8	63	2259	240
173.4.46.186.static.anycast.cnt-grms.ec	- 2	258	254	8	66	2272	254
190.152.252.153	- 2	256	251	8	55	484	198
10.9.3.17	- 3	252	246	97	139	447	259
10.9.3.17	- 2	260	256	99	148	953	325
ae-11.a01.nycmny17.us.bb.gin.ntt.net	- 2	260	256	94	142	953	273
ae-5.r21.nwrknj03.us.bb.gin.ntt.net	- 3	253	247	101	150	538	326
ae-3.r22.chcgil09.us.bb.gin.ntt.net	- 2	256	251	106	154	525	277
ae-30.a00.chcgil09.us.bb.gin.ntt.net	- 2	263	260	114	174	1967	349
ae-0.rackspace.chcgil09.us.bb.gin.ntt.net	- 2	256	251	117	161	486	259
No response from host	- 100	56	0	0	0	0	0
dcpe1-coreb.ord1.rackspace.net	- 2	259	255	107	160	1959	289
coreb-core6.ord1.rackspace.net	- 2	260	256	109	158	1006	331
core6-aggr1501a-7.ord1.rackspace.net	- 2	264	261	117	169	1006	349
thingspeak.com	- 2	263	260	108	163	1950	291

Elaborado por: Chanataxi Carlos, Pilca Henry

También se obtuvo retardos tomados a las 18:00 pm como se muestra en la Figura 4.11 de la cual se encontró un promedio de 190 ms.

Figura 4.11. Datos obtenidos a las 18:00 pm

WinMTR statistics							
Host	- %	Sent	Recv	Best	Avrg	Wrst	Last
dsldevice	- 0	225	225	1	2	17	1
10.52.0.1	- 2	212	209	8	127	4382	244
173.4.46.186.static.anycast.cnt-grms.ec	- 4	199	193	0	107	3730	3730
190.152.252.153	- 2	212	209	8	123	4615	241
10.9.3.17	- 4	199	193	96	200	3806	3806
10.9.3.17	- 2	207	203	100	214	3780	3780
ae-11.a01.nycmny17.us.bb.gin.ntt.net	- 2	212	208	93	188	962	456
ae-5.r21.nwrknj03.us.bb.gin.ntt.net	- 2	212	209	101	214	4574	343
ae-3.r22.chcgil09.us.bb.gin.ntt.net	- 3	200	194	106	224	4724	404
ae-30.a00.chcgil09.us.bb.gin.ntt.net	- 3	207	202	115	212	942	256
ae-0.rackspace.chcgil09.us.bb.gin.ntt.net	- 2	212	209	117	232	4615	336
No response from host	- 100	46	0	0	0	0	0
dcpe1-coreb.ord1.rackspace.net	- 4	199	192	107	188	673	247
coreb-core6.ord1.rackspace.net	- 2	211	207	109	207	942	466
core6-aggr1501a-7.ord1.rackspace.net	- 2	211	208	117	226	4600	234
thingspeak.com	- 1	211	209	108	244	4592	3983

Elaborado por: Chanataxi Carlos, Pilca Henry

Por último, se realizó la toma de retardos a las 20:00 pm como se aprecia en la Figura 4.12 dando como resultado un tiempo estimado de 199 ms para la transmisión de los paquetes.

Figura 4.12. Datos obtenidos a las 20:00 pm

WinMTR statistics							
Host	- %	Sent	Recv	Best	Avrg	Wrst	Last
dsldevice	- 0	228	228	1	3	89	9
10.52.0.1	- 5	198	190	8	82	1819	18
173.4.46.186.static.anycast.cnt-grms.ec	- 2	209	205	8	117	4010	27
190.152.252.153	- 2	212	208	8	108	1824	11
10.9.3.17	- 4	201	194	97	174	1933	118
10.9.3.17	- 2	212	208	100	203	1942	105
ae-11.a01.nycmny17.us.bb.gin.ntt.net	- 2	209	205	93	199	4296	94
ae-5.r21.nwrknj03.us.bb.gin.ntt.net	- 5	198	190	101	188	1805	104
ae-3.r22.chcgil09.us.bb.gin.ntt.net	- 4	199	193	0	225	4358	121
ae-30.a00.chcgil09.us.bb.gin.ntt.net	- 3	201	195	115	216	4312	115
ae-0.rackspace.chcgil09.us.bb.gin.ntt.net	- 3	208	203	117	218	1955	123
No response from host	- 100	47	0	0	0	0	0
dcpe1-coreb.ord1.rackspace.net	- 3	206	201	0	204	4312	123
coreb-core6.ord1.rackspace.net	- 3	204	198	109	202	1954	129
core6-aggr1501a-7.ord1.rackspace.net	- 5	198	190	117	187	1930	117
thingspeak.com	- 3	204	198	108	204	1943	124

Elaborado por: Chanataxi Carlos, Pilca Henry

Los retardos que mantuvo el sistema en el periodo de tiempo que se tomaron los datos, son favorables para que se pueda trabajar con el sistema sin que el mismo tenga problemas de transmisión y demoras en la red que se encuentra trabajando, porque de lo contrario no se podría trabajar con normalidad ya que podría tardarse mucho en receptor o enviar instrucciones, pero estas pruebas reflejan que el trabajo del dispositivo es bueno y fiable para los usuarios.

4.3 Disponibilidad de la red

De la misma manera se utilizó el software WinMTR para verificar la disponibilidad de la red o tiempo de funcionamiento de la red la cual se mide en porcentaje, este porcentaje representa el tiempo en que nuestro dispositivo tarda en cumplir las exigencias de conectividad de la misma.

En la figura 4.13 se indican los porcentajes de cada uno de los saltos que realizan nuestros paquetes hasta llegar al servidor thingspeak.com, en total se enviaron 1192 paquetes y se recibieron 1170, los datos antes mencionados fueron extraídos en un tiempo prolongado de aproximadamente 1 hora, teniendo como resultado un 2% de pérdida de tiempo de funcionamiento de la red bastante favorable para el trabajo del dispositivo con la nube, reduciendo el riesgo de pérdidas de datos en todo el día de trabajo de NODEMCU-32S.

Figura 4.13. Disponibilidad de la red

WinMTR statistics								
Host	- %	Sent	Recv	Best	Avrg	Wrst	Last	
192.168.1.1 -	0	1295	1295	1	3	41	6	
10.52.0.1 -	3	1182	1157	7	93	3150	9	
173.4.46.186.static.anycast.cnt-grms.ec -	2	1192	1169	8	88	2952	18	
190.152.252.153 -	2	1198	1177	0	93	2809	18	
10.9.3.17 -	2	1195	1173	0	182	2785	104	
10.9.3.17 -	2	1200	1181	99	201	3268	109	
ae-11.a01.nycmny17.us.bb.gin.ntt.net -	3	1167	1138	93	187	2844	118	
ae-5.r21.nwrknj03.us.bb.gin.ntt.net -	3	1178	1152	101	191	3031	131	
ae-3.r22.chcgil09.us.bb.gin.ntt.net -	3	1184	1158	105	188	3269	106	
ae-30.a00.chcgil09.us.bb.gin.ntt.net -	3	1165	1136	115	196	3107	119	
ae-0.rackspace.chcgil09.us.bb.gin.ntt.net -	3	1163	1134	117	198	3097	149	
No response from host -	100	262	0	0	0	0	0	
dcpe1-coreb.ord1.rackspace.net -	3	1175	1148	107	183	3254	136	
coreb-core6.ord1.rackspace.net -	2	1188	1166	108	200	3097	135	
core6-aggr1501a-7.ord1.rackspace.net -	3	1166	1139	117	207	4809	123	
thingspeak.com -	2	1192	1170	107	198	2841	117	

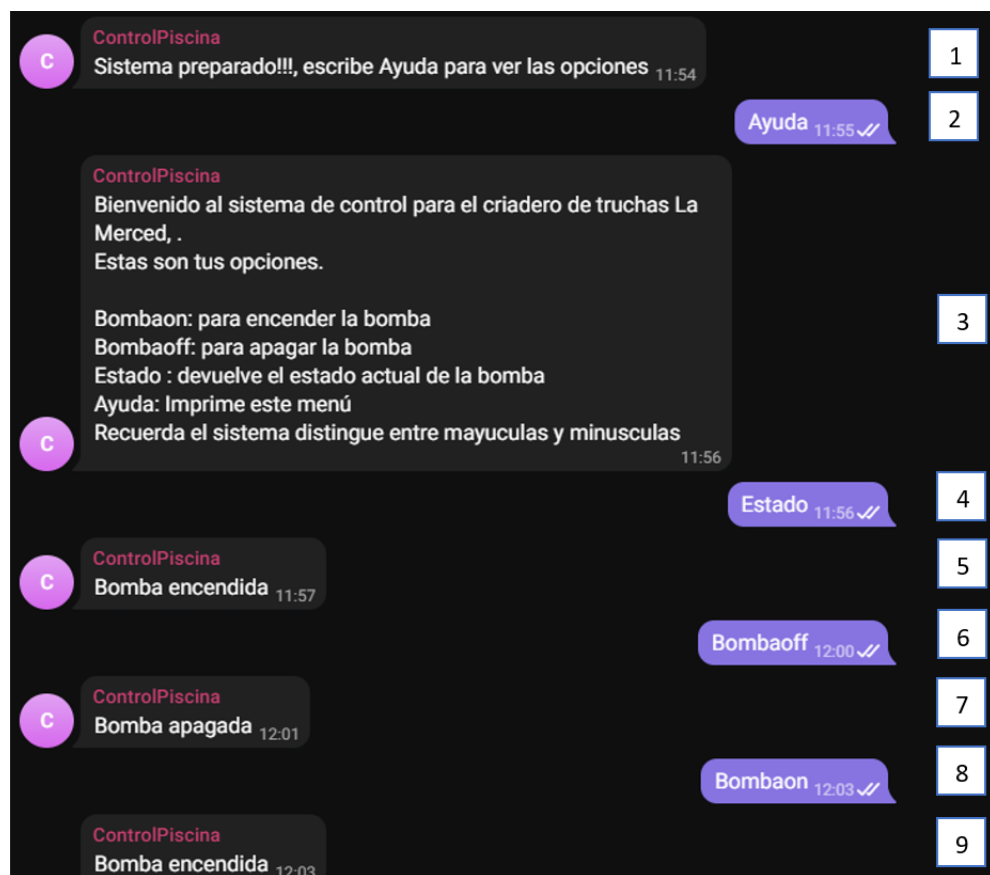
Elaborado por: Chanataxi Carlos, Pilca Henry

4.4 Pruebas de envío y recepción de datos entre el NodeMCU-32s y el usuario.

Respecto a los mensajes que se encuentran en la Figura 4.1 se detalla de forma numérica cada uno de los mensajes del chat como me muestra a continuación:

- 1- Mensaje de conexión correcta a la red del NODE MCU-32S la cual envía un mensaje de texto una vez que se encuentra totalmente conectado el dispositivo y se encuentra transmitiendo datos.
- 2- Comando de texto Ayuda escrito por cualquiera de los usuarios para poder desplegar las opciones y poder ejecutarlas desde Telegram.
- 3- Menú del sistema para realizar el control del actuador en la piscina.
- 4- Comando Estado escrito por uno de los usuarios para ver el estado del actuador.
- 5- Respuesta del sistema informando el estado del actuador.
- 6- Comando Bombaoff escrito por un usuario para apagar la bomba de agua.
- 7- Respuesta del sistema generado por la desactivación de la bomba.
- 8- Comando Bombaon escrito por un usuario para encender la bomba de agua.
- 9- Respuesta del sistema generado por la activación de la bomba.

Figura 4.14. Interfaz del chat de la red social Telegram



Chat en el que se muestra las opciones que tiene el sistema el cual puede ser controlado por la red social Telegram. Elaborado por: Chanataxi Carlos, Pilca Henry

Una vez realizadas las pruebas de envío y recepción de comandos de texto por parte del NODEMCU-32S y los dispositivos móviles de los usuarios se estableció un tiempo de respuesta aproximado entre 7s a 14s como máximo, este retardo se verá afectado directamente por la velocidad que el módulo de internet proporcione en ese instante de tiempo.

4.5 Error y Precisión de los sensores

En este punto se debe considerar que en cuanto más exacta es la medición que se realiza se necesita equipos de instrumentación más costosos debido a que los mismos tienen un error de toma de muestras más bajos. Por otro lado, la ausencia de precisión en los equipos puede resultar todavía más cara, específicamente si se encuentra tomando muestras en tiempo real debido a que esto perjudica la calidad del producto final. Es así que, la precisión tiene un papel importante en la selección de un sensor. (Jalloul, 2010)

Para los sensores usados en el proyecto se va a hacer uso del cálculo del error para poder sondear la precisión que se obtiene de cada uno de ellos. Para lo que se hace uso de las siguientes ecuaciones:

- Error Absoluto = Valor leído – Valor verdadero. Ec. (4.1)

- Error Relativo = Error Absoluto / Valor verdadero. Ec. (4.2)

4.5.1 Error y precisión del sensor de pH

$$\text{Error Absoluto} = 6,8 - 7 = 0,2$$

$$\text{Error Relativo} = 0,2 / 7 = 0,028$$

Del resultado del error relativo podemos deducir que el error que tiene el sensor de pH para este caso es bajo, con lo que obtenemos valores con un nivel de precisión bastante bueno.

4.5.2 Error y precisión del sensor de temperatura

$$\text{Error Absoluto} = 16,40 - 17 = 0,6$$

$$\text{Error Relativo} = 0,6 / 17 = 0,03$$

Se obtuvo un error relativo de 0,03 y de ello se puede decir que la precisión de este sensor es bastante pequeña y podemos recolectar muchos datos con poco error.

4.5.3 Error y precisión del sensor de oxígeno disuelto

$$\text{Error Absoluto} = 8,475 - 8,30 = 0,175$$

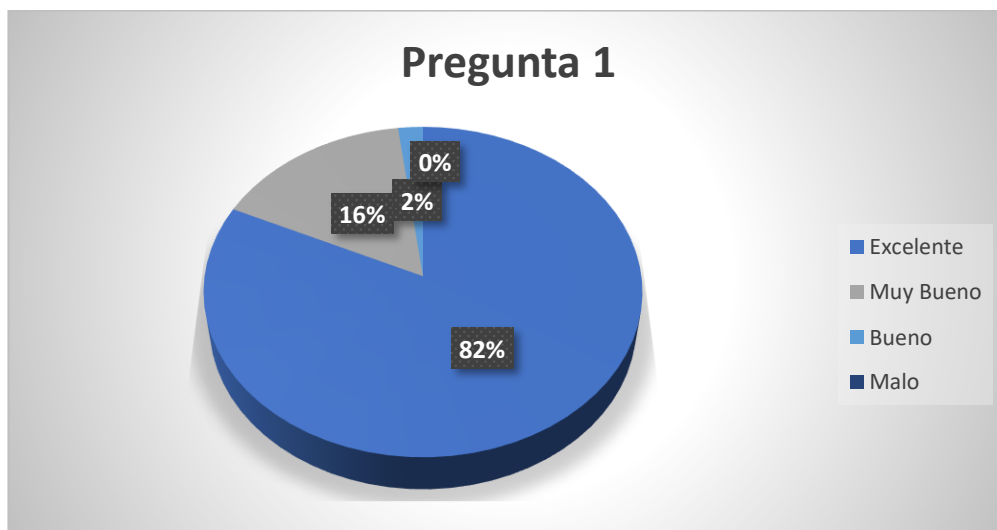
$$\text{Error Relativo} = 0,175 / 8,30 = 0,021$$

Se obtiene un error relativo de 0,021 para lo cual es bastante manejable para este sensor y se puede afirmar que, la precisión de este sensor es bastante buena para trabajar.

4.6 Encuesta para verificar la factibilidad del equipo

4.6.1. ¿Cómo calificaría el dispositivo IoT para el monitoreo de la calidad del agua?

Figura 4.15. Pregunta 1 de la encuesta



Resultados de las opciones de la pregunta 1. Elaborado por: Chanataxi Carlos, Pilca Henry

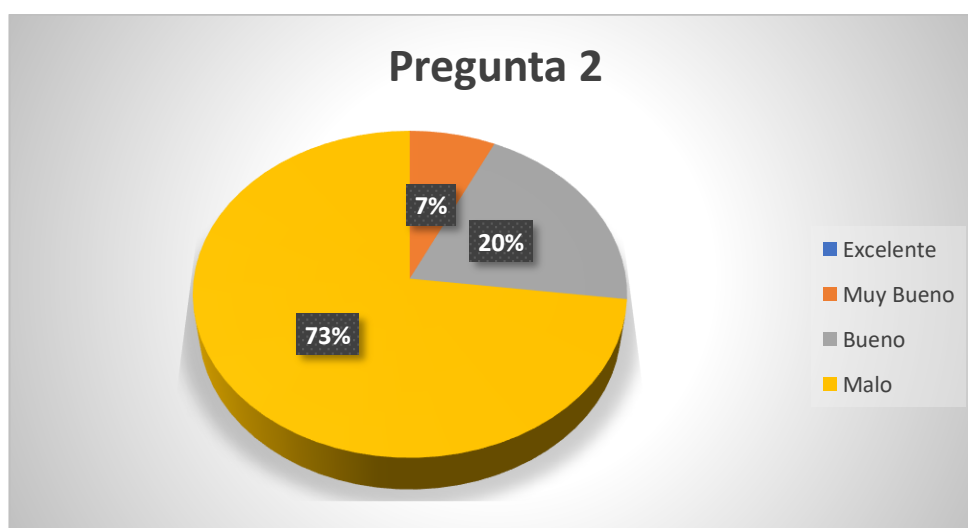
Tabla 4.1. Porcentaje de respuestas de la pregunta 1

Excelente	82%
Muy Bueno	16%
Bueno	2%
Malo	0%

Resultados de la pregunta 1. Elaborado por: Chanataxi Carlos, Pilca Henry

4.6.2. ¿Qué tan problemático es encender la bomba de agua de forma manual?

Figura 4.16. Pregunta 2 de la encuesta



Resultados de las opciones de la pregunta 2. Elaborado por: Chanataxi Carlos, Pilca Henry

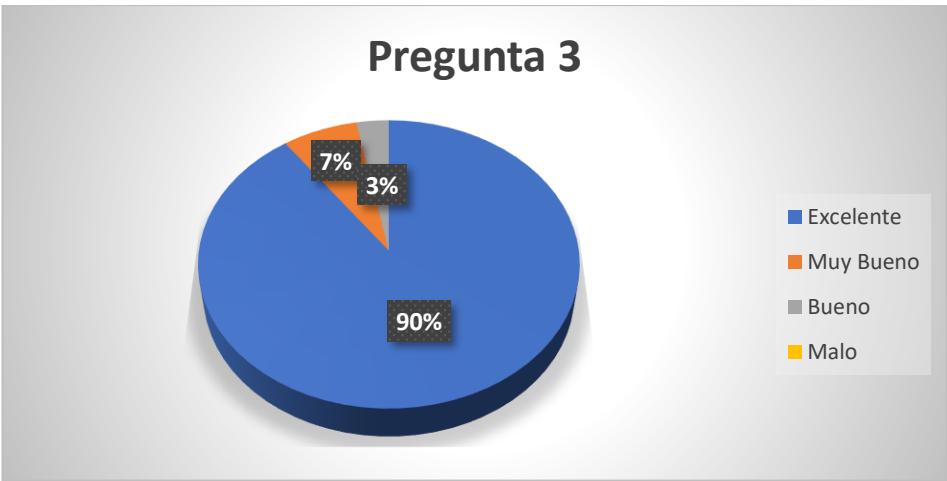
Tabla 4.2. Porcentaje de respuestas de la pregunta 2

Excelente	0%
Muy Bueno	7%
Bueno	20%
Malo	73%

Resultados de la pregunta 2. Elaborado por: Chanataxi Carlos, Pilca Henry

4.6.3. ¿Cómo calificaría el tamaño y espacio utilizado por el dispositivo IoT?

Figura 4.17. Pregunta 3 de la encuesta



Resultados de las opciones de la pregunta 3. Elaborado por: Chanataxi Carlos, Pilca Henry

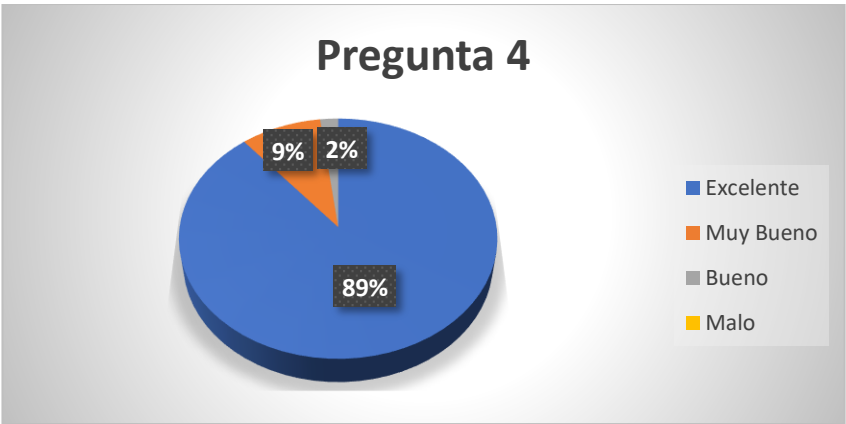
Tabla 4.3. Porcentaje de respuestas de la pregunta 3

Excelente	90%
Muy Bueno	7%
Bueno	3%
Malo	0%

Resultados de la pregunta 3. Elaborado por: Chanataxi Carlos, Pilca Henry

4.6.4. ¿Qué tan productivo le parecería poder controlar la calidad del agua por medio de una red social?

Figura 4.18. Pregunta 4 de la encuesta



Resultados de las opciones de la pregunta 4. Elaborado por: Chanataxi Carlos, Pilca Henry

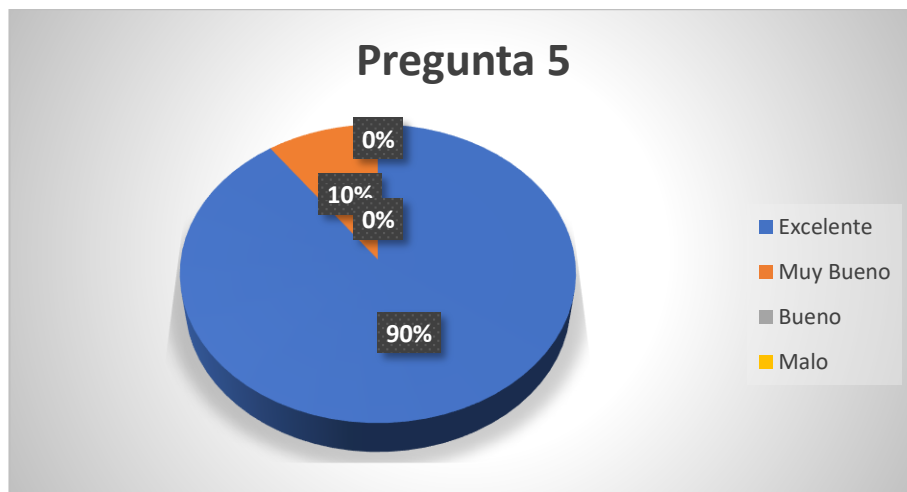
Tabla 4.4. Porcentaje de respuestas de la pregunta 4

Excelente	89%
Muy Bueno	9%
Bueno	2%
Malo	0%

Resultados de la pregunta 4. Elaborado por: Chanataxi Carlos, Pilca Henry

4.6.5. ¿Piensa que los comandos del dispositivo IoT son fáciles de utilizar?

Figura 4.19. Pregunta 5 de la encuesta



Resultados de las opciones de la pregunta 5. Elaborado por: Chanataxi Carlos, Pilca Henry

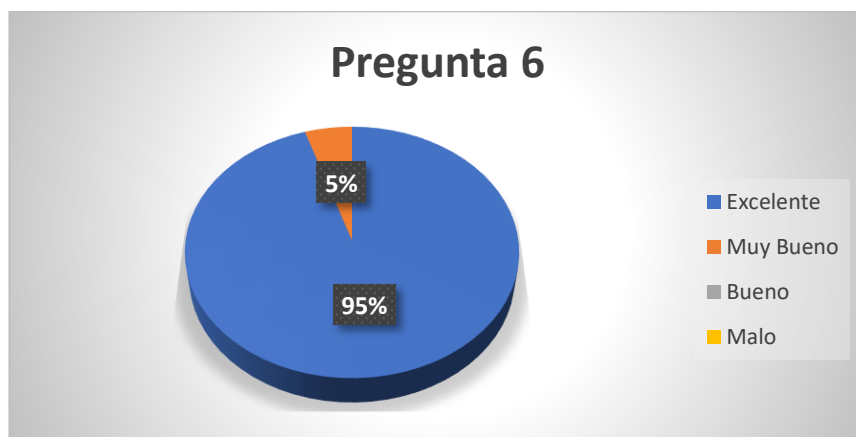
Tabla 4.5. Porcentaje de respuestas de la pregunta 5

Excelente	90%
Muy Bueno	10%
Bueno	0%
Malo	0%

Resultados de la pregunta 5. Elaborado por: Chanataxi Carlos, Pilca Henry

4.6.6. ¿Cómo calificaría la reducción de la tasa de mortalidad con el monitoreo IoT en la producción de truchas?

Figura 4.20. Pregunta 6 de la encuesta



Resultados de las opciones de la pregunta 6. Elaborado por: Chanataxi Carlos, Pilca Henry

Tabla 4.6. Porcentaje de respuestas de la pregunta 6

Excelente	95%
Muy Bueno	5%
Bueno	0%
Malo	0%

Resultados de la pregunta 6. Elaborado por: Chanataxi Carlos, Pilca Henry

La calificación general del dispositivo IoT es muy buena debido a que da solución a varios problemas que han ido aquejando a los productores en los criaderos de truchas y en específico al criadero de truchas La Merced. El hecho de que se pueda encender la electrobomba de forma remota es de gran ayuda para la persona encargada porque puede dedicarse a realizar otras actividades mientras sigue controlando la piscina de truchas de una forma fácil y accesible para la persona. Por otro lado, el tamaño que ocupa el dispositivo brinda un mayor confort al usuario ya que optimiza mucho espacio, y el control de la calidad del agua a través de una red social es muy llamativo para los usuarios ya que es lo que más se usa en la actualidad, brindando un servicio de calidad y asequible para cualquier persona que posea un Smartphone o un computador. La disminución de la tasa de mortalidad en las piscinas de truchas es la relevancia que ofrece el dispositivo después de todo, quedando los usuarios satisfechos y con la certeza de que el equipo cumple con sus expectativas para el monitoreo, porque se ofrece un dispositivo compacto, de control, manejable y de fácil uso.

CONCLUSIONES

Se analizaron diferentes sensores y dispositivos para la toma de muestras de la calidad de agua y se determinó que el NODEMCU-32S proporciona las suficientes entradas y salidas analógicas y digitales para trabajar con los respectivos elementos a utilizar, con respecto a los sensores el DS18B20 es el mejor para medir temperatura del agua ya que este permite realizar mediciones en líquidos y gases, de la misma manera se utilizó el sensor SEN0161 para detectar el nivel de PH debido a que este es el más preciso, además se empleó el sensor SEN0237 para ejecutar el sondeo de Oxígeno disuelto ya que su sonda galvánica proporciona un tiempo de respuesta elevado lo cual proporciona una estabilidad en la toma de datos almacenados en la nube.

Se diseñó un dispositivo IoT en Cloud con la capacidad de monitorear los parámetros del agua previa la obtención de datos por parte de los sensores de pH, temperatura y nivel de oxígeno disuelto para producir alarmas mediante la plataforma Thingspeak y la aplicación móvil Thingview las cuales permiten la supervisión de la calidad del agua de tal forma que se puedan enviar alertas por medio de la red social Telegram que permite realizar el control de una electrobomba la cual controla la calidad del agua en la piscina del criadero de truchas La Merced.

Se implementó un dispositivo IoT en cloud para monitorear y supervisar las condiciones adecuadas del agua que se ven afectados por los parámetros de pH que debe encontrarse en un rango específico de 6.5 a 8.5, el nivel de oxígeno disuelto puede mantenerse en el intervalo de 7.5 a 12 mg/L y la temperatura adecuada que se encuentre entre 13 a 18 °C, los datos proporcionados pueden visualizarse y controlarse de tal forma que se pueda mejorar la calidad de agua de la piscina de las truchas arco iris.

Se realizaron pruebas de latencia tanto al servidor como al dispositivo determinando un tiempo de 200 y 50 ms respectivamente para la transmisión de paquetes evitando la pérdida de los mismos en el transcurso del día, a su vez las pruebas de retardo determinaron un tiempo de 160ms desde el NODE MCU-32S hasta el servidor

ThingSpeak, en cuanto a la disponibilidad de la red del dispositivo IoT se obtuvo un 2% de pérdida de conectividad haciendo que el funcionamiento de este sea adecuado para los requerimientos del sistema.

RECOMENDACIONES

Se podría hacer una red IoT para que abarquen todas las piscinas de un criadero de truchas y que los datos se encuentren en una sola plataforma, haciendo más amigable la supervisión y control de la calidad del agua de todas las piscinas, y cuando estas sobrepasen los parámetros establecidos, automáticamente se enciendan los actuadores conectados en cada una de las piscinas.

Para que el sistema electrónico de las piscinas de truchas sea más robusto se podría implementar un control que, de acuerdo con una hora establecida deje caer la comida de las truchas y dependiendo de los tamaños que se tenga se deje caer una porción diaria haciendo uso de visión artificial.

BIBLIOGRAFÍA

- Alcocer, C. (14 de 09 de 2016). *Ingeniería de las Telecomunicaciones*. Obtenido de <https://blog.telecom.pucp.edu.pe/index.php/2016/09/14/cual-es-la-diferencia-entre-los-estandares-ieee-802-11ah-y-802-11af/>
- Angos, C. P. (2017). *Repositorio institucional UPB*. Obtenido de <https://repository.upb.edu.co/handle/20.500.11912/3111>
- Areny, R. P. (2003). Sensores y acondicionadores de señal. Barcelona: Marcombo S.A.
- Artero, C., Nogueras, M., & Mànuel, A. (2012). Ph Sensor. *Universitat Politècnica de Catalunya*, 1.
- BICS. (12 de 05 de 2019). *Global IoT*. Obtenido de <https://bics.com/es/services/conectividad-iot-global/>
- C. Lozoya, P. M. (2018). Estimación del retardo en una red WIFI: Análisis comparativo. 1.
- DMCA. (16 de 12 de 2020). *Apps on windows*. Obtenido de <https://appsonwindows.com/apk/218807/>
- Ealde. (30 de 03 de 2019). *Ealde Business School*. Obtenido de <https://www.ealde.es/cloud-computing-caracteristicas-funcionamiento/>
- Espressif. (15 de 01 de 2021). *ESP 32 Series*. Obtenido de https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf
- Google Play. (06 de 12 de 2020). Obtenido de https://play.google.com/store/apps/details?id=com.cinetica_tech.thingview&hl=es_EC
- Gupta, A. K., & Johari, R. (18 de 04 de 2019). *IEEE Xplore*. doi:10.1109/IOT-SIU.2019.8777342
- HOHENSEE, B. (2014). Introducción a Android Studio. Mexico: Babelcube Inc .
- Hossain, M., Khan, R., Noor, S. A., & Hasan, R. (2016). Jugo: A Generic Architecture for Composite Cloud as a Service. *IEEE Xplore*. doi:10.1109/CLOUD.2016.0112
- IFTTT. (23 de 11 de 2019). *IFTTT*. Obtenido de https://ifttt.com/maker_webhooks
- Inc, F. (28 de 08 de 2019). *IFTTT*. Obtenido de <https://platform.ifttt.com/docs>
- Jalloul, C. (24 de 07 de 2010). *Wika*. Obtenido de https://www.wika.ch/upload/TA_JUL10_Automatica_es_es_25921.pdf

- Linden Tibbets, J. T. (14 de 12 de 2010). *IFTTT*. Obtenido de https://ifttt.com/explore/welcome_to_ifttt
- MathWorks. (24 de 01 de 2021). *ThingSpeak*. Obtenido de <https://thingspeak.com/>
- NS, J. (2020). Connect Web Services to Microsoft Teams with Webhooks. doi:https://bibliotecas.ups.edu.ec:2582/10.1007/978-1-4842-6476-8_7
- Pérez, Á. A. (2013). Oxígeno Disuelto presentes en la Piscicultura bajo Condiciones de Estanque Artificial. *Scientia et Technica*. doi:<https://doi.org/10.22517/23447214.8781>
- Ristemi, I., Apostolova Trpkovska, M., & Cico, B. (2019). MyGitIssues Web Application as a Solution in Dealing with Issues on GitHub. *IEEE Explore*, 2.
- Stephen M. Rumble, D. O. (2011). It's Time for Low Latency. 1.
- Valdeolmillos, C. (06 de 11 de 2020). *Mcpro*. Obtenido de <https://www.muycomputerpro.com/2020/11/06/sector-iot-complicara-2021-opciones-conectividad>
- Vishwakarma, S. K., Upadhyaya, P., Kumari, B., & Mishra, A. K. (2019). Smart Energy Efficient Home Automation System Using IoT. *IEEE Xplore*. doi:10.1109/IoT-SIU.2019.8777607

ANEXOS

Anexo 1. Código de programación del dispositivo NODEMCU-32S

Anexo1.1 Código de inicialización de librerías

```
#include <OneWire.h>
#include <DallasTemperature.h>
#include <WiFiClientSecure.h>
#include "ThingSpeak.h"
#include "WiFi.h"

#include <UniversalTelegramBot.h>
```

Anexo 1.2 Código para la asignación de pines.

```
const int led12 = 4;
int estadoLed12 = 0;
int inicio = 1;
String chat_id;

// Pin donde se conecta el ADC
const int pinDatosDQ = 13;
const int pHpin = 35;

//Constantes Oxigeno
const uint16_t DO_Table[41] = {
    14460, 14220, 13820, 13440, 13090, 12740, 12420, 12110, 11810, 11530,
    11260, 11010, 10770, 10530, 10300, 10080, 9860, 9660, 9460, 9270,
    9080, 8900, 8730, 8570, 8410, 8250, 8110, 7960, 7820, 7690,
    7560, 7430, 7300, 7180, 7070, 6950, 6840, 6730, 6630, 6530, 6410};

uint8_t Temperaturet;
uint16_t ADC_Raw;
uint16_t ADC_Voltage;
uint16_t DO;

int16_t readDO(uint32_t voltage_mv, uint8_t temperature_c)
{
    #if TWO_POINT_CALIBRATION == 0
        uint16_t V_saturation = (uint32_t)CAL1_V + (uint32_t)35 * temperature_c - (uint32_t)CAL1_T * 35;
        return (voltage_mv * DO_Table[temperature_c] / V_saturation);
    #else
        uint16_t V_saturation = (int16_t)((int8_t)temperature_c - CAL2_T) * ((uint16_t)CAL1_V - CAL2_V) / ((uint8_t)CAL1_T - CAL2_T) + CAL2_V;
        return (voltage_mv * DO_Table[temperature_c] / V_saturation);
    #endif
}
```

1.3 Código para la verificación y autenticación de credenciales del sistema.

```
#include <UniversalTelegramBot.h>
//Token de Telegram BOT se obtienen desde Botfather en telegram
#define BOT_TOKEN "1864351047:AAEq-DxVpr6lmo9PSTSbtRB2hLF4lUc0HE0"

//ID_Chat se obtiene de telegram
#define ID_Chat "-576777743"

//Datos de Conexion Wifi
const char* ssid = "RAMIRO CHANATAXI CNT-EXT";
const char* password = "1707335640";

//Identificacion Thigspeak
unsigned long channelID = 1402753;
const char* WriteAPIKey = "6T68HDNNISVCGX3R";
```

1.4 Código para el llamado de la función temperatura.

```
OneWire oneWireObjeto(pinDatosDQ);
DallasTemperature sensorDS18B20(&oneWireObjeto);
void temperatura(){

    sensorDS18B20.requestTemperatures();
    Serial.print("Temperatura: "); // Leemos y mostramos los datos de los sensores DS18B20
    Serial.print(sensorDS18B20.getTempCByIndex(0));
    Serial.println(" C");

    //Envio de datos a ThingSpeak
    ThingSpeak.setField (2,sensorDS18B20.getTempCByIndex(0));
}
```

1.4 Código para el llamado de la función PH.

```
void PH(){
    float pH = (1023 - analogRead(pHpin)) / 73.07; //Lectura y transformación de valores de PH 0-14
    Serial.print("PH = ");
    Serial.println(pH);

    ThingSpeak.setField (1,pH);
}
```

1.5 Código para el llamado de la función oxígeno.

```
/**Variables Oxigeno**
#define DO_PIN 34

#define VREF 5000 //VREF (mv)
#define ADC_RES 1024 //ADC Resolution

//punto simple de calibracion Mode=0
//punto doble decalibracion Mode=1
#define TWO_POINT_CALIBRATION 0

#define READ_TEMP (25) //Temperatura actual del agua °C

//La calibración de un solo punto debe llenarse CAL1_V y CAL1_T
#define CAL1_V (1123) //mv
#define CAL1_T (20) //°C
//La calibración de dos puntos debe llenarse CAL2_V y CAL2_T
//CAL1 Punto de alta temperatura, CAL2 Punto de baja temperatura
#define CAL2_V (1118) //mv
#define CAL2_T (15) //°C
void oxigeno(){
    Temperaturet = (uint8_t)READ_TEMP;
    ADC_Raw = analogRead(DO_PIN);
    ADC_Voltage = uint32_t(VREF) * ADC_Raw / ADC_RES;
    float Ox=readDO(ADC_Voltage, Temperaturet)*0.003;

    Serial.print("Ox = ");
    Serial.println(Ox);

    //ThingSpeak.setField (3,readDO(ADC_Voltage, Temperaturet));
    ThingSpeak.setField (3,Ox);
}
```

Anexo 1.6 Código para el control de la bomba de agua por medio de la red social Telegram.

```
void mensajesNuevos(int numerosMensajes)
{
    for (int i = 0; i < numerosMensajes; i++)
    {
        String chat_id = bot.messages[i].chat_id;
        String text = bot.messages[i].text;

        //////////Luz 1 en el pin 12////////
        if (text == "Bombaoff")
        {
            digitalWrite(led12, LOW); //
            estadoLed12 = 1;
            bot.sendMessage(chat_id, "Bomba apagada", "");
        }

        if (text == "Bombaon")
        {
            estadoLed12 = 0;
            digitalWrite(led12, HIGH); //
            bot.sendMessage(chat_id, "Bomba encendida", "");
        }
        //////////Estado de la bomba //////////

        if (text == "Estado")
        {
            ///Estado luz 1///
            if (estadoLed12)
            {
                bot.sendMessage(chat_id, "Bomba apagada", "");
            }
            else
            {
                bot.sendMessage(chat_id, "Bomba encendida", "");
            }
        }
        if (text == "Ayuda")
        {
            String ayuda = "Bienvenido al sistema de control para el criadero de truchas La Merced, " ".\n";
            ayuda += "Estas son tus opciones.\n\n";
            ayuda += "Bombaon: para encender la bomba \n";
            ayuda += "Bombaoff: para apagar la bomba \n";
            ayuda += "Estado : devuelve el estado actual de la bomba\n";
            ayuda += "Ayuda: Imprime este menú \n";
            ayuda += "Recuerda el sistema distingue entre mayuculas y minusculas \n";
            bot.sendMessage(chat_id, ayuda, "");
        }
    }
}
```

Anexo 1.7 Código para verificar la conectividad WiFi y envío de datos al servidor Thingspeak.

```
//Verificacion de conexion Wifi
while (WiFi.status() != WL_CONNECTED){
    delay(500);
    Serial.print(".");
}
Serial.print("WiFi Conectado");
Serial.println(WiFi.localIP());
//Inicializacion libreria ThingSpeak
ThingSpeak.begin(Client);

Serial.println(WiFi.localIP());
.f(inicio == 1){
Serial.println("Sistema preparado");
//Enviamos un mensaje a telegram para informar que el sistema está listo
bot.sendMessage(ID_Chat, "Sistema preparado!!!, escribe Ayuda para ver las opciones", "");
inicio = 0;
```

1.8 Código de verificación para el envío y recepción de comandos por parte del dispositivo IoT y Telegram.

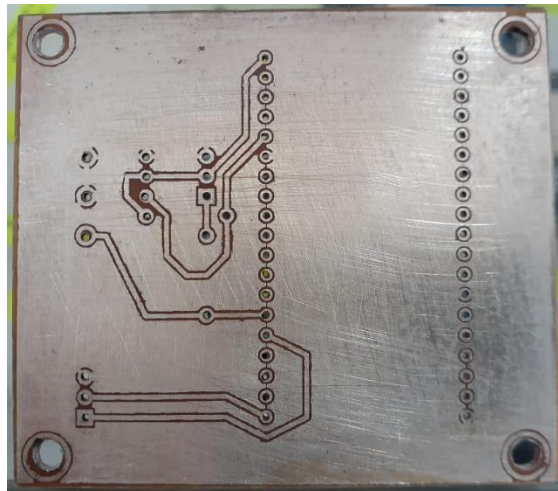
```
ThingSpeak.writeFields(channelID,WriteAPIKey);
Serial.println("Datos enviados a ThingSpeak");
delay(14000);

//Verifica si hay datos nuevos en telegram cada 1 segundo
if (millis() - tiempoAnterior > tiempo)
{
    int numerosMensajes = bot.getUpdates(bot.last_message_received + 1);

    while (numerosMensajes)
    {
        Serial.println("Comando recibido");
        mensajesNuevos(numerosMensajes);
        numerosMensajes = bot.getUpdates(bot.last_message_received + 1);
    }

    tiempoAnterior = millis();
```

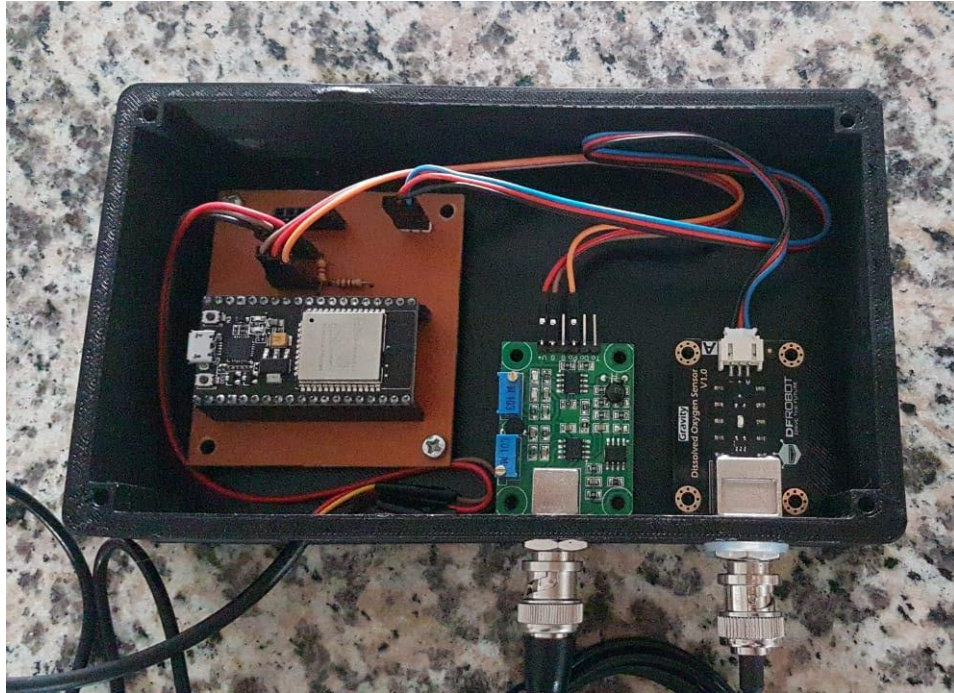

Anexo 2. PCB impresa en cobre



Anexo 3. Impresión 3D de la caja y tapa para el dispositivo



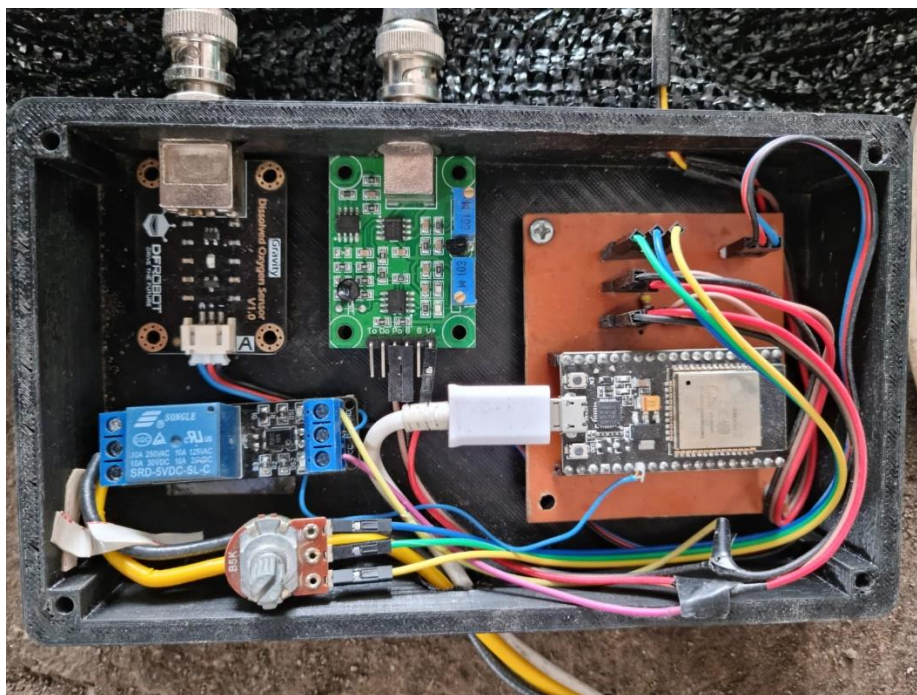
Anexo 4. Conexiones de los módulos de pH, oxígeno disuelto y temperatura con el NODEMCU-32S



Anexo 4. Dispositivo IoT con sus respectivos sensores



Anexo 5. Implementación del dispositivo IoT



Anexo 6. Ubicación de sensores en la piscina del criadero de truchas La Merced

