UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y COMUNICACIONES

TEMA:

"SISTEMA DE CONTROL ELECTRÓNICO PARA ACUARIOS UTILIZANDO TECNOLOGÍAS GSM Y VOIP"

Proyecto de Trabajo de Graduación. Modalidad: Proyecto de Investigación, presentado previo la obtención del título de Ingeniero en Electrónica y Comunicaciones.

LÍNEA DE INVESTIGACIÓN: Sistemas de Control

AUTOR: Jessica Andrea Vivanco Correa

TUTOR: Ing. César Granizo, Mg.

Ambato - Ecuador

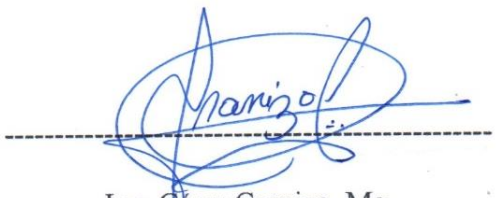
Septiembre 2017

APROBACIÓN DEL TUTOR

En mi calidad de tutor de investigación sobre el tema: "SISTEMA DE CONTROL ELECTRÓNICO PARA ACUARIOS UTILIZANDO TECNOLOGÍAS GSM Y VOIP" de la señorita Jessica Andrea Vivanco Correa, estudiante de la Carrera de Ingeniería Electrónica y Comunicaciones de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato, considero que el informe de investigación reúne los requisitos suficientes para que continúe con los trámites y consiguiente aprobación de conformidad con el numeral 7.2 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ambato Septiembre, 2017

EL TUTOR



Ing. César Granizo, Mg.

AUTORÍA

El presente trabajo de investigación titulado "SISTEMA DE CONTROL ELECTRÓNICO PARA ACUARIOS UTILIZANDO TECNOLOGÍAS GSM Y VOIP" es absolutamente original, auténtico y personal en tal virtud, el contenido, efectos legales y académicas que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, Septiembre 2017

Jessica Andrea Vivanco Correa

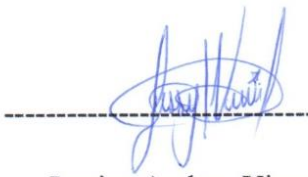
C.C: 1719383943

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación.

Cedo los derechos de mi Trabajo de Titulación, con fines de difusión pública, además autorizo su reproducción dentro de las regulaciones de la Universidad

Ambato, Septiembre 2017



Jessica Andrea Vivanco Correa

C.C: 1719383943

APROBACIÓN DE LA COMISIÓN CALIFICADORA

La Comisión Calificadora del presente trabajo conformada por los señores docentes Ing. Marco Jurado, Mg. e Ing. Geovanni Brito, Mg. revisó y aprobó el Informe Final del trabajo de graduación titulado "SISTEMA DE CONTROL ELECTRÓNICO PARA ACUARIOS UTILIZANDO TECNOLOGÍAS GSM Y VOIP", presentado por la señorita Jessica Andrea Vivanco Correa de acuerdo al numeral 9.1 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ing. Mg. Elsa Pilar Urrutia Urrutia

PRESIDENTA DEL TRIBUNAL

Ing. Marco Jurado

DOCENTE CALIFICADOR

Ing. Geovanni Brito

DOCENTE CALIFICADOR

DEDICATORIA

A Dios que cada día de vida me demuestra lo bondadoso que ha sido conmigo, llenándome de bendiciones.

A mis padres Zoila y Sixto que siempre me han apoyado, brindándome su cariño y comprensión, guiándome para ser una persona de bien.

A mis hermanos Catherine y Bryan que han estado a mi lado en los momentos más difíciles brindándome consejos y palabras de aliento.

A mi Abuelita Regina que fue como mi segunda madre y que desde el cielo me ha guiado.

A mis tíos Irene y José Luis que me brindaron su apoyo durante toda mi carrera Universitaria, haciéndome sentir como una hija más.

A José Julián que me ha brindado su apoyo incondicional para poder culminar con esta meta y me impulsa a seguir adelante.

Jessica Andrea

AGRADECIMIENTO

A Dios por brindarme la vida y regalarme a la mejor familia, por ser mi guía y fortaleza.

A mi familia que siempre me brindan su amor, comprensión y me han dado el mejor ejemplo inculcándome valores para ser una persona de bien.

Al Acuario Serpentario "San Martín" por su apoyo y colaboración en cada etapa de este proyecto.

A la Facultad de Ingeniería en Sistemas, Electrónica e Industrial por brindarme cada uno de los conocimientos para alcanzar esta meta tan anhelada.

Jessica Andrea

ÍNDICE DE CONTENIDO

Contenido **Página**

CAPÍTULO I

EL PROBLEMA DE INVESTIGACIÓN

CAPÍTULO II

MARCO TEÓRICO

CAPÍTULO III

METODOLOGÍA

CAPÍTULO IV

DESARROLLO DE LA PROPUESTA

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

En la presente investigación se detalla el desarrollo de un prototipo de Control Electrónico para acuarios utilizando tecnología GSM y VoIP, implementado en el Acuario Serpentario "San Martín" del Cantón de Baños, el cual tiene como objetivo controlar las variables físicas en un ambiente confinado.

Para el control del acuario se colocó un sensor de temperatura y humedad ambiental, un sensor de temperatura del agua, un sensor de luz y un sensor de pH, que adquieren los datos proporcionados por el ambiente, y el sistema determina si la temperatura del agua es menor a 28°C activa un calentador de agua y si la temperatura es superior a 32°C apaga el calentador, si la iluminación es menor a 25 lux entre las 7h00 y 18h00 se enciende una lámpara fluorescente, si el nivel de pH no se encuentra entre 6.5 y 7.5 se enciende una lámpara que indica que el agua debe ser cambiada y finalmente si la temperatura ambiental es superior a 25°C se activa un ventilador, la información obtenida de los sensores se almacena en una base de datos a la cual se puede acceder por medio de una llamada para verificar el estado del sistema.

La metodología del diseño está enfocada en realizar un control periódico de la temperatura ambiental, calidad del agua, cantidad de luz y temperatura del agua para el buen desarrollo y reproducción de los peces.

Cabe destacar que es importante el diseño e implementación del prototipo en el Acuario Serpentario "San Martín" de Baños, porque permite el control periódico de las variables físicas que intervienen en el desarrollo de los peces. Por esta razón los beneficiarios directos del presente proyecto de investigación son los propietarios del Acuario Serpentario "San Martín".

Palabras claves: Acuario, GSM, VoIP, Sistema de Control.

ABSTRACT

In the present investigation is detailed the development of a prototype of Electronic Control for aquariums using GSM and VoIP technology, implemented in the "San Martín" Serpentary Aquarium of the Canton of Baños, is aimed at controlling the physical variables of a confined environment.

For the control of the aquarium, were installed a temperature and humidity sensor, a water temperature sensor, a light sensor and a pH sensor, which acquire the data provided by the environment, and the system determines if the temperature of the water is less than 28 °C activates a water heater, and if the temperature is above 32 ° C turn off the heater, if the illumination is less than 25 lux between 7h00 and 18h00 a fluorescent lamp is turned on, if the pH level is not between 6.5 and 7.5 a lamp is lit indicating that the water must be changed and finally if the ambient temperature is higher than 25° C a ventilador is activated, the information obtained from the sensors is stored in a database which can be accessed by means of a call to check the status of the system.

The design methodology is focused on periodic control of the environmental temperature, water quality, amount of light and water temperature for the good development and reproduction of the fish.

It is important to highlight the design and implementation of the prototype in the "San Martin" Serpentary Aquarium, because it allows the control of the periodicity of the physical variables involved in the development of the fish. For this reason, the direct beneficiaries of this investigation project are the owners of the "San Martin" Serpentary Aquarium.

Keywords: Aquarium, GSM, VoIP, Control System.

ACRÓNIMOS

AGI: Asterisk Gateway Interface

BSC: Base Station Controller

BSS: Base Station Subssystem

BTS: Base Transceiver Station

CAO: Control de Acuarios por Ordenador

GSM: Sistema Global para las Comunicaciones Móviles

GPIO: General Purpose Input Output

IoT: Internet of things

IP: Internet Protocol

LX: Lux

MS: Mobile Station

MSC: Mobile Switching Center

NSS: Network Switching Subsystem

OSS: Operation and Service Subsystem

SCADA: Supervisory Control and Data Adquisition.

SE: Sistemas Embebidos

SIP: Session Initiation Protocol

VB: Visual Basic.

VoBB: Voice over Broad Band

VoIP: Voice over IP

INTRODUCCIÓN

La implementación del prototipo para el Control Electrónico del Acuario Serpentario "San Martín" del Cantón de Baños, tiene como propósito entregar a los dueños del acuario una herramienta electrónica que permite monitorear los parámetros principales del hábitat donde se desarrollan los peces y de esta manera facilitar el proceso de mantenimiento del acuario. El proyecto de investigación se encuentra estructurado de la siguiente manera:

En el Capítulo I, se detalla la problemática que posee el Acuario Serpentario "San Martín" del Cantón de Baños de la provincia de Tungurahua, además de la justificación junto con los objetivos propuestos para dar solución al presente proyecto de investigación.

En el Capítulo II, se desarrolla el marco teórico en el cual se describen los principales componentes utilizados en la implementación del Sistema de Control para Acuarios, además se realiza un análisis de las tecnologías y sensores para determinar la mejor alternativa al momento de la implementación del sistema.

En el Capítulo III, se especifica las diferentes técnicas de investigación que fueron desarrolladas para la obtención de información y se describe de manera general las etapas para el desarrollo de la presente investigación.

En el Capítulo IV, se describe el proceso mediante el cual se recolectó la información necesaria para el desarrollo del proyecto, describiendo paso a paso las etapas para el desarrollo de la investigación.

En el Capítulo V, se muestran las conclusiones, recomendaciones y anexos obtenidos durante el desarrollo e implementación del prototipo para el Control Electrónico del Acuario Serpentario "San Martín" de Baños.

CAPÍTULO I

EL PROBLEMA DE INVESTIGACIÓN

1.1 Tema de Investigación:

SISTEMA DE CONTROL ELECTRÓNICO PARA ACUARIOS UTILIZANDO TECNOLOGÍAS GSM Y VOIP.

1.2 Planteamiento del Problema:

El World Association of Zoos and Aquariums (WAZA), señala que la temperatura en los acuarios debe ser controlada dependiendo del tipo de especies con las que se cuente en el acuario, un calentador de agua permite estabilizar las temperaturas apropiadas para que las especies se desarrollen en un ambiente climático favorable para su crecimiento y nutrición. El control de temperatura es vital en un acuario por que la falta o el exceso del mismo, conlleva a contraer enfermedades a causa del estrés, la reproducción de ciertas especies también se ve afectado por la falta de regulación de la temperatura. [1]

La Asociación Latinoamericana de Parques Zoológicos y Acuarios (ALPZA), detalla que la alimentación es el pilar fundamental para que las especies conserven su salud y su vitalidad. La acumulación del alimento en las peceras contaminan el agua, desarrollando bacterias y hongos, convirtiéndose en uno de los problemas más comunes que tienen que enfrentar los peces en un acuario, esto se puede observar directamente en la pigmentación de los peces, con la aparición de manchas de color gris o blanco, en algunos casos ocasionando que se pudran las aletas y produciendo la muerte de los peces, un análisis visual del color de las especies acuáticas, indican el estado de salud que pueden estar presentando. [2]

En el Ecuador existe una variedad de acuarios en los cuales la calidad de agua es vital para la convivencia de las especies, el agua tiende a oxidarse con mayor facilidad en acuarios, de la calidad del agua depende la salud y vitalidad de las especies, el control

continuo del cambio de agua se puede realizar en base a los niveles del potencial de hidrógeno, nivel de agua y nivel de oxígeno. [3] [4]

El tratamiento de agua en los acuarios es muy importante, al momento de realizar la limpieza y al cambiar el agua se recomienda dejar reposar la misma en un período de 15 a 20 horas como mínimo para purificar la misma, desechando el cloro que afecta directamente a los peces, ya que es perjudicial para las branquias, otras membranas y tejidos sensibles de los peces. [5]

1.3 Delimitación:

1.3.1 Delimitación de contenidos:

- Área académica: Comunicaciones.
- Línea de investigación: Tecnologías de la Información y de la Comunicación.

1.3.2. Delimitación espacial: El proyecto de investigación se desarrolló en el Acuario Serpentario "San Martín", de la ciudad de Baños de la provincia de Tungurahua.

1.3.3. Delimitación temporal: El presente proyecto de investigación se desarrolló en el período Marzo 2017 – Septiembre 2017 de acuerdo a lo establecido en el Reglamento de Graduación para Obtener el Título Terminal de Tercer Nivel de la Universidad Técnica de Ambato.

1.4 Justificación:

En los acuarios turísticos es de vital importancia la estética de las peceras, para poder visualizar claramente las especies de peces. De la transparencia de las peceras dependerá el agrado de los turistas para recomendar y volver a visitar el acuario. La calidad de agua, refleja un ambiente limpio de impurezas, evitando que se formen bacterias y hongos que atenten contra la vida de los peces, es por ello que para el cambio de agua en el acuario "San Martín" se realiza un control visual de la pureza del agua, para determinar si ésta debe ser cambiada, por esta razón con la implementación del presente proyecto en el acuario se logra tener un control periódico de la calidad del agua.

Al no contar con un sistema de control electrónico en el acuario "San Martín", se debe revisar diariamente la variación de la temperatura para verificar que se encuentre en los rangos establecidos, debido a que el nivel de temperatura varía según las especies

de peces, esta variación debe ser controlada periódicamente porque de no hacerlo, ocasionaría enfermedades que pueden ser visualizadas con la aparición de ciertas manchas en los peces, también serían propensos a verse afectados en su reproducción y en un caso extremo ocasionando la muerte de los peces.

Los beneficiarios directos de la implementación del presente proyecto son los dueños del acuario porque se ahorran tiempo, al no tener que revisar periódicamente parámetros como: la cantidad de luz, la temperatura ambiental, la calidad y temperatura del agua. Los beneficiarios indirectos son los turistas porque la estética del acuario crea un ambiente agradable y se puede disfrutar con mayor agrado las diferentes especies de peces.

Los propietarios del Acuario Serpentario "San Martín" de la ciudad de Baños se encuentran gratificados con el desarrollo del presente proyecto por los beneficios que conllevó la implementación del mismo, como es el control remoto de las peceras, ahorrándoles tiempo y dinero, es por esto que hace que el proyecto sea altamente factible en su desarrollo e implementación.

Este proyecto es factible debido a la compatibilidad de los equipos a utilizar y la facilidad del lenguaje de programación para el control de la pecera, con la utilización de tecnologías de hardware libre el proyecto resulta factible debido a los bajos costos en la implementación.

Es por ello que la implementación del presente proyecto resulta muy útil cuando se requiera un control remoto de las peceras o cuando los dueños deseen cerrar el acuario, sin tener la preocupación del control de temperatura, humedad, o iluminación en el acuario, siendo los beneficiarios directos los dueños del acuario que con una sola llamada se informarían del estado del sistema.

El desarrollo del presente proyecto no posee limitantes porque puede ser implementado en cualquier acuario, brindando la factibilidad del control y monitoreo continuo para el cuidado de los peces.

1.5 Objetivos

1.5.1 Objetivo General:

- Implementar un sistema de control electrónico para acuarios utilizando tecnologías GSM y VoIP.

1.5.2 Objetivos Específicos

- Analizar las variables físicas que intervienen en el comportamiento de los peces en un ambiente controlado.
- Determinar el tipo de tecnología que se utilizan para el monitoreo remoto de una pecera en el Acuario Serpentario ‘‘San Martín’’.
- Diseñar el prototipo de sistema de monitoreo del ambiente confinado de una pecera en el acuario ‘‘San Martín’’.

CAPÍTULO II

MARCO TEÓRICO

2. 1 Antecedentes Investigativos

Una vez realizada la investigación bibliográfica en la Universidad Técnica de Ambato y en los repositorios de las diferentes Universidades, se han encontrado algunos trabajos similares al planteado en este proyecto de investigación.

Víctor Lapuente Solórzano, realizó el control y monitorización de un acuario en tiempo real mediante tecnología Open Source, el objeto del proyecto es el desarrollo de un sistema electrónico basado en el proyecto Open Aquarium que integre a la plataforma Raspberry Pi, para poder monitorizar y controlar la información necesaria, recibida por diferentes sensores, para la preservación de acuarios. A través de Raspberry el usuario pudo interactuar con Open Aquarium, pudiendo visualizar los datos de los sensores y utilizar los actuadores, a través de una pantalla táctil, y a su vez, se utilizó para que en una aplicación web el usuario pueda visualizar los mismos datos y un streaming del acuario. Como resultado de este proyecto se ha obtenido un sistema electrónico de bajo coste, que sustituye a los caros dispositivos utilizados en el sector de la automatización de acuarios, permitiendo al usuario tener un control total sobre su acuario de un modo sencillo y eficaz. 2015, (p.3). [6]

Virgilio Valencia, desarrolló un sistema de control de acuarios, el sistema permitió al usuario automatizar los parámetros de los que depende el hábitat de los peces, realizando el control de las condiciones óptimas del acuario, para lo cual se implementó un módulo, desde el cual se puede visualizar y manipular los procesos fundamentales necesarios para obtener un hábitat saludable para los peces, a través de la tarjeta principal de control "Arduino Mega 2560" se controlarán los diferentes módulos como la alimentación, iluminación, bombas de circulación, temperatura y

suministro de energía de emergencia, de acuerdo a la información que se almacene de cada uno de los sensores utilizados en el proyecto, como resultado se ha obtenido que los módulos de control han permitido realizar las acciones vitales del sistema como son: ciclado de agua, control de iluminación, temperatura y acidez, permitiendo monitorear en tiempo real el estado de las variables principales del acuario. (p.136). [7]

Néstor de Juan Vázquez, realizó el Control, Adquisición y Supervisión de datos, para control de acuarios mediante arduino y VB (Visual Basic), el aplicar un control electrónico a un acuario facilita su mantenimiento y preservación de especies acuáticas, mediante el empleo de una tarjeta de adquisición de datos y un software SCADA, el usuario tiene el control del acuario, de esta manera puede informarse sobre las variables como la temperatura, acidez, iluminación y alimentación, con el objetivo de permitirle al usuario tomar la decisión correcta. SCADA está basado en un entorno web que hace uso de la tecnología IP, es decir permite la comunicación a través de la red, obteniendo como resultado el control remoto del acuario. (p.6). [8]

Almeida Roberto, realizó el diseño e implementación de un Sistema Automatizado de Control de cambio de agua y mantenimiento de acuarios medianos y pequeños, en el que se implementó un sistema de control automático para el mantenimiento y manejo de la calidad del agua y hábitat de un acuario, la interacción entre el sistema de control automático y el dueño del acuario, se desarrolló por medio de botones y un display de visualización, que permite tener un control remoto en tiempo real del estado de las variables físicas que intervienen en el acuario, como es el potencial de hidrógeno, el oxígeno disuelto, la temperatura y alimentación. El sistema automatizado de control de cambio de agua, se lo realizó mediante la tecnología de hardware libre como es arduino y raspberry pi, mediante las pruebas realizadas en el desarrollo del proyecto se logró obtener un correcto funcionamiento del control y manejo del acuario. (p. 8). [9]

Patricio Bos, desarrolló un Sistema de Control de Acuario con la plataforma CIAA (Computadora Industrial Abierta Argentina), con la implementación de un firmware que permite controlar un acuario en forma remota mediante una interfaz web embebida en la plataforma CIAA-NXP, utilizando el Sistema Operativo de Tiempo Real freeRTOS y el stack TCP/IP, con lo cual se busca implementar una interfaz que

permita visualizar la información más importante sobre el estado del sistema, y de esta manera se logre actuar sobre las variables que no se encuentran en los rangos establecidos, brindando un sistema confiable al usuario. Se logró obtener una interfaz más amigable que permite el manejo controlado del sistema, logrando obtener conocimientos sobre la arquitectura ARM Cortex para la programación de la plataforma CIAA-NXP. (p. 15). [10]

Antonio Castro Snurmacher, señala que CAO1 es un software para el sistema de Control de Acuarios por Ordenador, basado en la plataforma de hardware libre como es arduino. En él se pretende acercar el desarrollo de un controlador de acuario, por medio de un arduino Mega 2560, la aplicación terminada de CAO1 consta de 22 módulos y aproximadamente 6000 líneas de código, fue desarrollado para las personas que podrían tener algunas limitaciones técnicas, con el fin de controlar sus acuarios con un sistema computarizado propio y personalizado. Dando como resultado un sistema de control fácil de manejar desde cualquier ordenador, y principalmente con un costo mínimo por la utilización de tecnologías de hardware libre como lo es arduino. (p. 10). [11]

César Cusi y Fernando Sánchez, realizaron el estudio y diseño de un prototipo para el monitoreo de acuarios utilizando tecnología WIFI (IEEE 802.11b/g/n) enfocado al IoT (Internet of things), mediante una plataforma raspberry pi y el sistema operativo Android, para la implementación del sistema de control se utilizó una comunicación entre la plataforma raspberry pi y arduino, para el monitoreo del acuario, se utilizó sensores de conductividad eléctrica, temperatura ambiental, luz y temperatura del agua, la información proporcionada por los sensores fue almacenada en una base de datos, para la obtención de la información se desarrolló una aplicación Android con App Inventor, que permite visualizar el estado del sistema por medio de un dispositivo que tenga acceso a la red, el prototipo realiza acciones automáticamente cuando detecta parámetros fuera de rango, el desarrollo del proyecto resulta beneficioso porque se puede realizar el control y adquisición de información desde cualquier lugar por medio de internet, siendo un proyecto rentable por la utilización de software libre y el fácil manejo de la aplicación desarrollada con App Inventor. (p. 19). [12]

2.2 Fundamentación Teórica

2.2.1 Acuariofilia

La acuariofilia es la afición a la cría de peces y otros organismos acuáticos en un acuario, bajo condiciones controladas de temperatura, cantidad de luz, y calidad del agua. Esta actividad ha evolucionado notablemente a lo largo de los años, desde la cría de carpas doradas en estanques hasta los sofisticados ecosistemas acuáticos. [13]

De acuerdo al artículo del Laboratorio de Investigación de Recursos Acuáticos, se estima que alrededor del mundo existen aproximadamente 2 millones de personas que cuentan con acuarios marinos, estos acuarios se encuentran ubicados en puntos estratégicos para la preservación y conservación de especies, la mayor cantidad de acuarios se encuentran en zonas tropicales y subtropicales. Según el artículo "From ocean to aquarium" de Wabnitz,C, estima que en los acuarios se generan ganancias sobre $300 millones de dólares anualmente, por lo que considera a la acuariofilia es una actividad económicamente rentable.

Una gran cantidad de peces son exhibidos en los acuarios por sus vistosos colores y pigmentación es por ello que al momento de adquirir peces se busca de preferencia peces machos jóvenes debido a que presentan mayor colorido que las hembras. [14]

El Acuario y sus efectos tranquilizadores

El hombre moderno de ciudad sufre de la ausencia de lugares con vegetación para distraerse y esto causa estrés, fatiga y ansiedad añadiendo a esto la soledad, es decir la falta de comunicación por distintas razones como puede ser falta de tiempo o espacio reducido, he ahí nace la relación entre el ser humano y los animales, la atracción por animales depende de necesidades fisiológicas, que ayudan con el equilibrio afectivo de las personas.

En este sentido, los acuarios por sus efectos tranquilizantes llevan a relajarse y a encontrar un equilibrio satisfactorio, para que las personas puedan desestresarse y disfrutar del lugar por la belleza que representa el mismo y la tranquilidad que brinda a las personas. [15]

2.2.2 Componentes de un acuario

Los principales componentes de un acuario son los siguientes:

- **Recipiente:** El recipiente generalmente es de forma rectangular y de vidrio para la visualización de las diferentes especies de peces.
- **Calentador de agua:** El calentador de agua es un aparato eléctrico que es sumergido totalmente en la pecera y tiene la función de mantener estable la temperatura del agua.
- **Termómetro:** El termómetro sirve para el control periódico de temperatura en la pecera.
- **Luz:** La luz es indispensable en un acuario para poder brindar un ciclo biológico natural día-noche, se utiliza lámparas fluorescentes de 25 watt, para una mejor estética del acuario y para una preservación de peces y plantas.
- **Aireador:** El aireador es una bomba eléctrica de diafragma que permite oxigenar el agua y provocar un vacío dentro del filtro para atraer impurezas.
- **Filtro:** El filtro contiene carbón activado y algodón para filtrar impurezas.
- **Plantas:** Las plantas son un componente opcional, pueden ser plantas naturales o artificiales
- **Alimentos:** Los alimentos deben ser suministrados en cantidades moderadas para la preservación y reproducción de los peces.
- **Sustratos de fondo:** Los sustratos de fondo pueden ser piedras de colores o conchas pequeñas.
- **Especímenes:** Los especímenes son los habitantes del acuario, como por ejemplo los diferentes tipos de peces. [16]

2.2.3 Sistema Global para las Comunicaciones Móviles

El Sistema Global para las Comunicaciones Móviles (GSM), es un estándar para definir medios de comunicación móvil como mensajería de texto y llamadas telefónicas, de todas las tecnologías celulares usadas en la actualidad GSM es la más extendida.

GSM se crea para dar solución a los problemas de los sistemas móviles celulares que los habían creado descoordinados e incompatibles en muchos países del mundo.

Con la telefonía GSM cualquier usuario puede llamar o ser llamado en cualquier lugar dentro del área de cobertura internacional.

Ventajas de la red GSM

Las ventajas de la red GSM son las siguientes:

- **Amplia cobertura.-** Tiene una amplia cobertura por su uso generalizado en todo el mundo.
- **Gran variedad de teléfonos.-** Existe una gran variedad de teléfonos que operan en GSM, debido a la gran cobertura que brinda el sistema global para comunicaciones móviles.
- **Sin cargos de roaming.-** Los usuarios no pagan una tarifa de roaming para las llamadas internacionales.

Desventajas de la red GSM

Las desventajas de la red GSM son las siguientes:

- **Retraso de ancho de banda.-** Uno de los mayores inconvenientes de la red GSM, es que múltiples usuarios comparten el mismo ancho de banda.
- **Interferencia electrónica.-** Se producen interferencias electrónicas, debido a que GSM utiliza una tecnología de pulso de transmisión, es por ello que algunos lugares como hospitales y aviones, requieren que los teléfonos celulares se encuentren apagados.

Estructura del Sistema GSM

La estructura básica del sistema GSM está organizada como una red de células radioeléctricas cubriendo totalmente el área de servicio, dichas células se encuentran en una BTS (Base Transceiver Station) que trabajan con distintos canales de radio distribuidas según su plan celular. Una estación base BTS (Base Transceiver Station) se conecta a una estación base BSC (Base Station Controller) y esta maneja la red de radio en su totalidad; la estación base BSC (Base Station Controller) se conecta a una central de conmutación móvil MSC (Mobile Switching Center). Este es el corazón de la red GSM como responsable de la inicialización, enrutamiento, control y finalización de las llamadas. [17]

2.2.4 GoIP

El GoIP es un Gateway GSM que tiene como objetivo principal comunicar la red de teléfono móvil con la red de telefonía VoIP, el GoIP permite instalar varias líneas celulares, para realizar llamadas de manera centralizada desde un servidor de Voz IP.

Figura 2.1: GoIP.
Fuente: Fotografía tomada por el investigador.

En el GoIP se inserta una tarjeta sim y se procede a conectar los cables de red para realizar las configuraciones correspondientes.

Por defecto viene con la dirección IP 192.168.8.1, con el nombre usuario "admin" y la contraseña "admin", una vez ingresado al GoIP se puede colocar una dirección IP estática para desconectar el cable de red y tener una mejor movilidad.

En la parte frontal consta de niveles de señalización para verificar el funcionamiento del GoIP, entre los cuales consta: Power, RUN, LAN, PC, Channel. [18]

El GoIP, está diseñado con las siguientes especificaciones:

- Salida de red IP (LAN)
- Una entrada para tarjeta SIM. (CARD)
- Alimentación DC 12V. (Anexo A-literal d).

Figura 2.2: Entradas y Salidas del GoIP
Fuente: Fotografía tomada por el investigador.

2.2.5 Asterisk

Asterisk es un líder mundial en plataformas de telefonía de código abierto, es un software que puede convertir un ordenador de propósito general en un sofisticado servidor de comunicaciones VoIP. Asterisk es utilizado por empresas pequeñas, medianas y grandes, centros de llamadas, transportistas y agencias del gobierno a nivel mundial.

Asterisk Gateway Interface (AGI)

El Asterisk Gateway Interface, abreviado como AGI, es una interfaz del sistema de Asterisk, que permite desarrollar aplicaciones externas mediante diferentes lenguajes de programación como Perl, Php, C y Bourne Shell.

Es por ello que estas aplicaciones pueden estar escritas en distintos lenguajes de programación por ejemplo librerías en php que simplifican su escritura, haciendo modificaciones a base de datos, consulta de variables, controlar el dialplan mediante una llamada desde el archivo extensions.conf de Asterisk. [19]

2.2.6 VoIP

El protocolo de Voz sobre IP es una tecnología desarrollada para poder realizar comunicaciones de voz en tiempo real a través de redes IP, en un principio desarrolladas para el transporte de datos. Voz sobre IP, se refiere a la transmisión del tráfico de voz, sobre redes basadas en internet, en lugar de las redes telefónicas tradicionales.

Telefonía IP

Se denomina telefonía IP a la comunicación hablada entre dos o más personas distantes entre sí, utilizando una red pública que facilita el intercambio de la voz.

Telefonía IP, telefonía sobre internet, voz sobre banda ancha ó VoBB y la voz sobre IP ó VoIP tienen un mismo significado, ya que es el servicio que permite la transmisión de la voz utilizando internet.

VoIP se utiliza para facilitar la comunicación interna ya sea dentro de una institución o de una empresa, reemplazando al sistema tradicional de centrales. [20]

Transmisión de la voz

La comunicación de la voz, a través de una red IP consiste en los siguientes pasos:

- Hablar por un micrófono para captar las ondas sonoras.
- Se realiza el proceso de digitalización y codificación de la señal, para convertirla en un flujo de bits.
- Transmitir la información en tiempo real.
- Se realiza el proceso de decodificación.

- Conversión de la señal de digital a análogo
- Producir las ondas sonoras mediante un altavoz.

Figura 2.3: Transmisión de la voz en redes digitales
Fuente: Carballar José Antonio. "VoIP, La telefonía de Internet". [20]

Elementos necesarios

Para llevar a cabo la comunicación y transmisión de la voz humana es necesario el empleo de los siguientes elementos:

- **Computadora.-** La computadora es utilizada para realizar el proceso de envío de información, es necesario contar con la disposición de un micrófono y parlantes, también es necesario contar con un software que permita realizar la digitalización y codificación.
- **Teléfono IP.-** El teléfono IP tiene una apariencia muy similar a la de los teléfonos tradicionales que disponemos en casa, pero por ser un teléfono IP consta de elementos internos que permiten convertir la voz e información IP y viceversa.
- **Teléfono tradicional con adaptador IP.-** El teléfono tradicional requiere de un adaptador IP, para realizar la conversión a IP y viceversa.

Figura 2.4: Terminales telefónicos de VoIP.
Fuente: Carballar José Antonio. "VoIP, La telefonía de Internet". [20]

Ventajas de la telefonía IP

Las ventajas de la telefonía IP son las siguientes:

- Bajo coste, es decir es mucho más económico comparado con los sistemas tradicionales.
- Calidad de la voz en las comunicaciones IP por la incorporación de banda ancha.
- Es muy sencillo la introducción de la telefonía IP en internet.
- La tecnología IP está incluida en todos los dispositivos personales como computadoras, laptops, celulares, tabletas etc. [20]

2.2.7 Raspberry Pi

La raspberry pi es un ordenador de placa reducida, muy potente y de bajo costo, capaz de realizar funciones como cualquier computadora, con la ventaja que posee una estructura física reducida.

Montaje

Para montar la raspberry pi es necesario:

- Un Cargador 1,5 A.
- Una pantalla con HDMI o una pantalla con RCA.
- Teclado y ratón, Cable Ethernet
- Tarjeta SD 32 Gb. (Anexo A-literal c). [21]

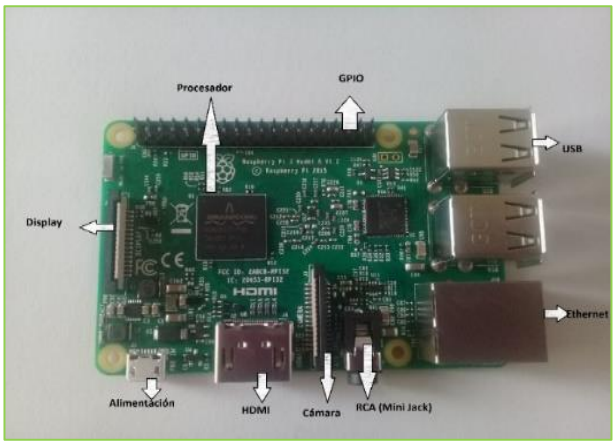


Figura 2.5: Montaje de Raspberry Pi.
Fuente: Fotografía tomada por el investigador.

Sistema Operativo

Uno de los sistemas operativos más utilizados en la raspberry pi es Raspbian, que se trata de un Debian (Linux) optimizado para este ordenador, el sistema operativo será instalado en la tarjeta SD para una mejor optimización de la tarjeta.

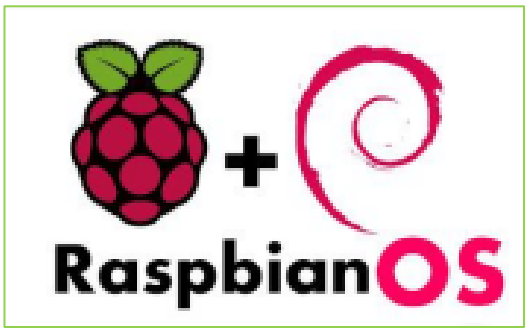


Figura 2.6: Sistema Operativo en la Raspberry Pi.
Fuente: García Víctor Suárez, "Introducción a la Raspberry Pi". [21]

Aplicaciones de la Raspberry Pi

Entre las aplicaciones que se pueden mencionar están las siguientes:

- **Computadora de oficina**.- La Computadora, puede ser utilizada como un ordenador de oficina, gracias al entorno de Raspbian.
- **Programación.-** Tiene un lenguaje de programación muy fácil de manejar como es Python, que permite activar los puertos GPIO de la Raspberry Pi.
- **IoT.-** El Internet de las cosas, por el desarrollo que tiene, se puede utilizar la Raspberry Pi para la comunicación con diferentes servidores como por ejemplo Windows para realizar aplicaciones IoT.
- **Juegos.-** Los juegos son otra aplicación que se puede dar a la Raspberry Pi, al convertir a la Raspberry en una retroConsola, con distribuciones como Lakka o RecalBox. [21]

Puertos GPIO

La GPIO (Entrada/Salida de propósito general), es una interfaz que permite conectar la Raspberry Pi con el mundo exterior, por medio de pines que permiten la comunicación con el pequeño computador.

Consta de 40 clavijas de contacto, dividida en dos filas de 20, se encuentra ligeramente separada una de la otra, son unos conectores que se encuentran soldados a la Raspberry. [22]

Figura 2.7: GPIO en la Raspberry Pi.
Fuente: Fotografía tomada por el investigador.

2.2.8 Arduino

Plataforma Arduino

Arduino es una placa electrónica de hardware libre, en la cual se emplea la programación para el desarrollo de aplicaciones mediante el uso de sensores se puede tomar valores de variables físicas para ser controladas mediante la programación en la placa electrónica de código abierto y puede afectar al entorno mediante el control de luces, motores, etc.

Massimo Banzi fue un profesor del Instituto IVRAE, de quién nació la idea de utilizar una placa electrónica de bajo costo para que sea accesible para sus estudiantes y ellos puedan aprender ya que en esos tiempos las placas electrónicas existentes tenían costos muy elevados. El nombre de arduino se debe a un lugar que era muy frecuentado por Massimo Banzi el bar Italiano llamado “di Re Arduino”.

Figura 2.8: Logo de arduino
Fuente: Enríquez Rafael, “Guía de Usuario de Arduino”. [23]

Ventajas

Las ventajas de Arduino son las siguientes:

- **Económico:** Es mucho más barato que otras placas electrónicas.
- **Multiplataforma:** Puede ser ejecutado desde diferentes Sistemas Operativos.
- **Entorno de programación simple:** Es muy fácil de usar y de programar.
- **Código abierto:** El software Arduino está publicado como herramienta de código abierto. [23]

Arduino Uno

Arduino Uno es una placa microcontroladora basada en la ATmega 328P, consta de catorce pibes digitales de entrada/salida, seis entradas analógicas, un cristal de cuarzo de 16 MHz, una conexión USB, un conector de alimentación, una cabecera ICSP y un botón de reinicio.

Fue la primera placa electrónica que salió al mercado, desarrollada como referencia para el desarrollo de posteriores placas. (Anexo A-literal a) [24].

Figura 2.9: Placa Arduino Uno.
Fuente: Fotografía tomada por el investigador.

Arduino Ethernet Shield

Arduino Shield Ethernet es una placa electrónica que permite conectar un Arduino a una red Ethernet, está basada en el chip Ethernet Wiznet w5100, es compatible con el Arduino Uno y Arduino Mega.

El Shield Ethetnet provee un conector Ethernet estándar RJ45, un conector lector de tarjeta Micro SD y el botón de reset. Para poder conectarse a internet es necesario el uso de la placa Ethernet Shield y de la librería Ethernet.h, al igual que arduino esta placa es de código abierto y de libre acceso.

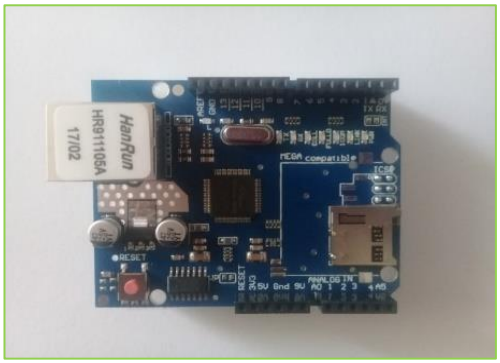


Figura 2.10: Arduino Ethernet Shield.
Fuente: Fotografía tomada por el investigador.

Características:

Las características del Arduino Shield Ethernet son las siguientes:

- Tensión de alimentación 5V.
- Controlador Ethernet: W5100 con una memoria interna de 16K.
- Velocidad de conexión 10/100 Mb
- Conexión con arduino a través del puerto SPI. (Anexo A-literal b). [24]

2.2.9 Sensores

Se define como sensor al dispositivo que es capaz de adquirir magnitudes físicas del medio y transformarlas en variables eléctricas por ejemplo la temperatura, la humedad, presión, potencial de hidrógeno, conductividad eléctrica, etc.

2.2.10 Sensor de temperatura del agua

El sensor de temperatura de agua, es un dispositivo que se encarga de captar la temperatura a la que se encuentra el agua, consta de un cable de poliuretano impermeable, el sensor mide con exactitud la temperatura en pozos, presas y estanques, donde se requiere el uso de cables largos, la longitud del cable no afecta la exactitud de las mediciones. [25]

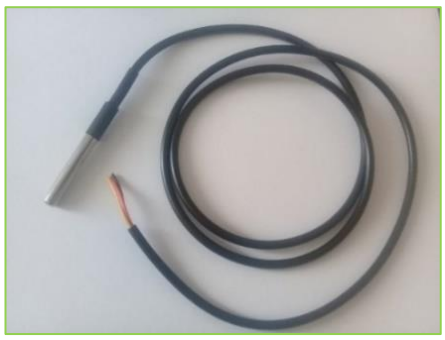


Figura 2.11: Sensor de temperatura del agua.
Fuente: Fotografía tomada por el investigador.

2.2.11 Sensor de luz

El sensor de luz, es un dispositivo que se encarga de medir la cantidad de luz que llega a una célula foto-eléctrica, causando así una reacción, como activar un circuito, una lámpara, etc. [26]

Figura 2.12: Sensor de luz.
Fuente: Fotografía tomada por el investigador.

2.2.12 Sensor de humedad y temperatura ambiental

El sensor de humedad y temperatura ambiental, es un dispositivo que se encarga de medir tanto la humedad como la temperatura del ambiente, utiliza un sensor capacitivo de humedad y un termistor para medir el aire circundante. [27]

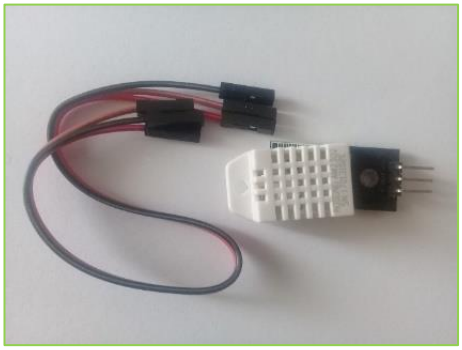


Figura 2.13: Sensor de humedad y temperatura ambiental.
Fuente: Fotografía tomada por el investigador.

2.2.13 Sensor de pH

El sensor de pH, es un dispositivo utilizado para medir la acidez o la alcalinidad de una solución, el pH indica la concentración de iones de hidrógeno presentes en dicha solución. [28]


Figura 2.14: Ejemplo de un sensor de pH.
Fuente: Fotografía tomada por el investigador.

2.2.14 Módulo Relé

Es un mecanismo electromagnético dirigido por un circuito eléctrico, el cual lo hace funcionar como un interruptor que sirve para encender uno o varios contactos, los mismos que posibilitan abrir o cerrar circuitos eléctricos independientes, todo esto se lo realiza mediante una bobina y un electroimán.

La bobina conecta los contactos mediante la creación de un campo magnético, mientras que el electroimán permite que los contactos se cierren. [29]

Figura 2.15: Módulo Relé
Fuente: Fotografía tomada por el investigador.

2.2.15 Base de datos

Base de datos es una entidad en la cual se pueden almacenar datos de manera estructurada, de forma que un programa de ordenador pueda seleccionar rápidamente los fragmentos de datos que sea de su requerimiento.

Una base de datos es un sistema de archivos electrónicos, que se organizan por campos, registros y archivos. [30]

Componentes

En un sistema de base de datos se identifican los siguientes componentes:

- Datos: Los Datos son la información almacenada en forma de archivos.
- Hardware: El Hardware consta de unidades de almacenamiento de información.
- Software: El Software son los programas de gestión de base de datos.
- Usuarios: Los Usuarios son los programadores y administradores de bases de datos. [31]

Sistema de gestión de base de datos

El Sistema de Gestión de Base de Datos es un programa que pone a disposición las técnicas de base de datos.

- Descripción centralizada de los datos, para evitar la replicación y que todas las aplicaciones se encuentren en un mismo lugar.

Los objetivos de un sistema de gestión de datos son los siguientes:

- Independencia de datos, los programas de aplicación deben verse afectados lo menos posible por cambios efectuados en datos que no se usan.
- Integridad de datos, la información que está en la base de datos debe cumplir con ciertos requisitos de calidad, para ello debe almacenarse correctamente.
- Seguridad de los datos, sólo personas autorizadas pueden tener acceso a la información almacenada en la base de datos. [31]

MySQL

MySQL, es un sistema gestor de base de datos simple y de un muy buen rendimiento, por lo cual es muy conocido y considerablemente utilizado por los usuarios. Esta opción se usa regularmente en aplicaciones comerciales debido a su fácil manejo.

MySQL es utilizado en varias plataformas y se distribuye libremente en internet bajo licencia GPL, cuenta con una estabilidad de un alto grado. [29]

PhpMy admin

Es una herramienta escrita en PHP con la que se opera la administración de MySQL mediante páginas web, usando internet. En la actualidad phpMyAdmin puede crear y eliminar bases de datos, envía datos en distintos formatos, crea, elimina y altera cualquier tabla y es capaz de ejecutar cualquier sentencia SQL. [32]

2.2.16 Servidores

Un servidor es una maquina informática que presta servicio a otras máquinas, ordenadores o usuarios, a los cuales les entrega todo tipo de información.

Tipos de servidores

A continuación se detalla los diferentes tipos de servidores.

- Servidor de Correo. - El servidor de correo trabaja con el e-mail de sus clientes y se encarga de almacenar, enviar, recibir y realizar todas las operaciones de dicho e-mail.

- Servidor Proxy.- El servidor proxy interviene como mediador, cuando un servidor no conoce la identidad del cliente que le ha hecho una petición.
- Servidor Web: El servidor web acumula documentos HTML, videos, imágenes, textos y cualquier tipo de información. Y además transmite y envía esta información a sus clientes.
- Servidor de Base de Datos: El servidor de base de datos proporciona a sus clientes servicios de gestión de base de datos y de almacenamiento.
- Servidores Clúster: Los servidores clúster se especializan en almacenar grandes cantidades de información, evitando que dicha información se pierda.
- Servidores Dedicados: Los servidores dedicados son propios para una sola persona o empresa.
- Servidores de Imágenes: Los servidores de imágenes están encargados de almacenar gran cantidad de imágenes, sin terminar con los recursos del servidor web. [33]

Apache

Apache es un servidor web que trabaja con software Open Source, es uno de los servidores más utilizados a nivel mundial ya que proporciona seguridad y un muy buen rendimiento.

El servidor Apache es un servidor web multiplataforma, es decir que puede trabajar con diferentes sistemas operativos y mantener su excelente rendimiento.

Apache es utilizado principalmente, para brindar servicio a páginas web, ya sean estáticas o dinámicas, es el servidor web del popular sistema XAMP, junto con MySQL y los lenguajes de programación Python, Php y Perl. [34]

2.3 Propuesta de Solución

En la presente investigación se desarrolló un sistema de control electrónico de acuarios con la finalidad de un monitoreo continuo y periódico, de las condiciones de calidad de agua, luminosidad, temperatura ambiental y del agua, para un buen desarrollo y reproducción de peces en el acuario.

CAPÍTULO III

METODOLOGÍA

3.1 Modalidad de la Investigación

Investigación Aplicada

El presente proyecto se define como proyecto de Investigación Aplicada, porque se realizó la implementación física en el Acuario "San Martín", como producto de la investigación realizada en el proyecto.

Investigación bibliográfica-documental

La investigación bibliográfica-documental se obtuvo de libros, revistas, artículos y base de datos científica de la Universidad Técnica de Ambato que proporcionaron el conocimiento de las indagaciones existentes como teorías, resultados, experimentaciones, instrumentos y técnicas usadas para llevar a cabo el proyecto.

Investigación de campo

Se realizó la investigación de campo para el análisis y estudio del sistema de control electrónico, para la obtención de información de la situación actual de los acuarios y el análisis de su entorno.

3.2 Población y Muestra

Por la característica de la investigación no se requiere población y muestra.

3.3 Recolección de Información

Para el desarrollo del proyecto, se utilizaron documentos, como libros, papers investigativos, proyectos de investigación previos tomados de los repositorios pertenecientes a la Universidad Técnica de Ambato, así como de repositorios de las diferentes Universidades del país, además de dispositivos electrónicos para la adquisición de datos.

3.4 Procesamiento y Análisis de Datos

Recopilación y análisis de la información obtenida de la investigación bibliográfica-documental.

De la información obtenida de los dispositivos electrónicos se realizó un análisis y una comparación de los datos para determinar si son los adecuados.

Realización de cuadros comparativos de las tecnologías, para establecer las ventajas y mejores precios al momento de seleccionar las tecnologías a utilizar.

Realización de cuadros comparativos de los diferentes sensores, para elegir los más óptimos para el análisis y la recolección de datos.

Determinación de las mejores alternativas en sensores y tecnologías.

Linealizar los datos de las variables para determinar si se encuentran en los rangos establecidos.

La información obtenida de cada sensor se almacenó en la base de datos para su posterior uso y lectura mediante una llamada a la central telefónica implementada en Asterisk.

La información obtenida en la investigación de campo sirvió para aclarar dudas y brindar un mejor sistema de control electrónico en el acuario.

3.5 Desarrollo del Proyecto

Para el desarrollo del proyecto, se llevó a cabo las siguientes actividades:

- Análisis de la situación actual del acuario "San Martín" de la ciudad de Baños.
- Estudio de las variables físicas que intervienen en el comportamiento de los peces como es la calidad de agua, la temperatura y su alimentación.
- Análisis de los sensores que intervienen en la toma de datos para el procesamiento y análisis de información
- Análisis de las tecnologías para el desarrollo del sistema de control electrónico.
- Determinación del tipo de tecnología para el monitoreo remoto de una pecera en el Acuario Serpentario "San Martín".

- Diseño del prototipo para el monitoreo del ambiente confinado en una pecera del Acuario “San Martín”.
- Simulación del prototipo para el monitoreo del ambiente confinado en una pecera del Acuario “San Martín”.
- Implementación física del prototipo en el Acuario Serpentario “San Martín”.
- Realización de pruebas en la central telefónica para establecer comunicación con la red GSM.
- Pruebas de funcionamiento y corrección de errores.
- Elaboración del Informe Final.

CAPÍTULO IV

DESARROLLO DE LA PROPUESTA

El presente proyecto de investigación está enfocado a atender los diferentes problemas que se presentan en el Acuario Serpentario "San Martín", en el control periódico de las variables físicas como es la temperatura de agua, la cantidad de luz, calidad del agua, que intervienen directamente con el desarrollo y reproducción de los peces.

En el desarrollo del presente proyecto se realizó el sistema de control electrónico para el pez "Astronotus Ocellatus", conocido comúnmente como "Óscar", este tipo de pez, comúnmente habita en el río Amazonas, bajo temperaturas cálidas.

Al no contar con un sistema de control electrónico en el Acuario Serpentario "San Martín", se debe revisar diariamente, la variación de la temperatura, la cantidad de luz presente en el acuario, la calidad del agua, y la temperatura ambiental.

Se debe controlar la temperatura del agua, porque el nivel de temperatura varía según las especies de peces, esta variación debe ser controlada periódicamente, en el Acuario "San Martín" se controla que la temperatura del pez "Oscar" se encuentre en un rango de temperaturas de 28°C a 32°C. El control de la calidad de luz, es muy importante en un acuario, porque se pretende brindar un ciclo biológico natural día/noche. La calidad del agua, brinda un ambiente agradable a los turistas, que disfrutan de la transparencia de las peceras para visualizar claramente a las diferentes especies de peces, además de brindar un ambiente libre de impurezas para que los peces se desarrollen con normalidad. Se realiza un control de la temperatura ambiental en el acuario, cuando esta es superior a 25°C, para que el clima sea agradable y los turistas disfruten del lugar.

Para la elaboración del presente Proyecto de Investigación, se detalla a continuación el desarrollo de la propuesta.

4.1 Análisis de la factibilidad

El estudio de la factibilidad permitió conocer las necesidades de este proyecto de investigación y su viabilidad en cuanto a los factores que intervienen para su desarrollo tales como:

a) Factibilidad Institucional

El presente proyecto de investigación posee factibilidad institucional, porque fue implementado en el Acuario Serpentario "San Martín".

b) Factibilidad Técnica

El presente proyecto de investigación posee factibilidad técnica, porque se cuenta con todos los materiales que son de fácil adquisición, tanto en el mercado nacional como en el internacional.

c) Factibilidad Bibliográfica

El presente proyecto de investigación posee factibilidad bibliográfica porque cuenta con la información necesaria, obtenida de repositorios de las diferentes Universidades, libros, papers, artículos científicos, así también del Acuario Serpentario "San Martín" de Baños.

d) Factibilidad Económica

El presente proyecto de investigación posee factibilidad económica porque el valor total del sistema es accesible para el investigador.

4.2 Requerimientos

Para el desarrollo del Sistema de Control Electrónico, fue importante conocer las características básicas del pez Óscar, para brindarle cuidados y protección según sus requerimientos, es decir, conocer los rangos de las variables físicas que intervienen en el acuario como es, la temperatura del agua, cantidad de luz y calidad del agua.

Para la preservación del acuario hay que tomar en cuenta los parámetros principales que intervienen en el ambiente confinado, del pez Óscar, porque de ello depende la vitalidad y reproducción de los peces.

Es necesario controlar ciertos parámetros físicos como son los siguientes:

- Temperatura del agua

- Potencial de Hidrógeno (pH)
- Iluminación
- Temperatura Ambiental

a) Temperatura en los acuarios

Es muy importante tomar en cuenta la temperatura del agua en un acuario, porque esta interviene directamente en el ritmo biológico de los peces, un cambio de temperatura puede afectar la salud de las especies e incluso producir su muerte, cuando se tiene un alto nivel de temperatura, hace que el metabolismo del pez consuma mayor energía.

En el Acuario Serpentario "San Martín", se requiere un control de la Temperatura del agua, del pez Óscar, verificando que la temperatura no sea inferior a 28°C, y de ser el caso se enciende un calentador de agua, cuando alcanza la temperatura de 32°C que se encuentra graduada en el calentador, este se desactiva.

b) Calidad de agua en los acuarios

Para un control de la calidad de agua se utilizan sensores de pH que permiten visualizar la alcalinidad del agua, el nivel de pH está cuantificado en relación al número de iones libres de hidrógeno presente en las moléculas de agua, se tiene una escala logarítmica que va desde 0 a 14, en el cuál se considera al nivel 7 como neutro, los niveles del 0 al 7 se denominan como ácidos y los niveles de 7 a 14 se denominan como alcalinos.

En el Acuario Serpentario "San Martín" de Baños, se requiere realizar un control periódico de la calidad del agua, en el hábitat del pez Óscar, controlando que los niveles de pH se encuentren en un rango de 6,5 a 7,5 que indican que el agua está libre de impurezas, caso contrario se enciende un luz piloto que indica que el agua debe ser cambiada.

c) Luminosidad en los acuarios

La iluminación permite tener un ambiente estéticamente agradable y simular un ambiente natural, es decir tener en cuenta el ciclo natural día-noche para que los peces y plantas cuenten con un normal desarrollo, es recomendable utilizar lámparas fluorescentes, que no son perjudiciales, al contrario de la luz que emite radiación ultravioleta.

En el Acuario Serpentario “San Martín” de Baños, se requiere una lámpara fluorescente de 20 W con una eficiencia de 68 lm/watt, dicha lámpara se enciende entre las 7h00 y 18h00, para brindar el ciclo biológico natural día-noche al pez Óscar.

d) Humedad y temperatura ambiental

La humedad ambiental indica la cantidad de vapor de agua presente en el aire y esta puede ser medida en base a la humedad absoluta y la humedad relativa. La humedad del aire se debe al vapor de agua que se encuentra presente en la atmósfera, la cantidad de vapor de agua que puede absorber el aire depende de su temperatura.

La temperatura ambiental es una magnitud física que indica la intensidad de frío o de calor presente en el ambiente, es decir es el estado del ambiente que se manifiesta en el aire en forma de calor, se presenta convencionalmente como, frío y caliente.

En el Acuario Serpentario “San Martín” de Baños, se requiere un ventilador eléctrico de 110 V, cuando la temperatura ambiente sea superior a 25°C. De esta manera se controla la temperatura ambiental, porque al encontrarse el Acuario en un lugar cálido y cerrado, provoca que los niveles de temperatura aumenten, produciendo fatiga y cansancio a los turistas.

4.3 Diseño del Prototipo para el Sistema de Control Electrónico del Acuario.

En el presente capítulo se realiza el diseño e implementación de la propuesta del trabajo de investigación utilizando la plataforma de Arduino, Raspberry Pi 3, sensores para el monitoreo de las variables físicas, una pequeña central telefónica para la comunicación y conocimiento del estado del acuario.

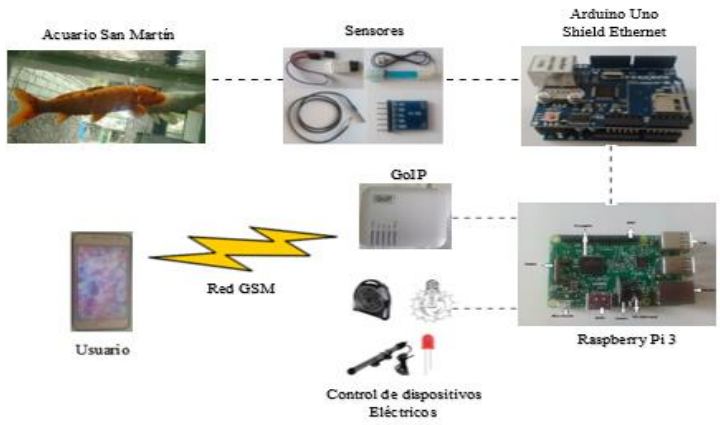


Figura 4.1: Diseño del Prototipo del Sistema de Control Electrónico.
Fuente: Investigador

4.4 Diagrama de bloques del Sistema de Control Electrónico del Acuario

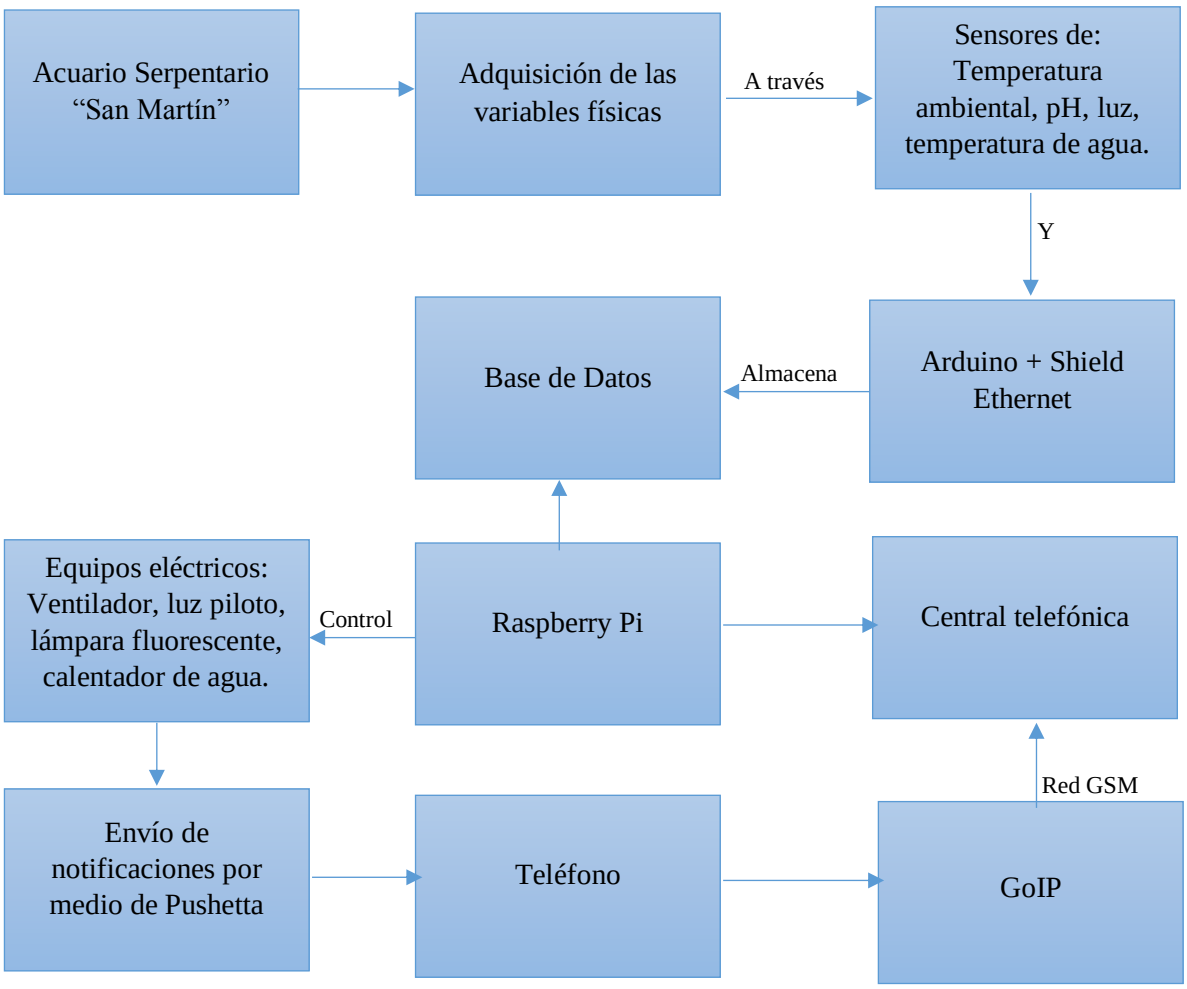


Figura 4.2: Diagrama de Bloques del Sistema de Control Electrónico.
Fuente: Investigador

Con la conexión de arduino, de la Shield Ethernet y de los sensores, de temperatura del agua, iluminación, pH y temperatura ambiental, se adquieren las variables físicas proporcionadas por el Acuario Serpentario "San Martín". La información obtenida de cada sensor, es almacenada en una base de datos creada en la Raspberry Pi. Para el control de las variables físicas antes mencionadas, se realiza la activación de equipos eléctricos como es un ventilador, una luz piloto, una lámpara fluorescente y un calentador de agua, que permite obtener un sistema electrónico controlado, además del envío de notificaciones a través de Pushetta cuando uno de los equipos eléctricos han sido activados. Con la creación de una central telefónica en la Raspberry PI y el GoIP, se logra acceder la información proporcionada por el sistema, por medio de una sola llamada telefónica desde cualquier lugar del mundo.

4.5 Análisis de Sensores

Para la selección de los sensores que intervienen en la toma de datos y procesamiento de información, se realizó un análisis para determinar el tipo de sensores a utilizar en el presente proyecto.

4.5.1 Análisis del sensor de humedad y temperatura ambiental

Se realizó un análisis entre los sensores de humedad y temperatura ambiental DHT22 y SHT11, a continuación, se detallan las características técnicas de cada uno.

Características técnicas de los sensores:

Tabla 4.1: Sensor SHT11 vs Sensor DHT22.

Especificaciones	**SHT11**	**DHT22**
Alimentación	2.4 – 5.5VDC	3.3V – 5VDC
Lectura de humedad	± 3%	± 2% - 5%
Lectura de temperatura	± 0.4°C	± 0.5°C
Capacidad de medir humedad	20% - 80%	0% - 100%
Capacidad de medir temperatura	-40°C – 123.8°C	-40°C – 125°C
Dimensiones	15.5mmx12mmx5.5mm	15.1mmx25mmx5.5mm
Costo	$15,50	$8,77

Fuente: Investigador basado en [35] [36].

El sensor DHT22 es ligeramente más preciso, tiene mayores rangos de medición, y es mucho más barato comparado con el sensor SHT11. Es por ello se eligió al sensor de humedad y temperatura ambiental DHT22.

4.5.2 Análisis del sensor de luz

Se realizó un análisis entre el sensor de luz BH1750 y el sensor HD2021T, a continuación se detallan las especificaciones técnicas de cada uno, para una mejor elección.

Tabla 4.2: Sensor HD2021T vs Sensor BH1750.

Especificaciones	**HD2021T**	**BH1750**
Alimentación	16-40Vdc o 24 Vac	3V- 5V
Potencia Absorbida	10mA	7mA

Temperatura de trabajo	-20 a 60°C	-40 a 100°C
Dimensiones	58mmx65mmx52mm	43mmx57mmx48mm
Campo de medida	0,002 – 2klux	0,5 – 4lux
Longitud del cable	150m	50cm
Costo	$18,00	$14,73

Fuente: Investigador basado en. [37] [38].

Se eligió el sensor digital de luz BH1750 porque tiene una alta precisión, fácil uso con la plataforma Arduino, por las librerías que vienen incluidas para el sensor BH1750, tiene un mayor campo de medida lux. El sensor puede ser configurado en su resolución, lo que afecta directamente la velocidad de medición. A continuación se detalla los 3 modos de resolución del sensor BH1750.

Tabla 4.3: Resolución del sensor BH1750.

Modo	**Resolución**	**Tiempo de medición**
High Resolution Mode2	0.5 lx	120 ms
High Resolution Mode	1 lx	120 ms
Llow Resolution Mode	4 lx	16 ms

Fuente: Mechatronics N. “Sensor de luz BH1750” [37].

Los tres modos de resolución del sensor BH1750 son de 0.5lx, 1lx y 4lx, al variar la resolución también varía el tiempo de medición, con una mayor resolución se tiene un menor tiempo de medición. Las tres resoluciones se subdividen en Continous y One Time. El modo Continous, se realizan mediciones constantemente y caso contrario se emplea el método One Time, éste módulo se apaga después de realizar la medida correspondiente.

4.5.3 Análisis del sensor de pH

Se realizó un análisis entre los sensores de pH SEN0161 y el sensor Sonda de electrodo de pH a continuación se detallan las especificaciones técnicas de cada uno para una mejor elección.

Tabla 4.4: Sensor SEN0161 vs Sensor GHL Profilux Ph.

Especificaciones	**SEN0161**	**GHL Profilux Ph Electrode BNC**
Alimentación	5V	4.5 V

Rango de medición	0 – 14 pH	0 – 14 pH
Medición de Temperatura	0 – 60°C	0 – 80°C
Precisión	± 0.1 pH (25°C)	± 0.05pH
Tiempo de Respuesta	≤ 1min	≤ 1min
Sensor de pH	Con conector BNC	Con conector BNC
Ajuste de ganancia	Del potenciómetro	-
Tamaño del Módulo	43mm x 32mm	150mm
Costo	$60,53	$79,60

Fuente: Investigador basado en [39] [40].

Para el desarrollo del presente proyecto se utilizó el sensor de pH SEN0161, a pesar de que las características técnicas entre los dos sensores son muy similares, el sensor análogo SEN0161 fue desarrollado especialmente para ser utilizado con arduino y su precio es ligeramente más económico.

4.5.4 Análisis del sensor de temperatura

Se realizó un análisis entre los sensores de temperatura del agua, DS18B20 y SDI-12. En la siguiente tabla se detalla las especificaciones técnicas entre los sensores de temperatura sumergibles, DS18B20 y SDI-12.

Características técnicas de los sensores:

Tabla 4.5: Sensor DS18B20 vs Sensor SDI-12.

Especificaciones	**DS18B20**	**SDI-12**
Alimentación	3 – 5.5V	7 -16V
Longitud del cable	108cm	18.25m
Rango de temperatura	-55°C – 125°C	-40 – 60-°C
Precisión	± 0.5°C	± 0.1°C
Resolución	9 – 12 bits	16 bits
Diámetro	4mm	1.6 cm
Intervalo de medición	750 ms	1s
Costo	$10,00	$8,50

Fuente: Investigador basado en [41] [42].

Para la elección del sensor de temperatura, se buscó un sensor que sea sumergible en el agua y para ello se encontró la mejor opción el sensor DS18B20 que es un sensor digital, este sensor es de uso exclusivo para el control de temperatura en peceras.

Otra razón por la que se eligió este sensor es por el fácil uso y manejo desde arduino, ya que cuenta con las librerías necesarias para su funcionamiento, como son las librerías, Dallas Temperature y OneWire.

4.6 Análisis de tecnologías

Raspberry Pi es una computadora completamente funcional que posee una estructura física reducida y Arduino es un microcontrolador, el cual sólo es un componente de una computadora.

A continuación, se detallan algunas características técnicas de Arduino y Raspberry Pi

Tabla 4.6: Arduino y Raspberry.

Especificaciones	**Arduino Uno**	**Raspberry PI B**
Procesador	ATMega 328	Arm11
Velocidad	16 MHz	700 MHz
RAM	2 KB	1GB
Voltaje de entrada	5V	5V
USB	n/a	4
Audio	n/a	HDMI, Analógico
Video	n/a	HDMI, Analógico
Ethernet	n/a	10/100
I/O	14 GPIO, 6 – 10 bit análogo	40 GPIO
Tamaño	2.95” x 2.1”	3.37” x 2.125”
Sistema Operativo	n/a	Linux
Entorno	Arduino ID	Linux, IDLE, Open-Embedded, QEMU, Scratchbox, Eclipse.
Costo	$ 11	$65

Fuente: Investigador basado en [43] [44] [45]

La Raspberry Pi es mucho más rápida que un Arduino, cuando se habla de velocidad de reloj, porque es aproximadamente 42 veces más rápido, también es mucho más rápido en cuanto a memoria RAM , por ende tiene un precio elevado, comparado con un Arduino. La Raspberry Pi cuenta con más del doble de entradas y salidas de propósito general que un Arduino, lo que permite la interacción con un mayor número de elementos eléctricos y electrónicos.

En el desarrollo del presente proyecto se utilizó tanto el Arduino como la Raspberry Pi, porque se tuvo una deficiencia de voltaje en la activación de sensores mediante el uso de la Raspberry, es por ello que fue necesaria la utilización de un Arduino, para la alimentación y adquisición de información proporcionada por los sensores.

4.6.1 Arduino

La siguiente tabla muestra una rápida comparación entre las principales características de algunas placas Arduino.

Tabla 4.7: Características de Arduinos

Nombre	**Procesador**	**Voltaje**	**Velocidad de la CPU**	**Análogo In/Out**	**Digital IO/PWM**	**Costo**
Uno	ATmega328p	5V	16 MHz	6/0	14/6	$12,00
101	Intel	3.3V	32 MHz	6/0	14/4	$30,00
Mega2560	ATmega2560	5V	16 MHz	6/0	54/15	$20,50
Gemma	ATtiny85	3.3V	8 MHz	1/0	3/2	$12,95
Micro	ATmega32U4	5V	16 MHz	12/0	20/7	$9,50
Due	ATSAM3X8E	3.3V	84 MHz	12/2	54/12	$32,00
Nano	ATmega168	5V	16 MHz	8/0	14/6	$40,00

Fuente: Arduino P. “Compare Boards” [46]

El arduino que se eligió para el desarrollo del presente proyecto, fue el Arduino Uno, porque cumple con los requerimientos para el desarrollo del proyecto al integrarse con la placa de Arduino Shield Ethernet, para el envío de la información proporcionada por los sensores hacia la base de datos, también porque su precio es económico.

4.6.2 Raspberry Pi

Para el desarrollo del presente proyecto de investigación, se analizó las diferentes Raspberry Pi, que existen en el mercado.

A continuación se detallan las características de algunas Raspberry Pi.

Tabla 4.8: Características de Raspberry

Características	Raspberry PI B	Raspberry PI B+	Raspberry PI 2	Raspberry PI 3
CPU	ARMv6 700 MHz	ARMv6 700 MHz	ARMv7 900 MHz	ARMv8 1,2 GHz
RAM	512 MB	512 MB	1024 MB	1024 MB
Ethernet WIFI	Ethernet 10/100	Ethernet 10/100	Ethernet 10/100	Ethernet 10/100 – WIFI 802.11 b/g/n
Alimentación	5V 1A	5V 1A	5V 2A	5V 2.5A
Velocidad de subida	2.2 MB/s	2.2 MB/s	2.2 MB/s	2.2 MB/s
Velocidad de Bajada	2.84 MB/s	2.84 MB/s	2.84 MB/s	2.84 MB/s
Temperatura Mínima	33.6°C	31.5°C	34.2°C	33.1°C
Temperatura Máxima	50.8°C	50.8°C	50.8°C	48.7°C
Costo	$40,00	45,00	$48,85	$70,00

Fuente: ElectronicLab "Raspberry Pi". [47]

Hay una ventajosa diferencia entre los dos primeros modelos de Raspberry Pi y los dos últimos, principalmente en la capacidad de la memoria RAM que se ha duplicado, pero siendo está aún muy limitada, por contar con 1 GB de memoria RAM, otra de las ventajas con respecto a los dos primeros modelos, son los puertos USB que se han expandido, anteriormente sólo se contaba con un puerto, actualmente la Raspberry Pi3, cuenta con cuatro puertos USB, por ende tiene un precio elevado con respecto a las versiones anteriores. Se eligió la Raspberry Pi 3, por ser la última versión disponible, porque se requiere más de dos puertos USB y porque cuenta con las características necesarias para el desarrollo del prototipo.

4.7. Diseño de Circuitos

De acuerdo a la selección de sensores y tecnologías, se realizó el diseño de los circuitos, para el desarrollo del prototipo del Sistema de Control Electrónico para el Acuario Serpentario "San Martín" del Cantón de Baños, de la provincia de Tungurahua.

a) Circuito del sensor de humedad y temperatura ambiental DHT22

El sensor de humedad y temperatura ambiental consta de 3 pines, que son los siguientes: Vcc, Gnd y Output. El sensor se alimenta desde Arduino a través de los pines Vcc y Gnd, a continuación se conecta la salida Output al pin digital 2 del Arduino. (Anexo B-literal c).

En la figura 4.3 se muestra el esquema eléctrico del sensor DHT22.

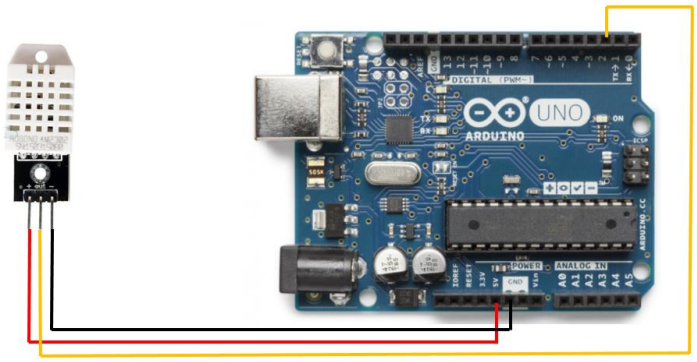


Figura 4.3: Esquema eléctrico del sensor DHT22.
Fuente: Investigador

b) Diseño del circuito del sensor de luz BH1750

El sensor de luz BH1750 consta de 5 pines, que son los siguientes: Vcc, Gnd, SCL, SDA y ADDR. El sensor se alimenta desde Arduino a través de los pines Vcc y Gnd, a continuación se conecta la señal SDA y SCL a los pines análogos 4 y 5 respectivamente. (Anexo B-literal b).

En la figura 4.4 se muestra el esquema eléctrico del sensor BH1750.

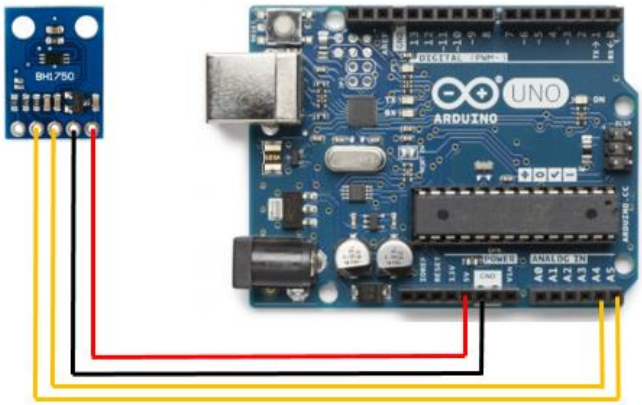


Figura 4.4: Esquema eléctrico del sensor BH1750.
Fuente: Investigador

c) Diseño del circuito del sensor de pH SEN0161

El sensor de pH SEN0161 consta de 3 pines, que son los siguientes: Vcc, Gnd y G_0. El sensor se alimenta desde Arduino a través de los pines Vcc y Gnd, a continuación se conecta la señal G_0 al pin análogo 3 del Arduino. (Anexo B-literal d).

En la figura 4.5 se muestra el esquema eléctrico del sensor SEN0161.

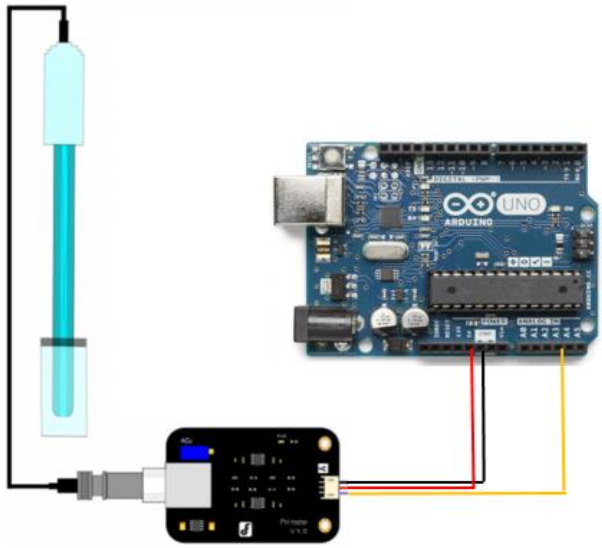


Figura 4.5: Esquema eléctrico del sensor SEN0161.
Fuente: Investigador

d) Diseño del circuito del sensor de temperatura del agua DS18B20

El sensor de temperatura del agua consta de 3 pines, que son los siguientes: Vcc, Gnd y Output. El sensor se alimenta desde Arduino a través de los pines Vcc y Gnd, a continuación se conecta la salida Output al pin digital 8 del Arduino. (Anexo B – literal a). En la figura 4.6 se muestra el esquema eléctrico del sensor DS18B20.

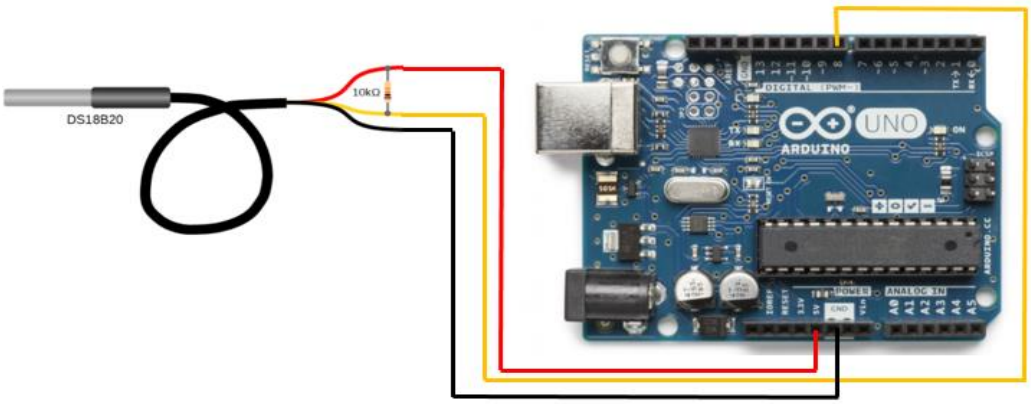


Figura 4.6: Esquema eléctrico del sensor DS18B20.
Fuente: Investigador

e) Diseño del Sistema de Control Electrónico instalado en el Acuario.

Se conectan los Sensores, el Módulo relé, la Raspberry Pi, el Arduino, el GoIP, el Router y los Equipos Eléctricos que funcionan a 110V, como se muestra en la figura 4.7.

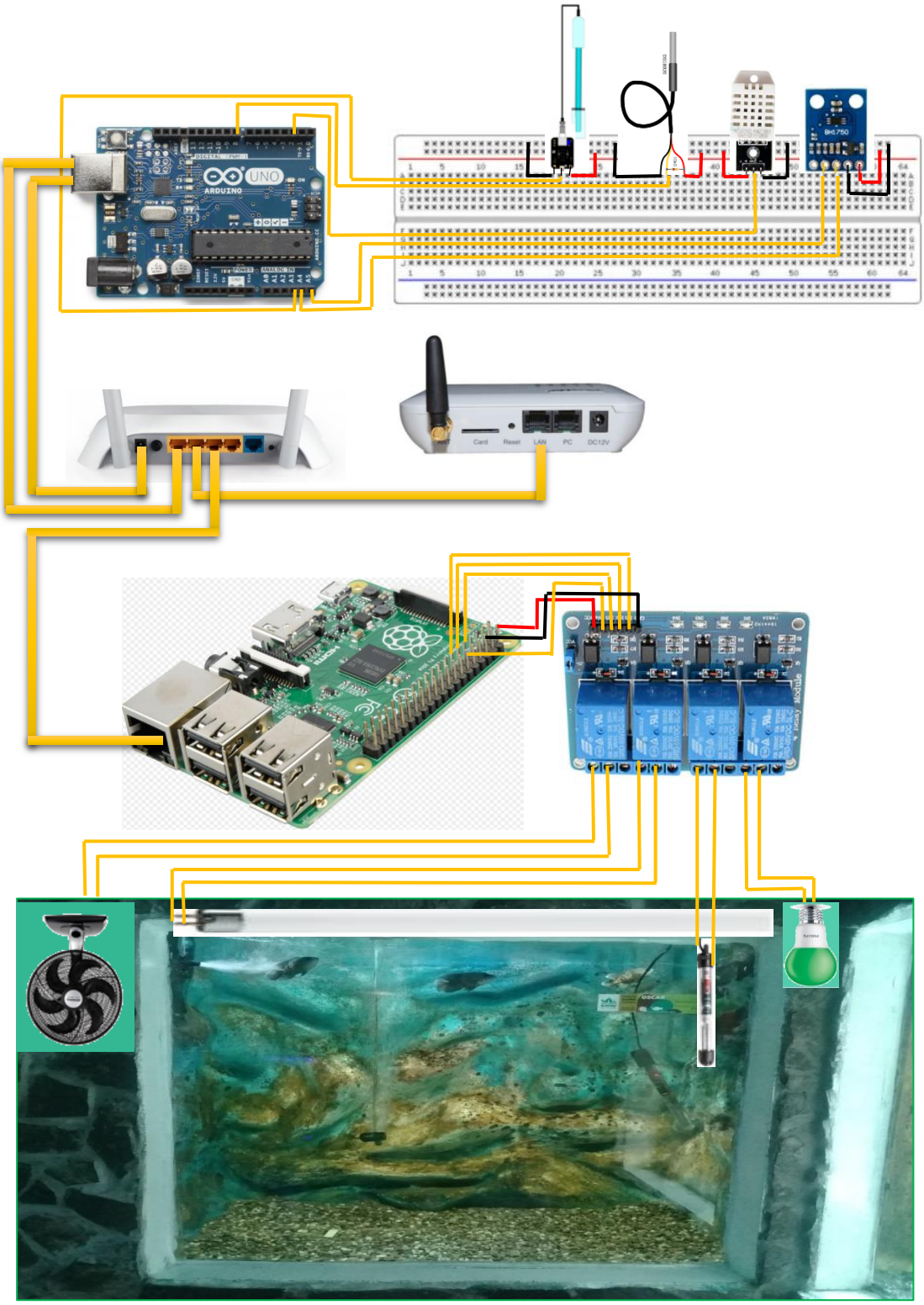


Figura 4.7: Diseño del Sistema de Control Electrónico instalado en el Acuario.
Fuente: Investigador

En la figura 4.7 se puede observar la conexión de sensores y tecnologías utilizas en el presente proyecto de investigación. A continuación se describe el proceso de conexión, que se llevó a cabo para implementar el prototipo del Sistema de Control Electrónico en el Acuario Serpentario "San Martín" del Cantón Baños de la provincia de Tungurahua.

Se alimentan todos los sensores a Vcc y Gnd, a continuación se envía la señal de cada sensor al Arduino. Las señales del sensor de luz BH1750 se conecta a las entradas análogas A4 y A5, la señal del sensor DHT22 se conecta a la entrada digital 2, la señal del sensor de agua DS18B20 se conecta a la entrada digital 8 y finalmente la señal del sensor de pH SEN0161 se conecta a la entrada análoga A3 del Arduino.

Se alimenta al arduino con el cable de impresora, que va conectado a la entrada USB del Router, y para el envío de información obtenida de los sensores, por medio de la red, se utiliza un cable UTP que se conecta desde la Shield Ethernet al Router.

A continuación se alimenta la Raspberry Pi con un cargador de 5V a 1.5A, se conecta un cable UTP desde el Router a la Raspberry para tener conexión a internet. De los puertos GPIO de la Raspberry Pi, se conectan al módulo relé para la activación de equipos eléctricos, que funcionan a 110v.

Del puerto GPIO 18 se conecta al módulo relé que permite la activación del ventilador, cuando la temperatura ambiental es superior a 25°C, del puerto GPIO 17 se conecta al módulo relé que permite la activación de la luz piloto, cuando el pH no se encuentra entre 6.5 y 7.5, del puerto GPIO 22 se conecta al módulo relé que permite la activación de la lámpara fluorescente, cuando la iluminación es inferior a 25 lux entre las 7h00 y 18h00, y finalmente del puerto GPIO 27 se conecta al módulo relé que permite la activación del calentador de agua, cuando la temperatura del agua es inferior a 28°C y cuando es superior a 32°C desactiva el puerto.

Finalmente se alimenta el GoIP con una fuente de 12V, y se conecta un cable UTP desde el Router al GoIP, para que se encuentre dentro de la misma red.

4.8 Implementación del Sistema Operativo en la Raspberry Pi

Para poder utilizar la Raspberry Pi 3, en primer lugar se instala el Sistema Operativo Raspbian, la imagen del Sistema Operativo se puede adquirir desde la página oficial de Raspberry Pi.

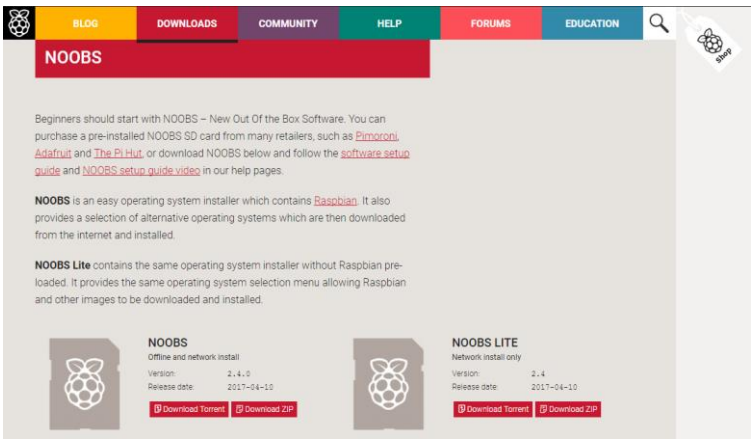


Figura 4.8: Descargar Imagen de Sistema Operativo
Fuente: Investigador

Para la instalación del Sistema Operativo, se necesita una tarjeta micro SD superior a 4GB para no tener inconvenientes de almacenamiento, en el desarrollo de este proyecto de investigación se ha utilizado una tarjeta SD de 32GB, es primordial que la tarjeta SD se encuentre vacía, por lo que se procede a formatearla, para formatear la SD y las particiones que contenga se realizó con el programa SDFormatter que se lo puede obtener en el siguiente enlace:

https://www.sdcard.org/downloads/formatter_4/eula_windows/index.html.

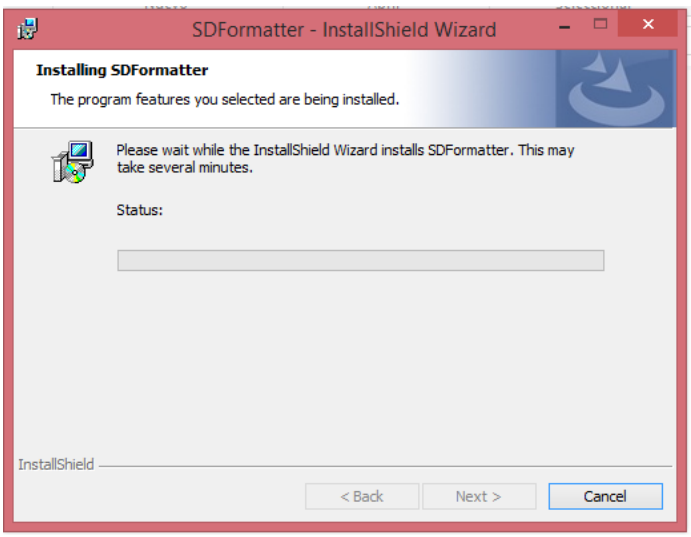


Figura 4.9: SDFormatter
Fuente: Investigador

Una vez instalado el programa se procede a formatear la Micro SD.

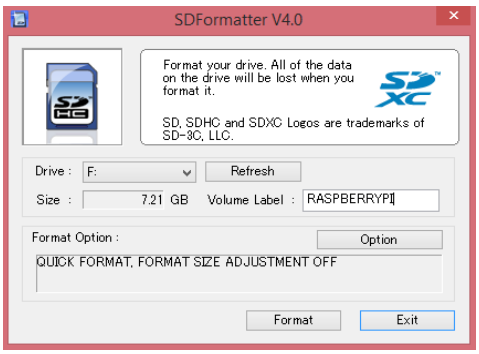


Figura 4.10: Formatear SD
Fuente: Investigador

4.9 Instalación del Sistema Operativo Raspbian

Para la instalación del Sistema Operativo en la Raspberry Pi, se descomprime el archivo descargado de la página oficial de Raspberry y se copia a la tarjeta SD, una vez concluido este proceso se inserta la SD en la raspberry Pi, y se procede a encenderla, a continuación se visualiza el sistema Operativo a Instalar y se selecciona Raspbian, el idioma y teclado en español como se puede observar en la figura 4.11.

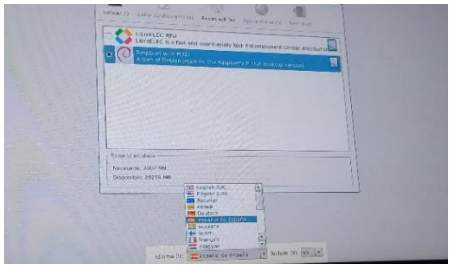


Figura 4.11: Instalación de Raspbian
Fuente: Investigador

El proceso de instalación se lleva a cabo en el transcurso de 10 a 20 minutos, todo el proceso es automático una vez insertada la tarjeta micro SD, una vez concluido la instalación del Sistema Operativo Raspbian, la Raspberry Pi está lista para utilizarse como se visualiza en la figura 4.12.

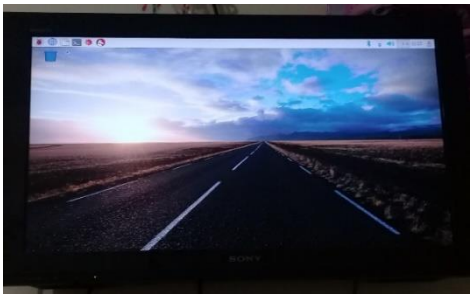


Figura 4.12: Inicio del Sistema Operativo
Fuente: Investigador

Una vez listo el Sistema Operativo Raspbian, se ingresa al terminal y se ejecutan los siguientes comandos que permiten actualizar los paquetes y versiones disponibles, este proceso se lo debe realizar antes de la instalación de cualquier programa en la Raspberry Pi.

- sudo apt-get update.
- sudo apt-get update.

4.10 Instalación del servidor Apache

Para la instalación del servidor Apache, se ejecuta el siguiente comando:

- sudo apt-get install apache2.

Para verificar que la instalación fue correcta, en el navegador se ingresa a: http://localhost y deberá visualizarse la siguiente página.

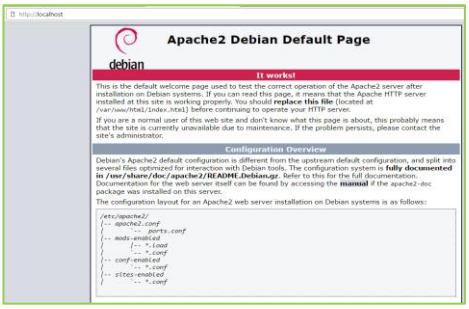


Figura 4.13: Página por defecto de Apache.
Fuente: Investigador

4.11 Instalación de PHP

Para la instalación de PHP se ejecuta el siguiente comando desde la terminal:

- sudo apt-get install php5.

Es importante instalar los paquetes que necesita PHP para su correcto funcionamiento, por lo cual se instalan con los siguientes comandos:

- sudo apt-get install libapache2-mod-php5.
- sudo apt-get install libapache2-mod-perl2 php5 php5-cli php5-common php5-curl php5-dev php5-gd php5-imap php5-ldap php5-mhash php5-mysql php5-odbc.

A continuación se ingresa al archivo html y se crear un archivo info.php.

- cd /var/www/html.

- sudo nano info.php.
- <?php phpinfo(); ?>.

Para comprobar la correcta instalación de PHP, se ingresa la siguiente dirección en el navegador: http://localhost/info.php, y se visualiza la página por defecto de PHP.

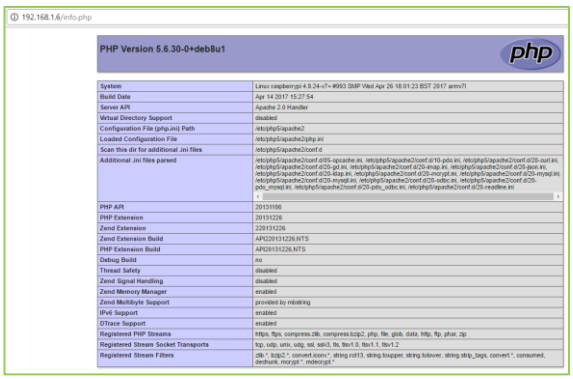


Figura 4.14: Página por defecto de PHP.
Fuente: Investigador

4.12 Instalación base de datos MySQL

Para la instalación de la base de datos MySQL se ejecuta el siguiente comando.

- sudo aptget install mysql-server mysql-client php5-mysql.

En el proceso de la instalación, se debe configurar la contraseña para el ingreso a MySQL, se inserta una contraseña y se acepta los términos.

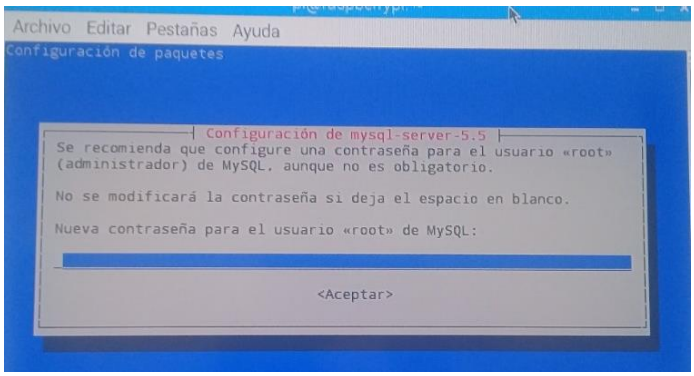


Figura 4.15: Configuración de MySQL
Fuente: Investigador

Es importante reiniciar el servicio de MySQL y se lo realiza con el siguiente comando:

- sudo service mysql start.

Se ingresa a MySQL, y a continuación se introduce la contraseña configurada durante la instalación.

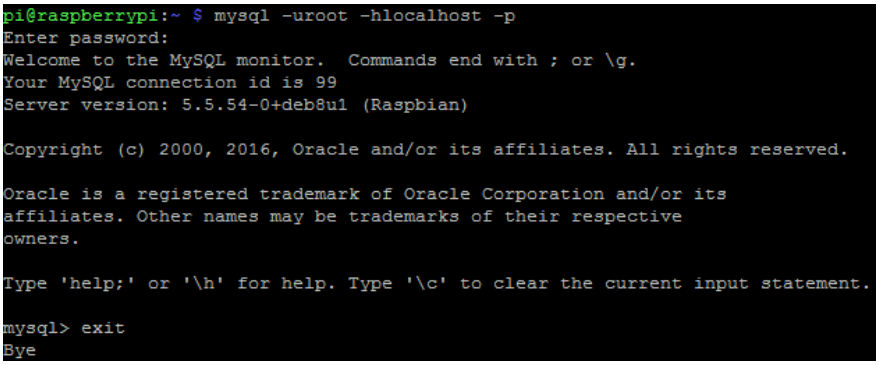


Figura 4.16: Ingreso a MySQL.
Fuente: Investigador

4.13 Instalación de PhpMyAdmin

Para la instalación de PhpMyAdmin se ejecuta el siguiente comando.

- sudo apt-get install phpmyadmin.

En el proceso de la instalación se debe realizar las siguientes configuraciones:

Elegir el servidor que debe ser configurado automáticamente al correr PhpMyAdmin, en la figura 4.17 se puede visualizar el servidor que se eligió para la instalación.

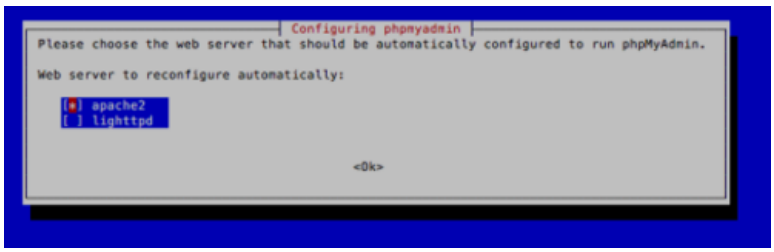


Figura 4.17: Elección del servidor Apache2.
Fuente: Investigador

A continuación se configura el acceso a la base de datos por medio de phpMyAdmin, se ingresa la contraseña de MySQL y a continuación se ingresa la clave que permitirá el acceso a phpMyAdmin.

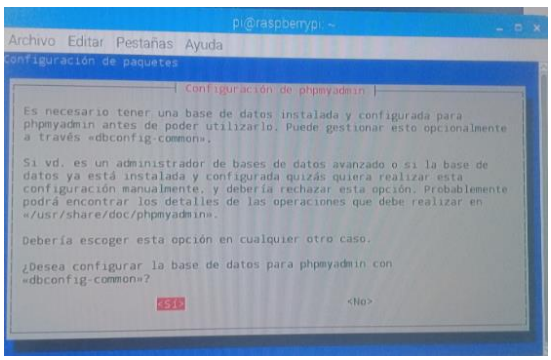


Figura 4.18: Configuración de PhpMyAdmin.
Fuente: Investigador

A continuación se realiza las siguientes configuraciones para el correcto funcionamiento de la base de datos:

En la terminal se ingresa al archivo de apache2.conf, y se añade al final del archivo la sentencia: Include /etc/phpmyadmin/apache.conf.

- sudo nano /etc/apache2/apache2.conf.

Se reinicia el servicio mediante el siguiente comando**:**

- sudo /etc/init.d/apache2 restart.

Ingresar al fichero php.ini:

- sudo nano /etc/php5/apache2/php.ini.

Una vez en el fichero, se agrega la siguiente línea de código: "extension=mywql.so", como se visualiza en la imagen 4.19.

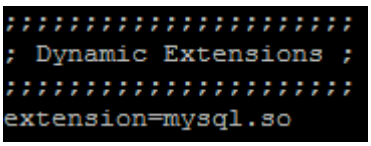


Figura 4.19: Archivo de configuración php.ini
Fuente: Investigador

Para actualizar los comandos insertados anteriormente, se reinicia la Raspberry Pi 3 mediante el siguiente comando:

- sudo reboot.

Se crea una carpeta con el nombre conf.d.

- sudo mkdir /etc/apache2/conf.d.

A continuación se ejecuta en la terminal el siguiente comando:

- sudo ln -s /etc/phpmyadmin/apache.conf /etc/apache2/conf.d/phpmyadmin.conf.

A continuación se actualiza el servidor Apache2 con el siguiente comando:

- sudo /etc/init.d/apache2 reload.

Finalmente para comprobar el correcto funcionamiento de PhpMyAdmin se ingresa al navegador y se inserta la siguiente dirección: http://localhost/phpmyadmin y si toda la instalación fue correcta se visualiza una imagen como se muestra en la figura 4.20.

Figura 4.20: Página por defecto de PhpMyAdmin.
Fuente: Investigador

4.14 Creación de la base de datos en PhpMyAdmin

Para ingresar a la base de datos, el usuario es root y la contraseña que haya sido configurada durante la instalación, a continuación se crea un nuevo usuario con el nombre: "arduin" y la contraseña: "godinmylife", como se visualiza en la imagen 4.21.

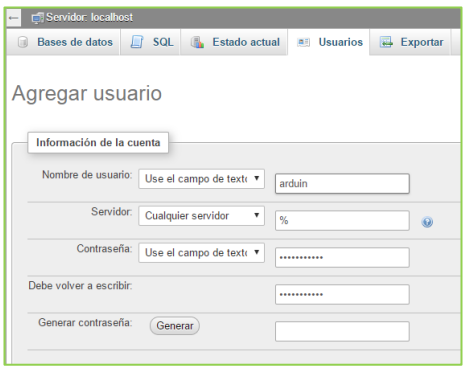


Figura 4.21: Creación de Usuario en PhpMyAdmin.
Fuente: Investigador

Se crea una base de datos, llamada "sensores", que almacena la información de cada uno de los sensores utilizados en el presento proyecto.

Figura 4.22: Base de datos sensor.
Fuente: Investigador

a) Sensor Temperatura de Agua

A continuación se selecciona la base de datos creada “sensores” y se crea una tabla de 2 columnas con el nombre “temperatura_agua”, como se indica en la figura 4.23.

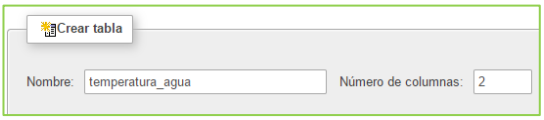


Figura 4.23: Tabla para sensor de temperatura_agua.
Fuente: Investigador

A continuación se configuran las 2 columnas en donde se almacena la información obtenida del sensor de temperatura de agua.

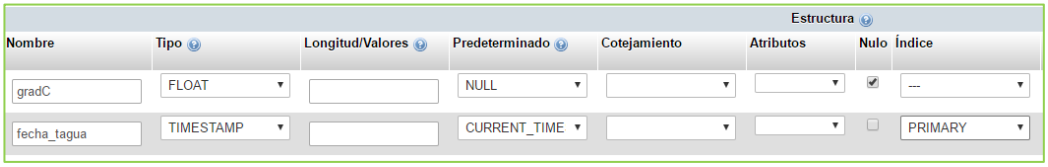


Figura 4.24: Configuración de las columnas de la tabla temperatura_agua
Fuente: Investigador

b) Sensor Temperatura Ambiente

Se selecciona la base de datos creada “sensores” y se crea una tabla de 2 columnas con el nombre “temperatura_ambiente”, como se indica en la figura 4.25.

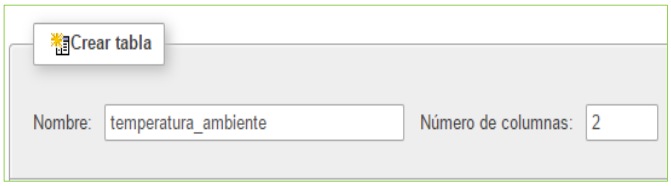


Figura 4.25: Tabla para sensor de temperatura ambiente.
Fuente: Investigador

A continuación se configuran las 2 columnas en donde se almacena la información obtenida del sensor de temperatura ambiental.

Figura 4.26. Configuración de las columnas de la tabla temperatura_ambiente.
Fuente: Investigador

c) Sensor Humedad Ambiente

Se selecciona la base de datos creada “sensores” y se crea una tabla de 2 columnas con el nombre “humedad_ambiente”, como se indica en la figura 4.27.

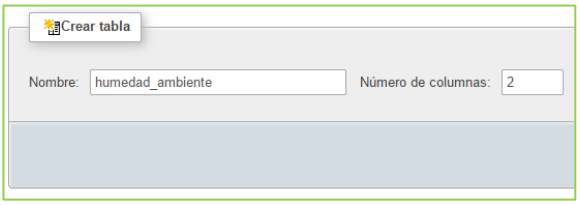


Figura 4.27: Tabla para sensor de humedad ambiente.
Fuente: Investigador

A continuación se configuran las 2 columnas en donde se almacena la información obtenida del sensor de humedad ambiental.

Figura 4.28: Configuración de las columnas de la tabla humedad_ambiente
Fuente: Investigador

d) Sensor de Iluminación

Se selecciona la base de datos creada “sensores” y se crea una tabla de 2 columnas con el nombre “iluminacion”, como se indica en la figura 4.29

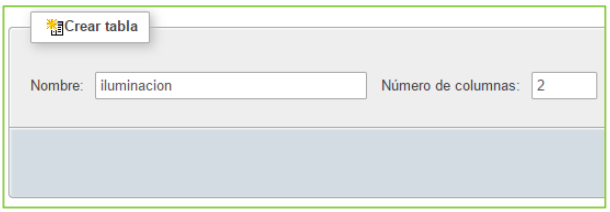


Figura 4.29: Tabla para sensor de iluminación.
Fuente: Investigador

A continuación se configuran las 2 columnas en donde se almacena la información obtenida del sensor iluminación.

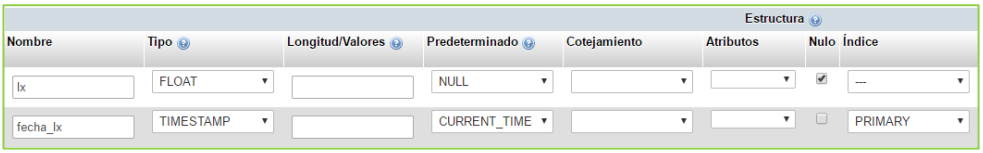


Figura 4.30: Configuración de las columnas de la tabla iluminación.
Fuente: Investigador

e) Sensor de pH

Se selecciona la base de datos creada “sensores” y se crea una tabla de 2 columnas con el nombre “nivel_pH”, como se indica en la figura 4.31.

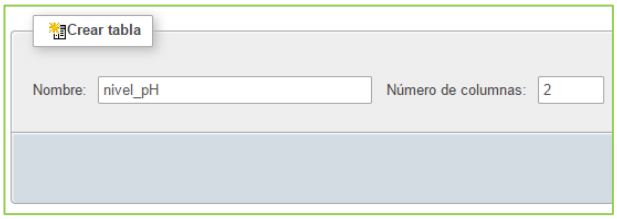


Figura 4.31: Tabla para sensor de pH.
Fuente: Investigador

A continuación se configuran las 2 columnas en donde se almacena la información obtenida del sensor de pH.

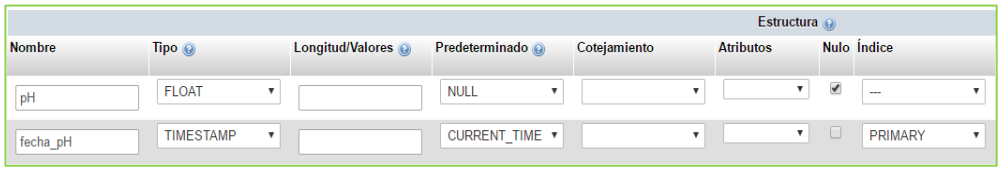


Figura 4.321: Configuración de las columnas de la tabla pH.
Fuente: Investigador

Finalmente se visualiza la base de datos con las tablas creadas para la obtención de información de los sensores utilizados en el presente proyecto.

Figura 4.33: Base de datos "sensores".
Fuente: Investigador

4.15 Conexión Arduino y Raspberry Pi3

Para la comunicación entre Arduino y la base de datos instalada en la Raspberry Pi3 se necesita de un lenguaje que pueda comunicar los dos entornos, y para ello se utiliza el lenguaje PHP. Para lo cual se crea una carpeta llamada “acuario”.

- cd/var/www/html
- mkdir acuario

En la carpeta creada se guardará todos los archivos en lenguaje PHP que permite la conexión y envío de información entre Arduino y la Base de Datos.

4.16 Instalación del servidor FTP

Para la transferencia de archivos desde Windows a Raspbian se instala el servidor FTP.

Primero se instala el programa Dreamweaver CS6, en la cual se programa el código PHP para la comunicación.

Para la instalación del servidor FTP desde la terminal se ingresa el siguiente comando.

- sudo apt-get install vsftpd.

A continuación se ingresa en el archivo vsftpd.conf.

- sudo nano /etc/vsftpd.conf.

En el archivo vsftps.conf, se descomenta las siguientes líneas para permitir la escritura de archivos en la Raspberry Pi.

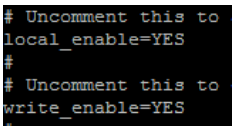


Figura 4.34: Configuración de archivo vsftpd.conf
Fuente: Investigador

Finalmente se actualiza el servidor ftp para empezar a utilizarlo.

- sudo service vsftpd restart.

Para tener comunicación entre Windows y Raspberry se instala Filezilla, que es un gestor de FTP muy utilizado por ser una multiplataforma de código abierto y software libre. Una vez instalado Filezilla, mediante el comando "ip add" se visualiza la IP a la que está conectada la Raspberry Pi3, como se indica en la figura 4.35.

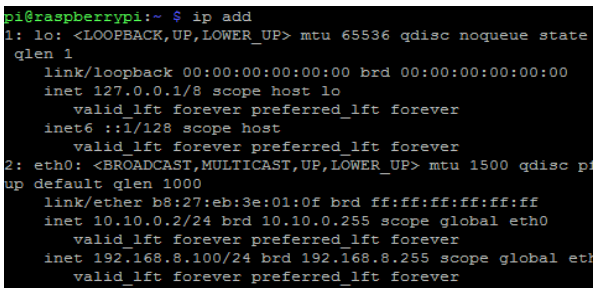


Figura 4.35: Visualización de la IP
Fuente: Investigador

A continuación se ingresa a Filezilla y remotamente al servidor, con la dirección IP de la Raspberry, es este caso 192.168.1.6 el usuario por defecto es pi y la contraseña si no ha sido cambiada con anterioridad es raspberry, el puerto de conexión 21 y se conecta como se visualiza en la figura 4.36.

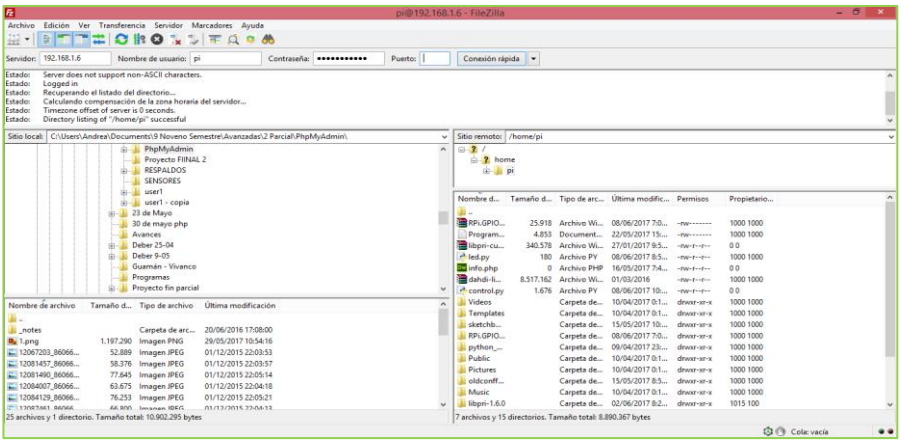


Figura 4.36: Filezilla
Fuente: Investigador

4.17 Instalación de Arduino

Para la instalación de arduino se ejecuta desde la terminal el siguiente comando:

- sudo apt-get install arduino arduino-core.

Se asignan permisos de escritura, lectura y ejecución mediante el siguiente comando:

- sudo chmod 666 /dev/ttyACM0.

Como se puede visualizar en la figura 4.37, el puerto serial no se encuentra activado.

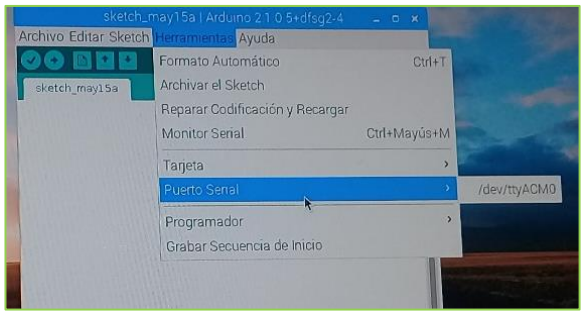


Figura 4.37: Puerto Serial Arduino.
Fuente: Investigador

Para activar el puerto serial, se ingresa los siguientes comandos.

- sudo apt-get install python-serial.
- sudo usermod –s –G dialout pi.

4.18 Asignación de un Dirección Estática

Para evitar cambios de direccionamiento lógico dentro de la red, se ha asignado una dirección estática. Para realizar la configuración se lo ha realizado mediante el siguiente comando.

- sudo nano /etc/network/interfaces.

Se edita el archivo de la siguiente manera, asignando la IP estática.

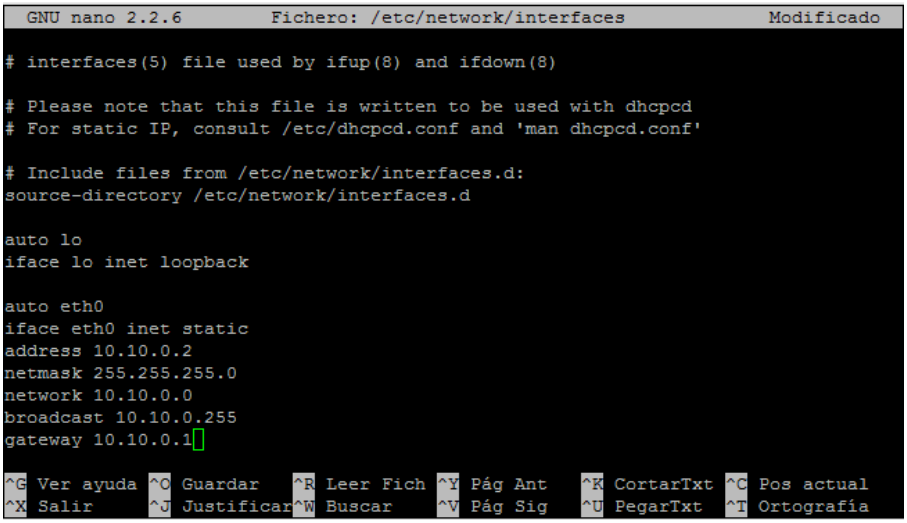


Figura 4.38: Asignación de IP estática.
Fuente: Investigador

Se guardan los cambios realizados y se reinicia el servicio mediante el comando:

- sudo /etc/init.d/networking restart.

4.19 Programación de sensores en Arduino

Para la programación de los sensores en arduino se cargan las librerías necesarias para la obtención de datos de los sensores y librerías para la Ethernet Shield, a continuación se declara los pines donde se conecta el dato del sensor, se define e inicializa variables, a continuación se detalla la programación utilizada para la lectura de la información proporcionada por el sensor de temperatura de agua. En la sección de Anexos (Anexo C-literal a), se encuentra la programación de todos los sensores ocupados en el presente proyecto.

```
//Sensor de Temperatura de Agua
float Tagua= s18b20.getTempCByIndex(0);
s18b20.requestTemperatures();
Serial.print("Temperatura_Agua: ");
Serial.print(Tagua);
Serial.println(" *C");
```

```
Serial.println("");
```

4.20 Proceso de envio de información desde Arduino al servidor

Para el proceso de envío de información obtenida de los sensores a la base de datos, se realiza el código de programación, que permite esta comunicación mediante el puerto 80, por medio de la instrucción GET se envía el dato por medio del script.php, y se realiza la conexión mediante el script "conexion.php".

// Sensor de humedad Ambiental

```
if (client.connect(IP_SERVIDOR, 80)>0)
{
client.print("GET /acuario/humedad_ambiental.php?RH=");
client.print(h);
client.println();
Serial.println("Dato enviado Humedad_Ambiental");
}
else
{
Serial.println("Sin conexion");
}
client.stop();
client.flush();
delay(2000);
```

Se realiza el mismo proceso para el envío de información con el resto de sensores utilizados en el presente proyecto, la programación del envío de datos sensados se lo puede visualizar en el (Anexo C-literal a).

4.21 Scripts para conexión con phpMyAdmin

Se crea el script "conexion.php" que contiene la información para la comunicación con la base de datos, declarando el nombre del usuario, la clave, el nombre de la base de datos, como se detalla a continuación:

```
<?php
// Autenticación
$equipobd = "localhost";
$usuariobd = "arduin";
$clavebd = "godinmylife";
$nombrebd = "sensores";
// Conexión con mysql
$conex =
mysqli_connect($equipobd,$usuariobd,$clavebd,$nombrebd);
?>
```

4.22 Scripts para el envío de información a la base de datos

Se crean los scripts que permiten enviar la información recibida desde el sensor de temperatura de agua, sensor de temperatura y humedad ambiental, sensor de iluminación y sensor de nivel de pH hacia la base de datos.

Primero se hace el llamado al script conexion.php, a continuación se lee el dato de la tabla creada en la base de datos (dependiendo del sensor requerido) mediante la instrucción GET, luego se guarda el valor para insertar en la base de datos y finalmente se inserta el valor en la base de datos. A continuación se detalla la programación para el sensor de temperatura del agua, la programación para el resto de sensores se detalla en la sección (Anexo C-literal b).

Sensor de Temperatura de Agua

Se crea el script temperatura_aguaa.php

Se crea el script que permiten enviar la información recibida desde el sensor de temperatura de agua a la base de datos.

```
<?php
// Se hace el llamado al script conexion.php
require("conexion.php");

// Se lee el dato de temperatura_agua  mediante GET
$datoagua = mysqli_real_escape_string($conex, $_GET['gradC']);

// Se guarda el valor para insertar en la base de datos
$insertartagua = "INSERT INTO temperatura_agua (gradC)
VALUES('".$datoagua."')";

// Se inserta el valor en la base de datos
mysqli_query($conex, $insertartagua);
mysqli_close($conex);
?>
```

```
pi@raspberrypi:/var/www/html/acuario $ sudo nano nivelph.php
pi@raspberrypi:/var/www/html/acuario $ sudo nano temperatura_aguaa.php
pi@raspberrypi:/var/www/html/acuario $ sudo nano temperatura_ambiental.php
pi@raspberrypi:/var/www/html/acuario $ sudo nano humedad_ambiental.php
pi@raspberrypi:/var/www/html/acuario $ sudo nano iluminacionn.php
pi@raspberrypi:/var/www/html/acuario $ sudo nano nivelph.php
```

Figura 4.39: Scripts creados en la carpeta acuario
Fuente: Investigador

4.23 Obtención de información sensada en la base de datos

Para él envió de información a la base de datos, se envía un valor numérico de la siguiente manera para comprobar que la programación se ha efectuado correctamente.

localhost/acuario/iluminacionn.php?lx=78

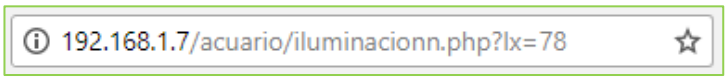


Figura 4.40: Envío de valor numérico a base de datos.
Fuente: Investigador

Y como se puede observar en la figura 4.41, el dato "78" ha sido enviado correctamente a la base de datos, de la misma manera se comprueba con el envío de información al resto de tablas de la base de datos.

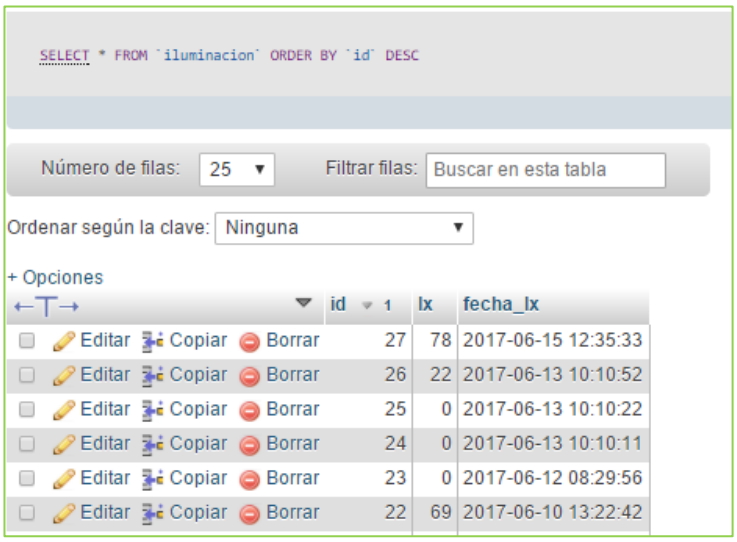


Figura 4.41: Lectura del valor enviado a la base de datos.
Fuente: Investigador

Una vez comprobado el correcto funcionamiento de lectura de información en la base de datos, se procede a conectar la placa Arduino + Ethernet Shield a la Raspberry PI3, y se visualiza la información proporcionada por los sensores en la base de datos.

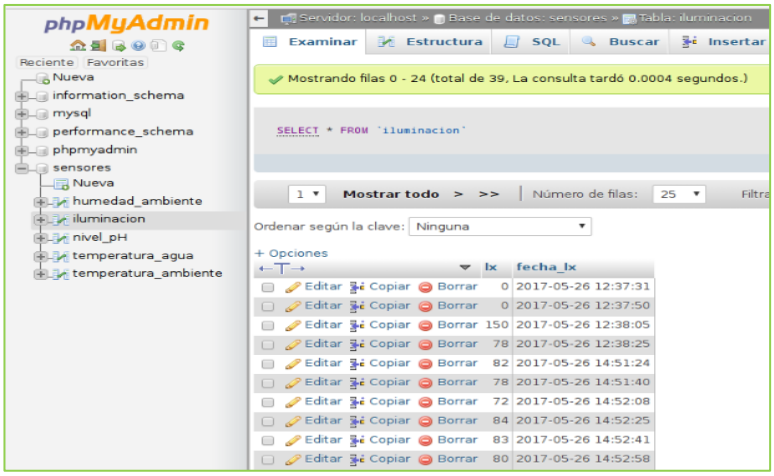


Figura 4.42: Valores sensados en la Base de datos.
Fuente: Investigador

4.24 Instalación de Asterisk

Para la instalación de Asterisk primero se instalan una serie de dependencias en el Sistema Operativo Debian.

- sudo apt-get install subversión.
- sudo aptitude install linux-headers-`uname -r` build-essential libxml2-dev libssl-dev libncurses-dev libnewt-dev doxygen subversion libmysqlclient-dev unixodbc-dev libmyodbc.

A continuación se dirige al directorio "cd /usr/src" para descargar Asterisk Una vez ingresado en el directorio mencionado, se descarga Asterisk mediante el siguiente comando:

- sudo svn checkout http://svn.asterisk.org/svn/asterisk/branches/11 asterisk-11.

A continuación se sitúa en el directorio "cd asterisk-11", y se instala los siguientes paquetes, para que cuando se quiera conectar Asterisk con líneas primarias no haya problemas en la conexión.

- sudo apt-get install libncurses5-dev libsqlite3-dev libssl-dev libiksemel-dev.

Se configura, compila e instala Asterisk.11.

- sudo ./configure --disable-xmldoc --disable-asterisskssl.

Figura 4.43: Configuración de Asterisk.11
Fuente: Investigador

Finalmente se ejecuta el siguiente comando para terminar con la instalación de Asterisk-11:

- sudo make menuselect & make & make install

Se reinicia el servicio de asterisk y se ingresa mediante el siguiente comando "sudo asterisk -r", y como se puede visualizar en la figura 4.44, Asterisk se encuentra instalado correctamente.

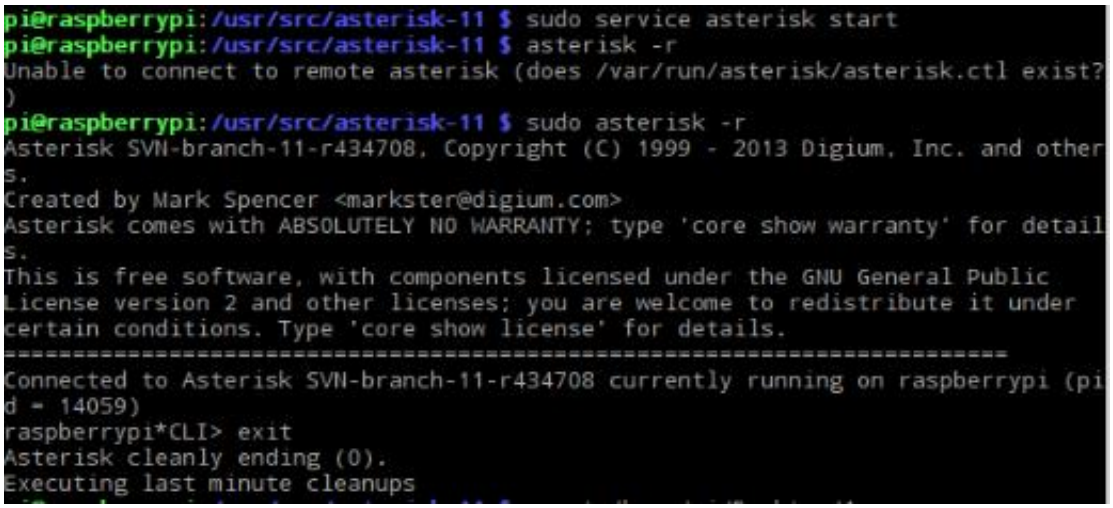


Figura 4.44: Ingreso a Asterisk
Fuente: Investigador

4.25 Instalación de Festival

Festival es un sistema de síntesis de voz, es decir un "text to speech", tiene como objetivo crear una relación entre las personas y los ordenadores a través de medios de audio. Es por ello que se ha visto necesaria la instalación de Festival, para integrar con Asterisk y poder realizar consultas del estado del sistema.

En primer lugar se instala Festival con el siguiente comando:

- sudo apt-get install festival.

Se ingresa a la carpeta donde se instaló Festival y se edita el archivo "festival.scm".

- cd /usr/share/festival/.
- sudo nano festival.scm.

En el archivo "festival.scm", se configura Festival con la voz predefinida, añadiendo al final del archivo las instrucciones, que se pueden observar en la figura 4.45.

```
;; Enable access to localhost (needed by debian users)
(set! server_access_list '("localhost\\.localdomain" "localhost"))

(define (tts_textasterisk string mode)
"(tts_textasterisk STRING MODE)
Apply tts to STRING. This function is specifically designed for
use in server mode so a single function call may synthesize the string.
This function name may be added to the server safe functions."
(let ((wholeutt (utt.synth (eval (list 'Utterance 'Text string)))))
(utt.wave.resample wholeutt 8000)
(utt.wave.rescale wholeutt 5)
(utt.send.wave.client wholeutt)))

(provide 'festival)
```

Figura 4.45: Configuración del archivo festival.scm.
Fuente: Investigador

Para integrar Festival con Asterisk se edita el archivo "festival.conf" que se encuentra en la dirección "cd /etc/asterisk".

- cd /etc/asterisk.
- sudo nano festival.conf.

En el archivo "festival.conf" se cerciora que las siguientes instrucciones se encuentren activadas y de no ser, se las activa, como se detalla en la figura 4.46.

[general]

host=localhost

port=1314

festivalcommand=(tts_textasterisk "%s" 'file)(quit)\n

```
  GNU nano 2.2.6          Fichero: festival.conf

;
; Festival Configuration
;
[general]
;
; Host which runs the festival server (default : localhost);
;
host=localhost
;
; Port on host where the festival server runs (default : 1314)
;
port=1314
;
; Use cache (yes, no - defaults to no)
;
usecache=yes
;
; If usecache=yes, a directory to store waveform cache files.
; The cache is never cleared (yet), so you must take care of cleaning it
; yourself (just delete any or all files from the cache).
; THIS DIRECTORY *MUST* EXIST and must be writable from the asterisk process.
; Defaults to /tmp/
;
cachedir=/var/lib/asterisk/festivalcache/
;
; Festival command to send to the server.
; Defaults to: (tts_textasterisk "%s" 'file)(quit)\n
; %s is replaced by the desired text to say. The command MUST end with a
; (quit) directive, or the cache handling mechanism will hang. Do not
; forget the \n at the end.
;
festivalcommand=(tts_textasterisk "%s" 'file)(quit)\n
```

Figura 4 .46: Configuración del archivo festival.conf
Fuente: Investigador

Finalmente se ejecuta festival, para guardar los cambios.

- usr/bin/festival –server > /dev/null 2>&1 &.

4.26 Instalación de voces en Español

En Asterisk las locuciones vienen instaladas de forma predeterminada en inglés, y para un mejor entendimiento en el presente proyecto se ha instalado las voces en español de la siguiente manera.

Se crea una carpeta "voces" en el directorio "cd /usr/src" para instalar las voces en español.

- sudo mkdir voces.

Se ingresa a la carpeta voces para instalar los paquetes en español que contienen los dígitos, la fonética, entre otros archivos para la activación de las voces en español.

- sudo wget https://www.sinologic.net/voces/voipnovatos-core-sounds-es-ulaw-1.4.tar.gz.
- sudo wget https://www.sinologic.net/voces/voipnovatos-extra-sounds-es-ulaw-1.4.tar.gz.

A continuación se descomprimen los archivos descargados para que Asterisk pueda hacer uso de los mismos.

- sudo tar zxvf voipnovatos-core-sounds-es-ulaw-1.4.tar.gz.
- sudo tar zxvf voipnovatos-extra-sounds-es-ulaw-1.4.tar.gz.

Finalmente se visualiza que los paquetes se encuentran en la carpeta voces, para la visualización se lo realiza con la instrucción "ls", como se puede visualizar en la figura 4.47.

```
pi@raspberrypi:/usr/src/voces $ ls
dictate  followme  silence
digits   letters   voipnovatos-core-sounds-es-gsm-1.4.tar.gz
es       phonetic  voipnovatos-extra-sounds-es-gsm-1.4.tar.gz
```

Figura 4.47: Visualización de los paquetes en la carpeta voces.
Fuente: Investigador

4.27 Festival con voz en español en Asterisk

Festival que se instaló en Asterisk, viene sin ninguna voz en español y por este motivo se escucha un poco diferente, porque es locución en inglés y además la calidad tampoco es buena, por eso es importante instalar las voces en español, existe un proyecto que es de la Junta de AndaLucía el cual consiste en dos paquetes para debían contiene la voz femenina y masculina para festival.

En este proyecto se utilizó la voz femenina de la Junta de AndaLucía, mediante Filezilla se transfiere el archivo "festvox-sflpc16k_1.0-1_all.deb" al directorio "cd /usr/src/".

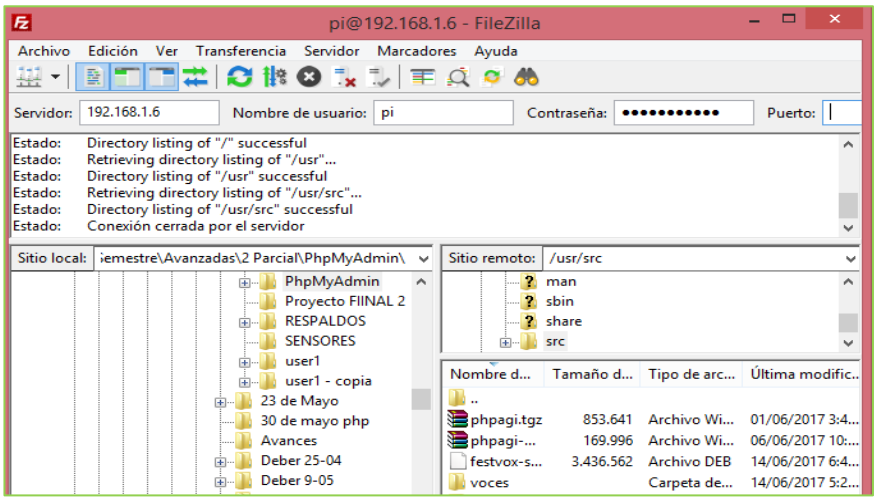


Figura 4.48: Transferencia de la voz en español mediante FileZilla.
Fuente: Investigador

A continuación se ingresa al directorio "cd /usr/src/" y se instala la voz femenina en español.

- sudo dpkg –i festvox-sflpc16k_1.1-1_all.deb.

A continuación se hace que festival utilice la voz femenina en español de forma predeterminada, para lo siguiente se dirige al directorio "cd /usr/share/festival/" y se modifica el archivo "siteinit.scm".

- sudo nano siteinit.scm.

Se agrega al final del archivo la siguiente línea de comando "(set! voice_default 'voice_JuntaDeAndalucia_es_sf_diphone)" que permite que Festival utilice la voz femenina por defecto.

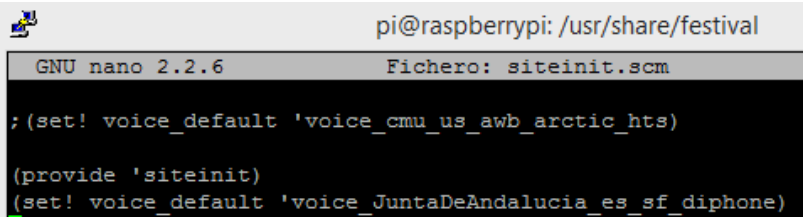


Figura 4.49: Insertar por defecto la voz en español.
Fuente: Investigador

Otro archivo que se modifica, se encuentra en el directorio "cd /etc/asterisk/", y se agrega las siguientes líneas de comando "text2wave= /usr/bin/text2wave", como se aprecia en la figura 4.50.

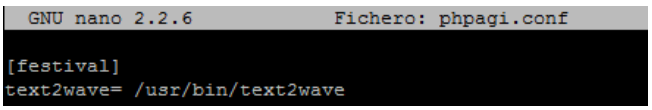


Figura 4.50: Lectura de texto utilizando festival.
Fuente: Investigador

Se modifica el archivo "voices.scm" que se encuentra en el directorio "cd /usr/share/festival/", para que Festival reconozca como voz principal, la voz femenina de la Junta de AndaLucía.

- sudo nano voices.scm.

Se cambia la línea "kal_diphone" por "JuntaDeAndalucia_es_sf_diphone".

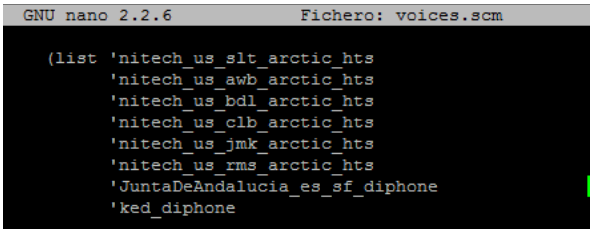


Figura 4.51: Agregar la voz femenina de la Junta de Andalucía.
Fuente: Investigador.

Y finalmente se inicia Asterisk, mediante el siguiente comando.

- sudo asterisk -rvvvvv

4.28 Asterisk Gateway Interface (AGI)

AGI permite desarrollar aplicaciones externas que pueden interactuar con Asterisk, en el presente proyecto estas aplicaciones están escritas en el lenguajes de programación "php", que permite controlar el dialplan mediante una llamada desde el archivo extensions.conf de Asterisk.

Para la implementación del IVR, con el call center es necesario utilizar la librería phpagi y el lenguaje de programación php.

Se descargar phpagi, y mediante FileZilla se transfiere el archivo al directorio cd /usr/src.

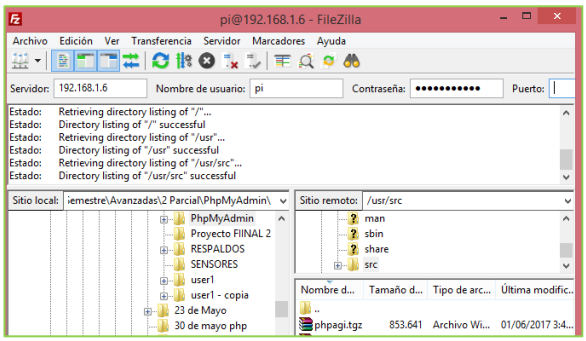


Figura 4.52: Transferencia de AGI mediante FileZilla.
Fuente: Investigador

A continuación se descomprime el archivo phpagi-2.20.

- sudo tar –xf phpagi-2.20.

Se ingresa al archivo y se copia el archivo phpagi.php y phpagi-asmanager.php en el directorio /var/lib/asterisk/agi-bin/.

- cp phpagi.php /var/lib/asterisk/agi-bin/.
- cp phpagi-asmanager.php /var/lib/asterisk/agi-bin/.

A continuación los archivos deben volverse ejecutables, para ello se lo realiza con los siguientes comandos:

- sudo chmod 777 /var/lib/asterisk/agi-bin/phpagi.php.
- sudo chmod 777 /var/lib/asterisk/agi-bin/ phpagi-asmanager.php.

4.29 Lectura de la información almacenada en la base de datos mediante Asterisk

A continuación se detalla la programación utilizada para la lectura del último valor sensado, que se encuentra almacenado en la base de datos.

Los archivos de Asterisk para ejecutar los AGI siempre deben estar en la siguiente dirección "cd /var/lib/asterisk/agi-bin". Por lo tanto se ingresa al directorio "cd /var/lib/asterisk/agi-bin" y se crea el archivo humedad_ambiente.php.

```
;Primero le dice al sistema que va a utilizar un intérprete para ejecutar un script en lenguaje php, -q desactiva los mensajes de error que puede enviar html.
#!/usr/bin/php -q
<?php
; Se llama a las librerías de AGI
require('phpagi.php');
;Se utiliza para activar la bandera de enviar todo lo que se realice a la consola como modo de depuración
error_reporting(E_ALL);
;Se crea una instancia de la clase AGI
$agi = new AGI();
$agi-> answer();

;Se hace la conexión a la base de datos
$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

;Se selecciona la tabla temperatura_agua y la fecha fecha_tagua como se encuentra declarado en la BD y se pide que lea únicamente el último dato con la sentencia "desc", y se entregan los resultados.
$query="SELECT * FROM temperatura_agua order by fecha_tagua desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

;Mediante la sentencia text2wav se pide que convierta el texto en audio y menciona el nombre del sensor.
$agi->text2wav("sensor de temperatura de agua");
$query="SELECT * FROM temperatura_agua order by fecha_tagua desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

;Finalmente lee el último valor sensado en la base de datos y cuelga la llamada.
$agi->text2wav("la temperatura del agua es $_re[gradC] grados... en la fecha $_re[fecha_tagua]");
$agi-> hangup();
?>
```

Finalmente se dan permisos a los archivos, mediante el comando: "sudo chmod +x /var/lib/asterisk/agi-bin/ humedad_ambiente.php".

Se realiza el mismo proceso para el resto de sensores, tomar en cuenta los nombres de la tablas y variables, para que no exista errores en la lectura de los datos. En el (Anexo C - literal c), se puede visualizar la programación para la lectura del último valor almacenado en la base de datos, de cada sensor utilizado en el desarrollo del presente proyecto.

4.30 Lectura de la información almacenada en la base de datos mediante Asterisk cuando un valor se encuentra fuera de rango.

A continuación se detalla la programación utilizada para la lectura del último valor sensado que se encuentra fuera de rango, almacenado en la base de datos.

Se ingresa al directorio "cd /var/lib/asterisk/agi-bin" y se crea el archivo leereportempagua.php

```
;Primero le dice al sistema que va a utilizar un intérprete para ejecutar un script en lenguaje php, -q desactiva los mensajes de error que puede enviar html.
#!/usr/bin/php -q
<?php
; Se llama a las librerías de AGI
require('phpagi.php');
;Se utiliza para activar la bandera de enviar todo lo que se realice a la consola como modo de depuración
error_reporting(E_ALL);
;Se crea una instancia de la clase AGI
$agi = new AGI();
$agi-> answer();

;Se hace la conexión a la base de datos
$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

;Se selecciona la tabla temperatura_agua y la fecha fecha_tagua como se encuentra declarado en la base de datos y se pide que lea únicamente el último dato con la sentencia "desc" pero cuando este no se encuentre  28°C y 32°C, finalmente se  entregan los resultados.
$query="SELECT * FROM temperatura_agua WHERE gradC not BETWEEN 28 and 32 order by fecha_tagua desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

;Mediante la sentencia text2wav se pide que convierta el texto en audio y menciona el nombre del sensor.
$agi->text2wav("sensor de temperatura de agua");
$agi->text2wav("El ultimo valor sensado fuera de rango se detalla a continuacion");
$query="SELECT * FROM temperatura_agua WHERE gradC not BETWEEN 28 and 32 order by fecha_tagua desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

;Finalmente lee el último valor sensado en la base de datos y cuelga la llamada.
$agi->text2wav("la temperatura del agua es $_re[gradC] grados...  en la fecha $_re[fecha_tagua]");
$agi-> hangup();
?>
```

Finalmente se dan permisos a los archivos, mediante el comando: "sudo chmod +x /var/lib/asterisk/agi-bin/ leereportempagua.php".

Se realiza el mismo proceso para el resto de sensores, tomar en cuenta los nombres de la tablas y variables, para que no exista errores en la lectura de los datos. En el (Anexo C - literal d), se puede visualizar la programación para la lectura del último valor almacenado fuera de rango, de cada sensor utilizado en el desarrollo del presente proyecto.

4.31 Archivo de configuración sip.conf y extensions.conf

En Asterisk hay dos archivos de configuración muy importantes, "sip.conf" y "extensions.conf", el primero sirve para configurar todo lo relacionado con el protocolo SIP (Session Initiation Protocol), se inicia con una sección [general] en la que se realiza la configuración para todos los usuarios. En el archivo extensions.conf, tiene como objetivo definir el dialplan que seguirá la central telefónica para el contexto declarado en el archivo sip.conf.

a) Archivo de configuración sip.conf

Primero se ingresa a Asterisk para poder configurar el archivo sip.conf:

- sudo nano sip.conf.

En el presente proyecto se creó el contexto gsm1, en el cual se configuran los parámetros de lenguaje, tipo, host, una contraseña entre otros, como se puede visualizar en la figura 4.53.

Figura 4.53: Archivo de configuración sip.conf.
Fuente: Investigador

b) Archivo de configuración extensions.conf

El archivo extensions.conf, es el corazón de Asterisk, aquí se declaran las extensiones a donde se puede llamar, es decir tiene como objetivo definir el dialplan o el plan de

numeración que seguirá la central telefónica para el contexto declarado en el archivo sip.conf.

Para configurar el archivo extensions.conf se dirige a la siguiente ruta.

- sudo nano extensions.conf

En el archivo extensions.conf se configuró la extensión "1" que dirige al usuario a un IVR, denominado ivr-soporte, y con la extensión "2" dirige al ivr-reporte.

```
[goip]
exten => 1,1,Goto(ivr-soporte,s,1)
exten => 2,1,Goto(ivr-reporte,s,1)
```

Figura 4.54: IVR soporte e IVR reporte.
Fuente: Investigador

4.31.1 Grabación de un menú IVR

Se crea una extensión para grabar el mensaje de bienvenida, en el presente proyecto se creó la extensión 9991 para realizar dicha función, se coloca prioridad "1" para que lo primero que realice sea contestar la llamada con la sentencia Answer(), a continuación espera 0.5 segundos, se graba el mensaje con el parámetro Record(menu_bienvenida.gsm) con el nombre que se desee dar al mensaje seguido de .gsm, en este caso menú_bienvenida.gsm, a continuación se espera 0.5 segundos, a continuación se verifica que la grabación este correcta con el parámetro Playback y el nombre grabado anteriormente es decir, Playback(menu_bienvenida), no hace falta colocar .gsm porque hace referencia a la grabación realizada en Record, y finalmente se cuelga la llamada.

```
;App para grabar Mensaje IVR
exten => 9991,1,Answer()
exten => 9991,n,Wait(0.5)
exten => 9991,n,Record(menu_bienvenida.gsm)
exten => 9991,n,Wait(0.5)
exten => 9991,n,Playback(menu_bienvenida)
exten => 9991,n,Hangup()
```

Figura 4.55: App para grabar mensaje IVR
Fuente: Investigador

a) IVR-soporte

En el ivr-soporte, se crea una extensión "s" con prioridad "1" y para contestar la llamada con la sentencia Answer(), a continuación se crea otra extensión "s" con prioridad "1" espera 0.5 segundos con la sentencia Wait(0.5), se crea otra extensión "s" con prioridad "1" que reproduce el menú de bienvenida con la sentencia Background(menu_bienvenida) y finalmente se crea otra extensión "s" con prioridad

“1” que espera 5 segundos a que se ingrese una de las opciones, con la sentencia WaitExten(5).

El ivr-soporte consta de 5 extensiones que se detallan a continuación:
La extensión “1” hace el llamado al archivo humedad_ambiente.php que realiza la lectura del último dato almacenado por el sensor de humedad ambiental DHT22, de la base de datos.
La extensión “2” hace el llamado al archivo iluminacion.php que realiza la lectura del último dato almacenado por el sensor de iluminación BH1750, de la base de datos.
La extensión “3” hace el llamado al archivo nivel_pH.php que realiza la lectura del último dato almacenado por el sensor de pH SEN0161, de la base de datos.
La extensión “4” hace el llamado al archivo temperatura_agua.php que realiza la lectura del último dato almacenado por el sensor de temperatura de agua DS18B20, de la base de datos.
La extensión “5” hace el llamado al archivo temperatura_ambiente.php que realiza la lectura del último dato almacenado por el sensor de temperatura ambiental DHT22, de la base de datos.

```
;CREAR IVR
[ivr-soporte]
exten => s,1,Answer()
exten => s,n,Wait(0.5)
exten => s,n,Background(menu_bienvenida)
exten => s,n,WaitExten(5)
;Leer último dato sensado
exten => 1,1,AGI(humedad_ambiente.php)
exten => 1,n,Goto(s,1)
exten => 2,1,AGI(iluminacion.php)
exten => 2,n,Goto(s,1)
exten => 3,1,AGI(nivel_pH.php)
exten => 3,n,Goto(s,1)
exten => 4,1,AGI(temperatura_agua.php)
exten => 4,n,Goto(s,1)
exten => 5,1,AGI(temperatura_ambiente.php)
exten => 5,n,Goto(s,1)
exten => *,1,Goto(s,1)
exten => t,1,Playback(thank-you-for-calling)
exten => t,n,Hangup()
exten => i,1,Playback(pbx-invalid)
exten => i,n,Goto(s,1)
```

Figura 4.56: IVR soporte
Fuente: Investigador

Al realizar una llamada se puede dar tres casos, el primero es que el usuario haya elegido una extensión entre “1” y “5” y realice la sentencia correspondiente.

El segundo caso es que el usuario digite una extensión que no se encuentra designada en el IVR, y para ello se crea una extensión "i" con prioridad "1" que reproduce el mensaje "Lo siento esta no es una extensión válida inténtelo nuevamente", seguido se crea otra extensión "i" con prioridad "n" que genera un salto al inicio del IVR para reproducir nuevamente el menú de bienvenida. Y finalmente el tercer caso es que no seleccione ninguna tecla entonces se crea una extensión "t" con prioridad "1" que reproduce el mensaje "gracias por llamar" y finalmente crea otra extensión "t" con prioridad "n" que cuelga la llamada.

b) IVR-reporte

El ivr-reporte consta de 5 extensiones que se detallan a continuación:

La extensión "1" hace el llamado al archivo leereporhumamb.php que realiza la lectura del último dato almacenado por el sensor de humedad ambiental DHT22, cuando este dato se encuentra fuera de rango, es decir que el dato proporcionado por este sensor no está entre 25% y 65%.

La extensión "2" hace el llamado al archivo leereporilum.php que realiza la lectura del último dato almacenado por el sensor de iluminación BH1750, cuando este dato se encuentra fuera de rango, es decir que el dato proporcionado por este sensor es menor a 25 lux entre las 7h00 y 18h00.

La extensión "3" hace el llamado al archivo leereporph.php que realiza la lectura del último dato almacenado por el sensor de pH SEN0161, cuando este datos se encuentra fuera de rango, es decir que el dato proporcionado por este sensor no está entre 6.5 y 7.5.

La extensión "4" hace el llamado al archivo leereportempagua.php que realiza la lectura del último dato almacenado por el sensor de temperatura de agua DS18B20, cuando este datos se encuentra fuera de rango, es decir que el dato proporcionado por este sensor no está entre 28 °C y 32°C.

La extensión "5" hace el llamado al archivo leereportempamb.php que realiza la lectura del último dato almacenado por el sensor de temperatura ambiental DHT22, cuando este datos se encuentra fuera de rango, es decir que el dato proporcionado por este sensor no está entre 1°C y 25°C.

```
[ivr-reporte]
exten => s,1,Answer()
exten => s,n,Wait(0.5)
exten => s,n,Background(holai)
exten => s,n,WaitExten(5)
;Leer el último dato fuera de rango
exten => 1,1,AGI(leereporhumamb.php)
exten => 1,n,Goto(s,1)
exten => 2,1,AGI(leereporilum.php)
exten => 2,n,Goto(s,1)
exten => 3,1,AGI(leereporph.php)
exten => 3,n,Goto(s,1)
exten => 4,1,AGI(leereportempagua.php)
exten => 4,n,Goto(s,1)
exten => 5,1,AGI(leereportempamb.php)
exten => 5,n,Goto(s,1)
exten => *,1,Goto(s,1)
exten => t,1,Playback(thank-you-for-calling)
exten => t,n,Hangup()
exten => i,1,Playback(pbx-invalid)
exten => i,n,Goto(s,1)
```

Figura 4.57: IVR reporte.
Fuente: Investigador

Para cerciorarse del funcionamiento de la central telefónica, se descarga un teléfono para realizar una llamada mediante VoIP, con el teléfono "3CXPhone", se lo ha realizado en el presente proyecto.

Figura 4.58: Teléfono VoIP 3CXPhone.
Fuente: Investigador

A continuación se registra el nombre y el dominio configurado en el archivo sip.conf., como se detalla en la figura 4.59.

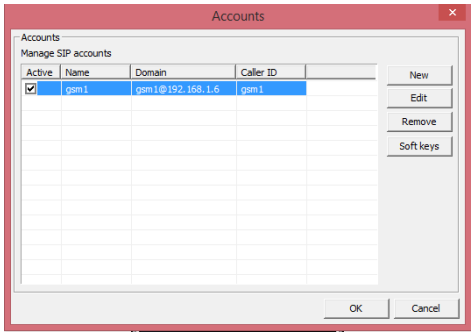


Figura 4.59: Configuración del teléfono VoIP 3CXPhone.
Fuente: Investigador

Al marcar “1” el sistema da la bienvenida y le pide que ingrese una extensión para consultar la información almacenados en la base de datos, y de esta manera se cerciora del funcionamiento de la central telefónica.

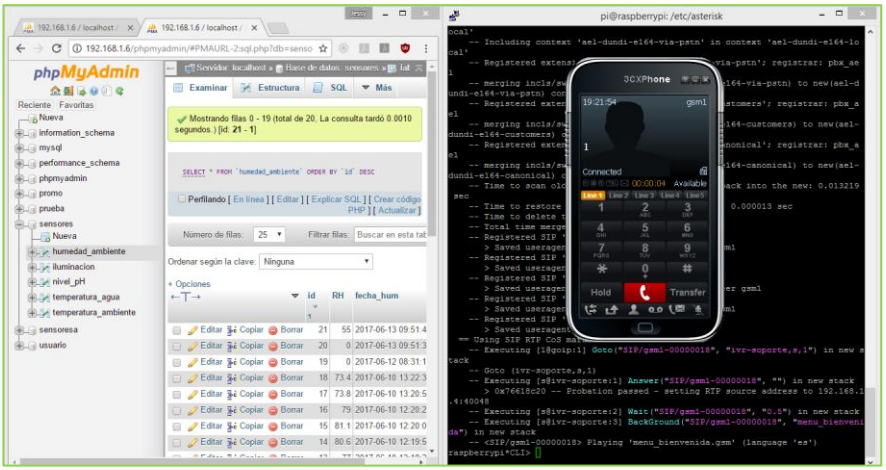


Figura 4.60: Llamada a la central telefónica.
Fuente: Investigador

4.32 GoIP

El gateway GSM llamado GoIP permite comunicar la red de teléfono móvil, con la red de telefonía VoIP instalada y configurada como se detalló anteriormente. En el GoIP se insertó una tarjeta sim y a continuación se configuró con Asterisk para usarlo como troncal para poder hacer y recibir llamadas.

Una vez insertada la tarjeta sim se ingresa al GoIP mediante la dirección configurada de fábrica que es 192.168.8.1, con el nombre de usuario “admin” y contraseña “admin” se ingresa al GoIP para realizar las configuraciones necesarias.

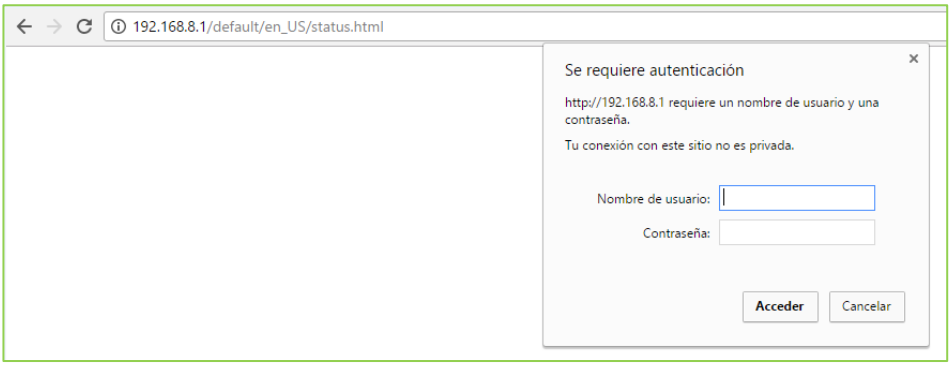


Figura 4.61: Ingreso a GoIP.
Fuente: Investigador

Para tener acceso al GoIP sin la necesidad del cable de red se ingresa a Configuraciones, Network y se le asigna una dirección Estática, como se puede apreciar en la figura 4.62.

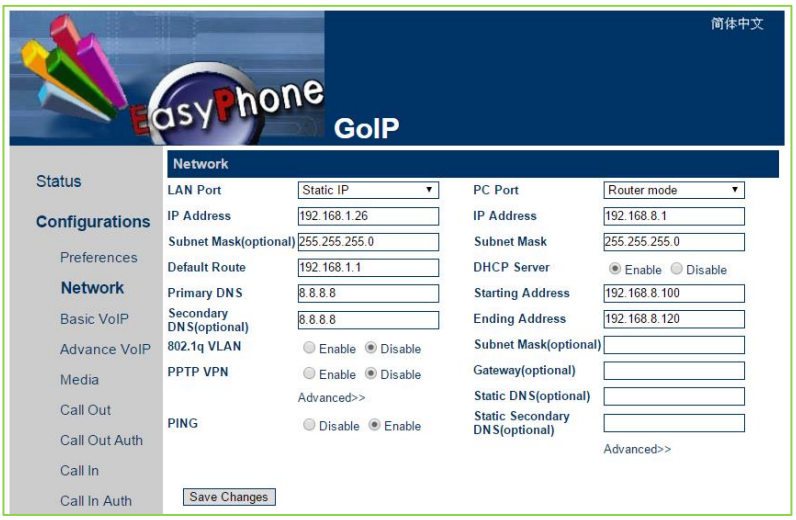


Figura 4.62: Asignación de IP Estática en GoIP.
Fuente: Investigador

A continuación se ingresa a Basic VoIP y se realiza la configuración con la información utilizada en Asterisk, que se colocó en el archivo sip.conf.

Endpoint Type= SIP Phone

Config Mode=Single Server Mode

Phone Number= gsm1

Display Name= gsm1

Authentication ID=gsm1

Password=••••

SIP Proxy=192.168.1.6 (Dirección IP de Asterisk)

SIP Registrar=192.168.1.6 (Dirección IP de Asterisk)

Re-register Period(s)=60

Figura 4.63: Configuración de Basic VoIP.
Fuente: Investigador

Finalmente para comprobar que todo fue configurado correctamente se ingresa a Status y se puede visualizar que se tiene comunicación con VoIP.

Figura 4.64: Verificación del Status en el GoIP
Fuente: Investigador

Se realiza la llamada a la SIM insertada en el GoIP y se elige la extensión que se desee para conocer la información almacenada en la base de datos.

4.33 Instalación de la Librería GPIO

Para poder utilizar los puertos GPIO de la Raspberry PI3 con Python, es necesario instalar las librerías GPIO, que se lo realiza de la siguiente manera:

Se descarga la librería GPIO y mediante FileZilla se transfiere el archivo descargado en la ruta “cd /usr/src”, como se visualiza en la figura 4.65.

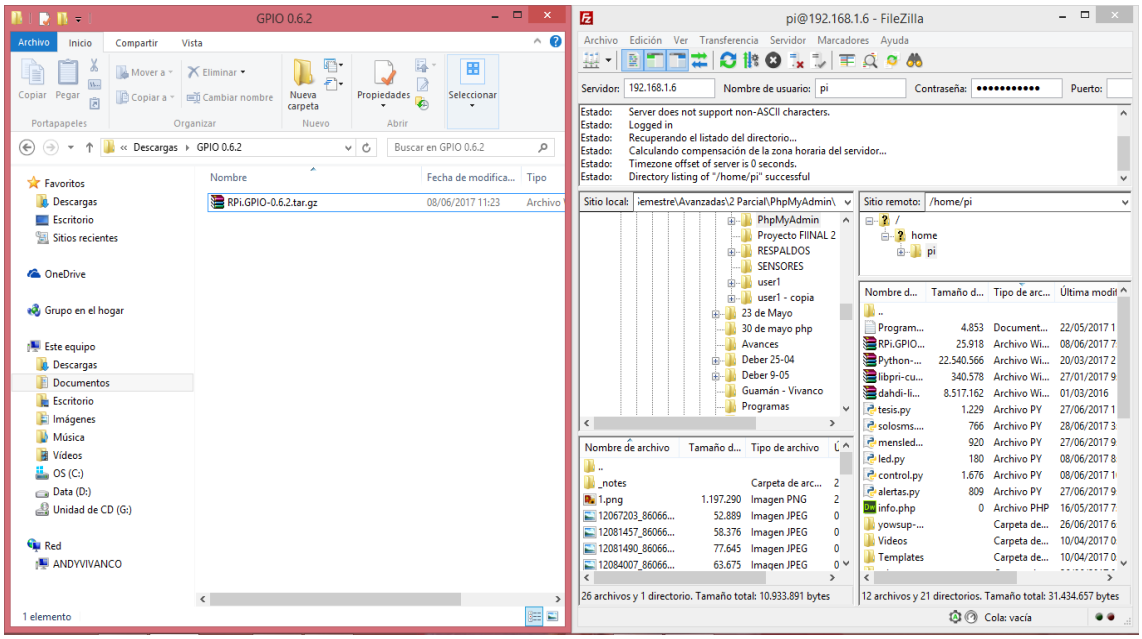


Figura 4.65: Transferencia de la librería GPIO.
Fuente: Investigador

A continuación se descomprime la librería GPIO.

- sudo tar zxvf RPI.GPIO-0.6.2.tar.gz.

Se ingresa a la carpeta descomprimida para proceder con la instalación de la librería.

- cd RPI.GPIO-0.6.2/.

Finalmente se instalan las librerías necesarias para la utilización de los puertos GPIO en la Raspberry con Python.

- sudo apt-get install python-dev.
- sudo python setup.py install.

4.34 Creación de una cuenta en Pushetta para recibir notificaciones

Para recibir las notificaciones cuando un valor sensado no se encuentra en el rango establecido, se ha realizado un sistema de alertas que notifique cuando el ventilador, la lámpara, el calentador de agua o la lámpara fluorescente se ha activado.

Pushetta es un programa que permite la creación de canales para enviar y recibir mensajes, estos mensajes pueden ser alertas que notifican al usuario cuando en el sistema se ha realizado un cambio de estado, es decir se ha activado una alerta.

Para crear una cuenta en Pushetta se realiza el siguiente procedimiento:

Primero se ingresa a la página oficial de Pushetta “www.pusheta.com”, que tiene la siguiente presentación.

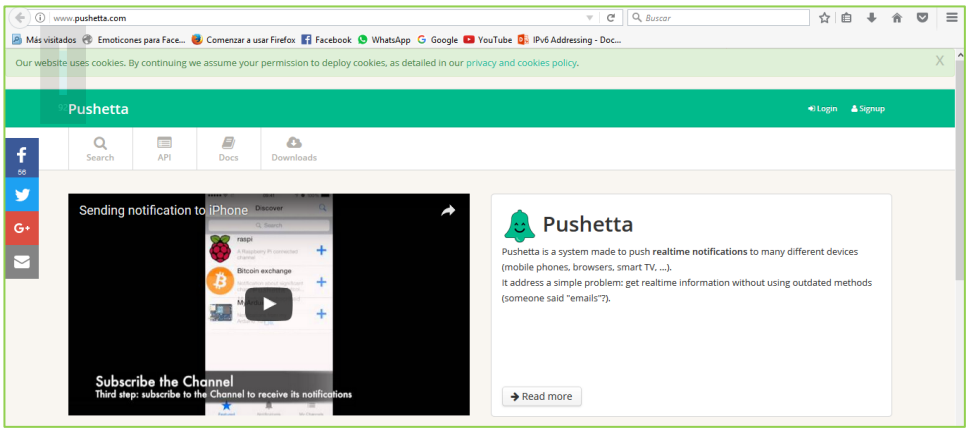


Figura 4.66: Ingreso a la página oficial de Pushetta.
Fuente: Investigador

A continuación se crea una cuenta con un correo electrónico, un nombre de usuario y una contraseña, que permite registrarse en Pushetta.

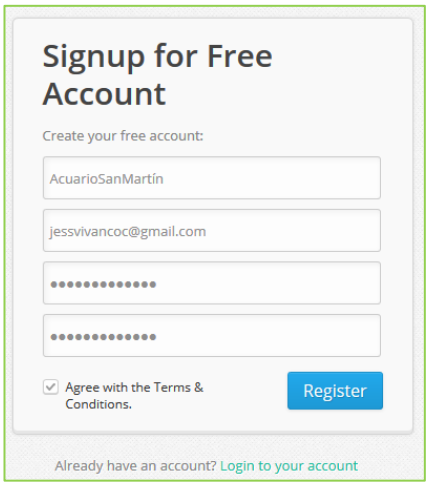


Figura 4.67: Creación de una cuenta en Pushetta.
Fuente: Investigador

A continuación se confirma la creación de la cuenta, ingresando al correo electrónico.

Una vez creada la cuenta, se crea un nuevo canal para el uso de la aplicación, como se visualiza en la figura 4.68.

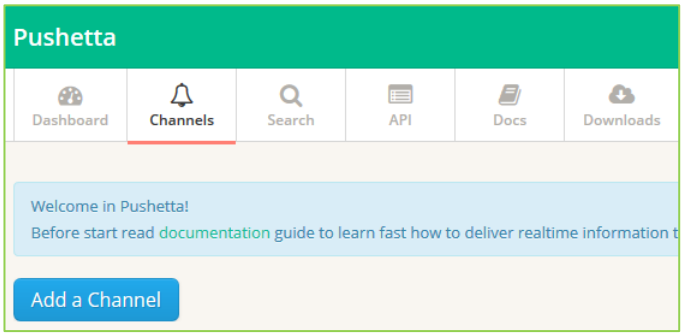


Figura 4.68: Creación de canal en Pushetta.
Fuente: Investigador

En la creación del canal, se ingresa el nombre del canal, una descripción de lo q se va a realizar y finalmente se selecciona crear.

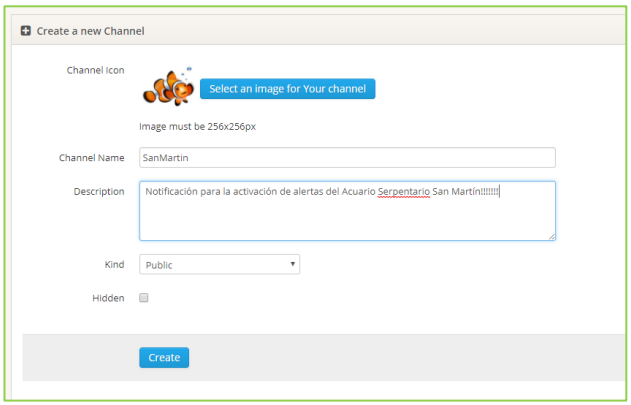


Figura 4.69: Ingreso de datos en Pushetta.
Fuente: Investigador

El canal está listo para utilizarse y se puede registrar al canal desde cualquier celular inteligente, únicamente buscando el canal por el nombre "SanMartin".

El canal cuenta con un código que permite interactuar con Pushetta desde arduino, php, python entre otros. El código conocido como "APIKEY" se lo puede encontrar en "Dashboard" de Pusheta.

Figura 4.70: API Key en Pushetta.
Fuente: Investigador

4.35 Registro en el Canal "SanMartin" mediante Pushetta.

En un teléfono inteligente se descarga la aplicación Pushetta por medio de "Play Store", como se visualiza en la figura 4.71.

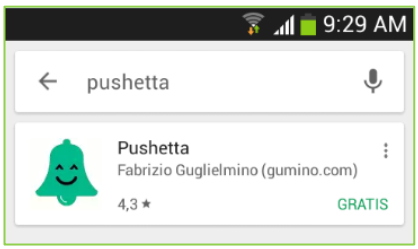


Figura 4.71: Play Store en Pushetta.
Fuente: Investigador

A continuación se busca el Canal creado para poder recibir las notificaciones del sistema. En la opción "Add custom cannel", se busca el canal deseado.

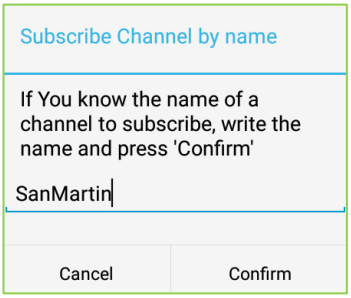


Figura 4.72: Subscripción en el canal "SanMartin"
Fuente: Investigador

Como se puede visualizar, se ha registrado en el canal "SanMartin". Y ya se puede recibir las notificaciones cuando el ventilador, la lámpara, el calentador de agua o la lámpara fluorescente se han activado.

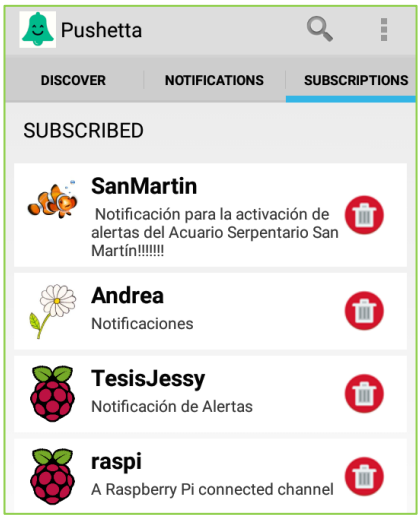


Figura 4.73: Canales subscritos en Pushetta.
Fuente: Investigador

4.36 Activación de Alertas

Para la activación de alertas es importante obtener únicamente el último dato almacenado en phpMyAdmin, para realizar la comparación de dicho dato con los rangos establecidos, en el requerimiento de las variables físicas que interviene en el Acuario, "San Martín", del pez "Óscar".

Se crea el archivo "ultimo_temperatura_agua.php" en la carpeta "acuario", primero se realiza la conexión con el usuario "arduin" y con la base de datos "sensores", a continuación se toma únicamente el último valor almacenado en la base de datos mediante la instrucción "desc" y se selecciona la tabla deseada en este caso, la tabla "temperatura_agua", y finalmente se recupera el valor obtenido y se lo imprime para poder visualizar si el valor ha sido el correcto.

```
<?php
$conex = mysql_connect('localhost','arduin','godinmylife');
mysql_select_db('sensores',$conex);
$selecciona = "SELECT * FROM temperatura_agua order by fecha_tagua desc";
$valortagua = mysql_query($selecciona);
$recupera=mysql_fetch_object($valortagua);
echo $recupera->gradC;
?>
```

Se realiza el mismo proceso para las demás tablas que se encuentran en la base de datos "sensores", tomar en cuenta los nombres de las tablas dependiendo de los sensores, para que no existan errores en la lectura de los datos. En el (Anexo C-literal e), se puede visualizar la programación de todas las tablas para obtener el último dato proporcionado por los sensores.

```
pi@raspberrypi:/var/www/html/acuario $ ls
conexion.php            temperatura_ambiental.php
humedad_ambiental.php   ultimo_humedad_ambiental.php
iluminacionn.php        ultimo_iluminacionn.php
nivelph.php             ultimo_ph.php
temperatura_aguaa.php   ultimo_temperatura_agua.php
temperatura_agua.php    ultimo_temperatura_ambiental.php
```

Figura 4.74: Scripts del último valor almacenado en la base de datos.
Fuente: Investigador

Para activar los puertos GPIO 17, 18, 22 y 27 de la Raspberry Pi3, se realiza la programación en Python. Primero se importan las librerías necesarias para la ejecución del código y activación de los puertos GPIO, el módulo "time" permite obtener la hora y fecha actual proporcionada por el servidor, el módulo "urllib2" permite a Python interactuar con páginas Web para adquirir datos o variables necesarias para la ejecución del programa.

A continuación se define los cuatro puertos que se utilizarán para la activación de las alertas, en este caso se ha activado los puertos 17, 18, 22 y 27.

Mediante los módulos "json" y "urllib2 se crea la función que usará el script para enviar las notificaciones a través de Pushetta. La instrucción while true, permite que se cree un ciclo para que siempre se ejecutando el código.

Se realiza el control para el sensor DHT22, si el dato obtenido por dicho sensor es mayor a 25°C, se activa el puerto GPIO 18, caso contrario el puerto permanece desactivado. El puerto GPIO 18 activa un ventilador que controla la temperatura ambiental.

Se realiza el control para el sensor DS18B20, si el dato obtenido por dicho sensor es menor a 28°C, se activa el puerto GPIO 27, caso contrario el puerto permanece desactivado. El puerto GPIO 27 activa un calentador de agua que controla la temperatura de la pecera.

Se realiza el control para el sensor BH1750, si el dato obtenido por dicho sensor es menor a 25 lux y se encuentra entre las 7h00 y 18h00, se activa el puerto GPIO 17, caso contrario el puerto permanece desactivado. El puerto GPIO 17 activa una lámpara que controla la luminosidad en el acuario.

Se realiza el control para el sensor SEN0161, si el dato obtenido por dicho sensor no se encuentra en el rango de 6.5 a 7.5 se activa el puerto GPIO 22, caso contrario el

puerto permanece desactivado. El puerto GPIO 22 activa una luz piloto que indica que el agua de la pecera debe ser cambiada.

Programación en Python para la activación de puertos GPIO en la Raspberry

```
#!/usr/bin/python
import RPi.GPIO as GPIO
import time
from urllib2 import urlopen
from datetime import datetime, time
import urllib2
import json

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

GPIO.setup(18,GPIO.OUT)
GPIO.setup(17,GPIO.OUT)
GPIO.setup(27,GPIO.OUT)
GPIO.setup(22,GPIO.OUT)

# Funcion que usa el script para enviar las notificaciones a Pushetta
def sendNotification(token, channel, message):
        data = {
                "body" : message,
                "message_type" : "text/plain"
        }

        req = urllib2.Request('http://api.pushetta.com/api/pushes/{0}/'.format(channel))
        req.add_header('Content-Type', 'application/json')
        req.add_header('Authorization', 'Token {0}'.format(token))

        response = urllib2.urlopen(req, json.dumps(data))

a = 0
b = 0
c = 0
d = 0
while True :

  #Control para el sensor de humedad y temperatura ambiental
  sensordht22 = urlopen('http://192.168.1.6:80/acuario/ultimo_temperatura_ambiental.php')
  tempdht22= float(sensordht22.read())
  if ( tempdht22 > 25 ) :
      GPIO.output(18, GPIO.LOW)
      a += 1
      if ( a == 1) :
          print("Alarma activada DHT22")
          sendNotification("4c744d705a1d416c522789577b496a177327329b", "SanMartin",
"Se ha activado el ventilador!!")
   else:
      GPIO.output(18, GPIO.HIGH)
      a = 0
      print("Alarma desactivada DHT22")
```

```
  #Control para el sensor de luz
  sensorluz = urlopen('http://192.168.1.6:80/acuario/ ultimo_iluminacionn.php')
  sensor1750 = float(sensorluz.read())
  tiempo = datetime.now()
  if ( sensor1750 < 25 and ( tiempo.hour >= 6 or tiempo.hour <= 18 ) ) :
      GPIO.output(22, GPIO.LOW)
      b += 1
      if ( b == 1) :
          print("Alarma activada LUZ")
          sendNotification("4c744d705a1d416c522789577b496a177327329b", "SanMartin",
"Se ha encendido la lampara")
  else:
      GPIO.output(22, GPIO.HIGH)
      b = 0
      print("Alarma desactivada LUZ")


  #Control para el sensor de temperatura de agua
  sensoragua = urlopen('http://192.168.1.6:80/acuario/ultimo_temperatura_agua.php')
  sensor18b20 = float(sensoragua.read())
  if ( sensor18b20 < 28  ) :
      GPIO.output(27, GPIO.LOW)
      c += 1
      if ( c == 1) :
          print("Alarma activada TEMP AGUA")
          sendNotification("4c744d705a1d416c522789577b496a177327329b", "SanMartin",
"Se ha encendido el calentador de agua")
  else:
      GPIO.output(27, GPIO.HIGH)
      c = 0
      print("Alarma desactivada TEMP AGUA")

  #Control para el sensor de pH
  sensorph = urlopen('http://192.168.1.6:80/acuario/ ultimo_ph.php')
  ph = float(sensorph.read())
  if ( (ph < 6.5) or (ph > 7.5)) :
      GPIO.output(17, GPIO.LOW)
      d += 1
      if ( d == 1) :
          print("Alarma activada ph")
          sendNotification("4c744d705a1d416c522789577b496a177327329b", "SanMartin",
"Porfavor cambiar de agua la pecera")
  else:
      GPIO.output(17, GPIO.HIGH)
      d = 0
      print("Alarma desactivada ph")
```

4.37 Implementación del prototipo

Se realiza el diseño del circuito que permite la interacción entre el Arduino y los sensores, Raspberry Pi3 y Módulo relé.

En la figura 4.75 se puede observar el diseño del circuito realizado en PCB-Wizard, que permite la conexión de los elementos y dispositivos electrónicos utilizados en el presente proyecto.

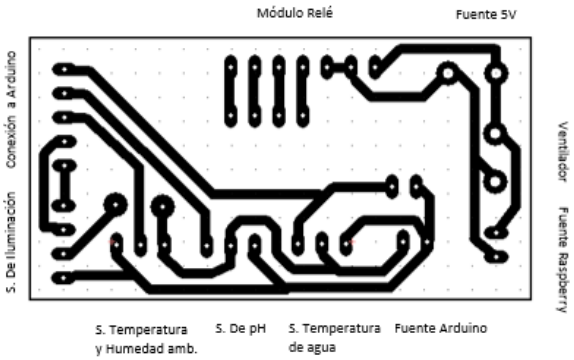


Figura 4.75: Circuito impreso para la placa.
Fuente: Investigador

Para la implementación de la placa en baquelita, primero se imprime el circuito en papel couche, para luego aplicarlo sobre la placa de cobre y poder obtener las pistas, mediante una plancha., como se puede observar en la figura 4.76.

Figura 4.76: Planchado de la placa.
Fuente: Investigador

Luego de realizar el planchado del circuito por un tiempo prolongado se procede a retirar el papel de la baquelita, y a limpiar la placa, a continuación se coloca el ácido en un recipiente plástico, donde se sumerge a la placa, moviéndola continuamente de un lado hacia otro, para que se produzca la reacción química y obtener las pistar del circuito.

Figura 4.77: Introducción de la placa en el ácido férrico.
Fuente: Investigador

Realizar las perforaciones con un taladro, para ubicar los componentes, y luego soldarlos, como se muestra en la figura 4.78.

Figura 4.78: Soldadura de los elementos en la placa.
Fuente: Investigador

Se conecta al circuito: el sensor de pH, el sensor de temperatura ambiental, el sensor de luz y el sensor de temperatura de agua, como se muestra en la figura 4.79.

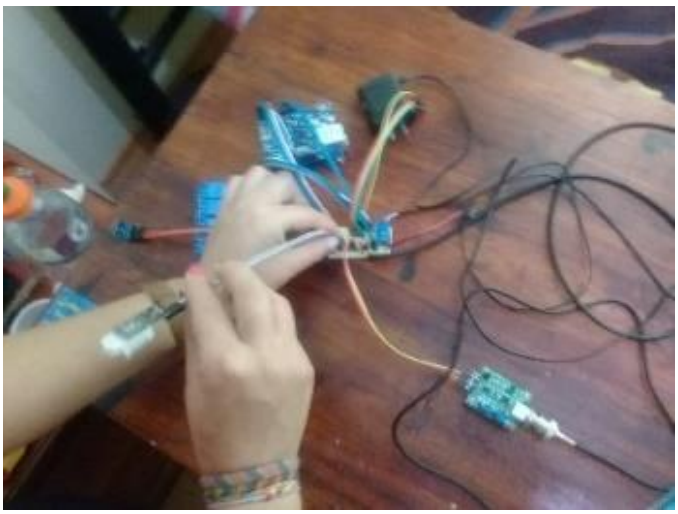


Figura 4.79: Conexión de cableado a la placa.
Fuente: Investigador

Finalmente se colocan los sensores y los equipos eléctricos en el Acuario para realizar pruebas de funcionamiento.

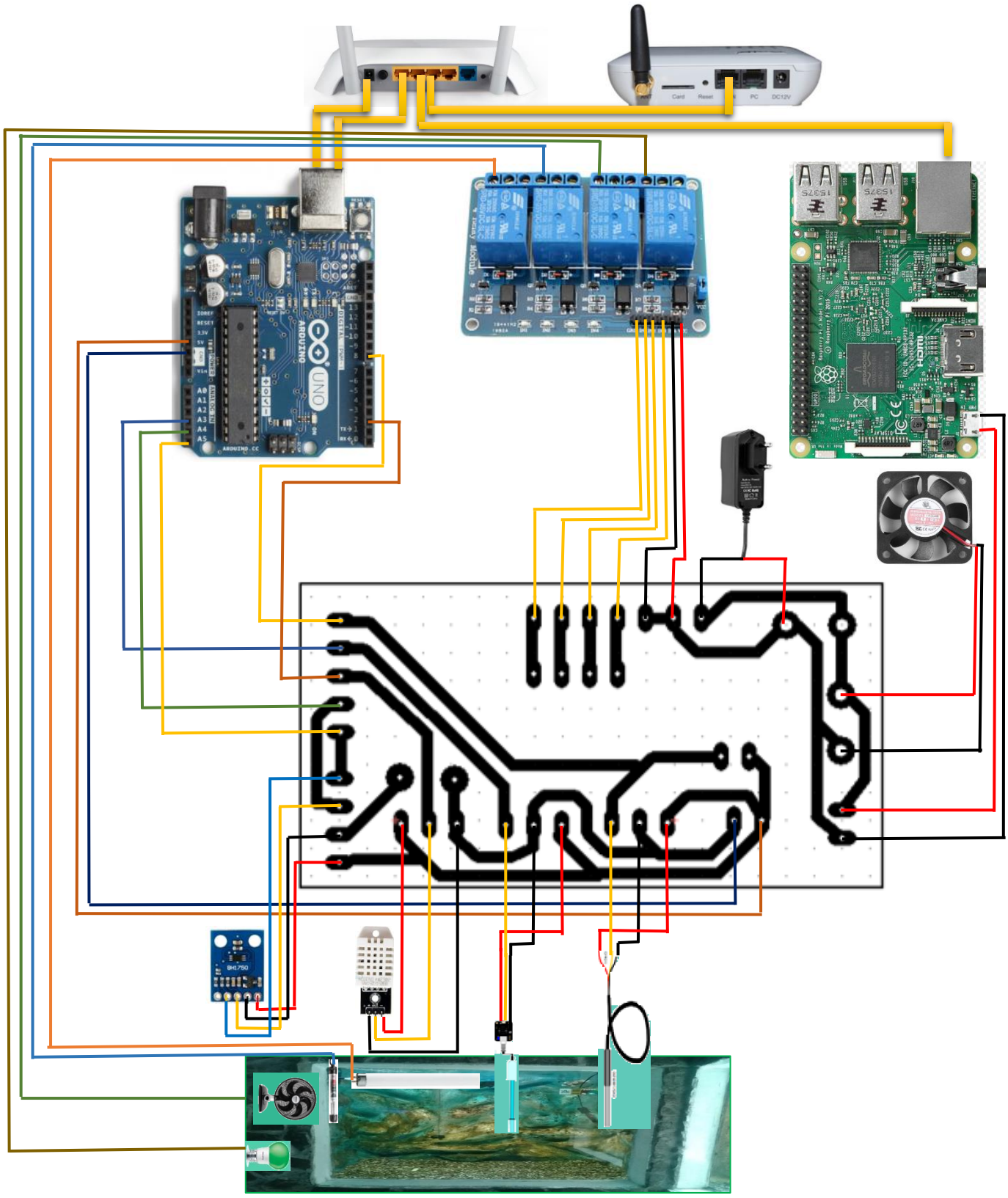


Figura 4.80: Sistema de Control Electrónico de Acuarios
Fuente Investigador

En la figura 4.81 se puede observar, todo el diseño del prototipo del Sistema de Control Electrónico de Acuarios, implementado en el Acuario Serpentario “San Martín” de Baños, de la provincia de Tungurahua.

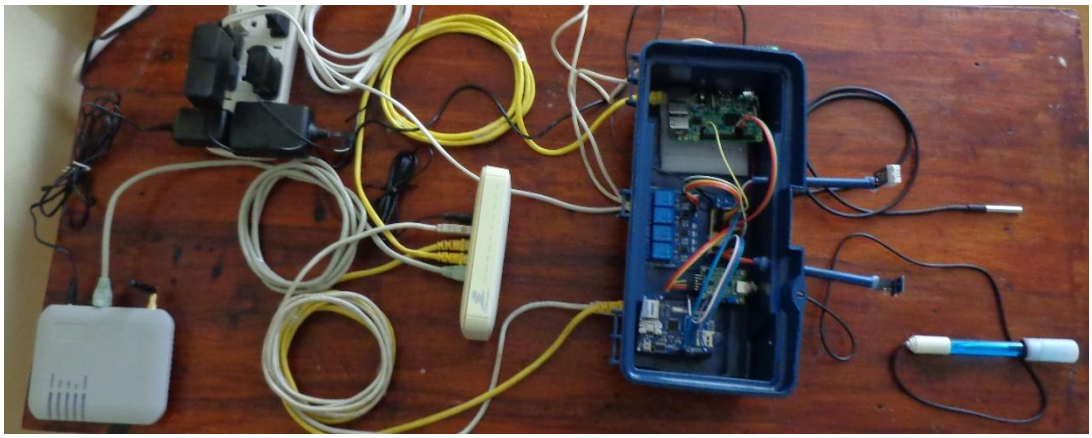


Figura 4.81: Diseño físico del Sistema de Control Electrónico de Acuario "San Martín".
Fuente Investigador

4.38 Pruebas del prototipo

4.38.1 Prueba de envío de los valores sensados mediante arduino

Se ejecuta el programa “solodatos.ino”, el mismo que permite observar los datos obtenidos de los sensores de temperatura de agua, temperatura ambiental, humedad ambiental, iluminación y pH.

En la figura 4.82 se puede visualizar la información obtenida de los sensores mediante arduino.

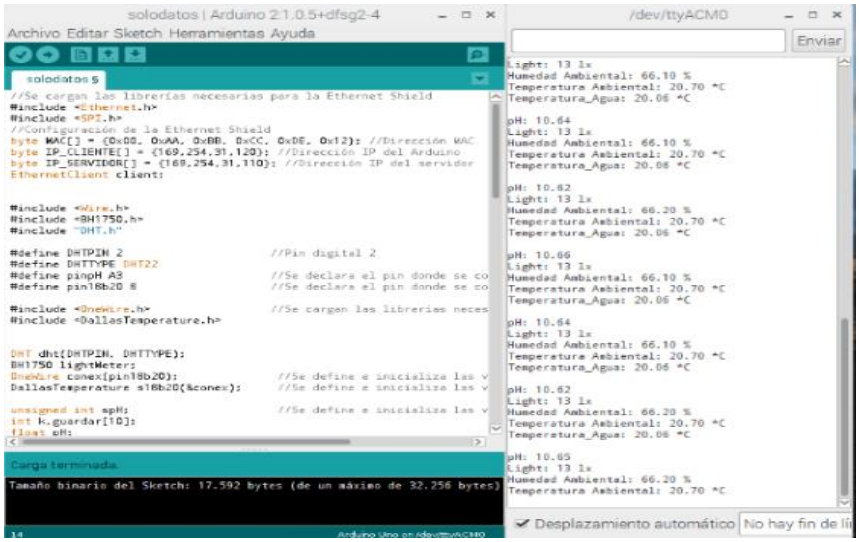


Figura 4.82: Prueba de envío de datos sensados mediante arduino.
Fuente Investigador

4.38.2 Prueba de envío de los valores sensados desde arduino a la base de datos

Se realiza la conexión de red entre la Raspberry Pi3 y el Arduino, a continuación se ejecuta el código correspondiente para el envío de los valores sensados a la base de datos y se cerciora de que la información obtenida de los sensores sea la misma que se almacena en phpMyAdmin.

En la figura 4.83 se puede visualizar que los valores sensados son enviados a la base de datos, y se puede cerciorar en arduino mediante el mensaje "Dato enviado".

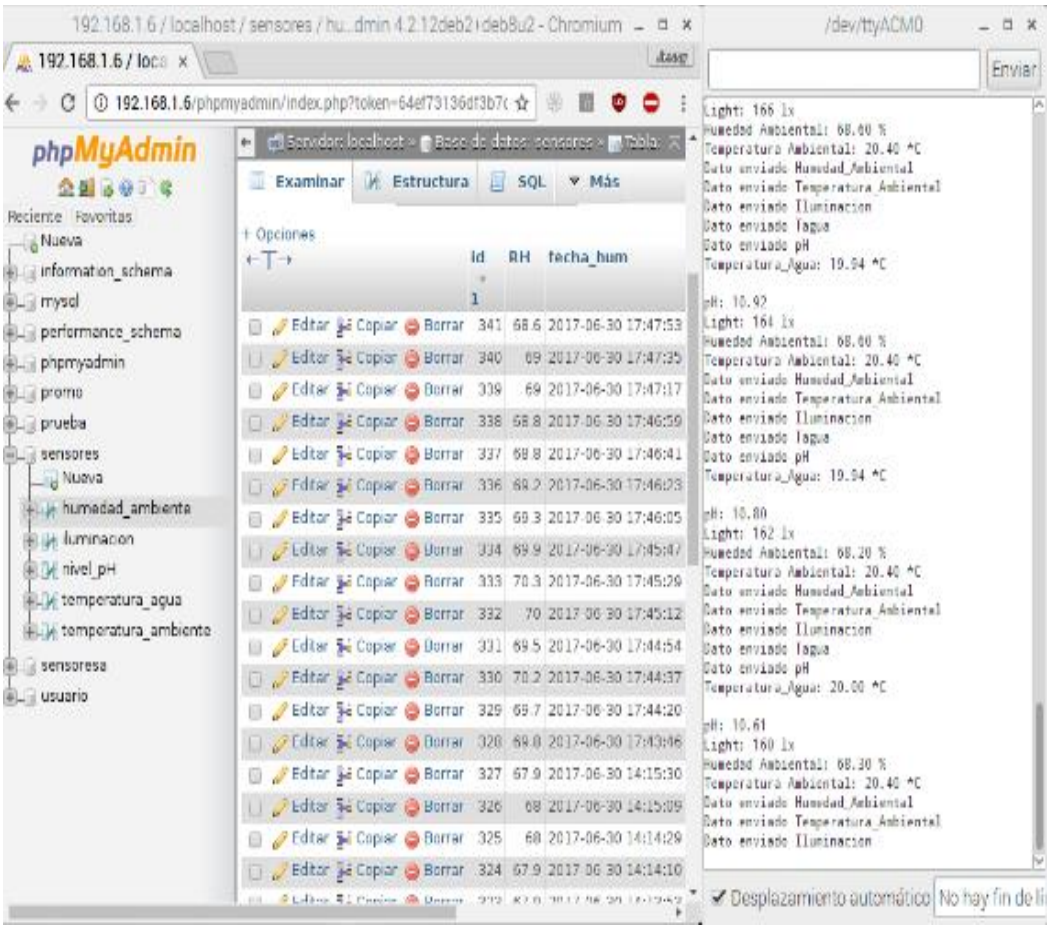


Figura 4.83: Prueba del envío de la información desde arduino a la base de datos.
Fuente: Investigador

4.38.3 Pruebas de llamada a la central telefónica

Al marcar la extensión "1" la central telefónica direcciona al IVR-soporte, a continuación contesta la llamada y se reproduce el mensaje de bienvenida, como se puede apreciar en la figura 4.84.

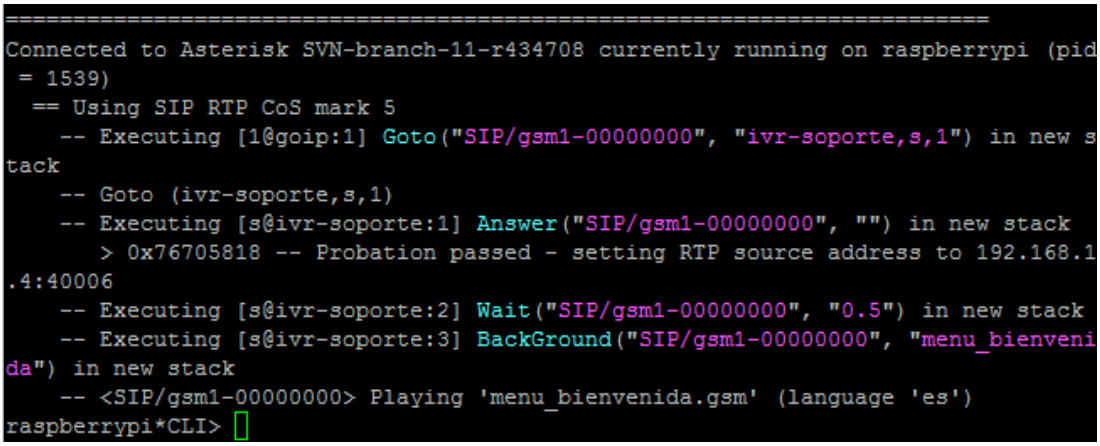


Figura 4.84: Prueba del menú de bienvenida en Asterisk.
Fuente: Investigador

Si no se digita ninguna extensión en el lapso de 5 segundos luego de haber culminado la reproducción del menú de bienvenida, la central telefónica cuelga la llamada mediante un mensaje "gracia por llamar", como se puede visualizar en la figura 4.85.

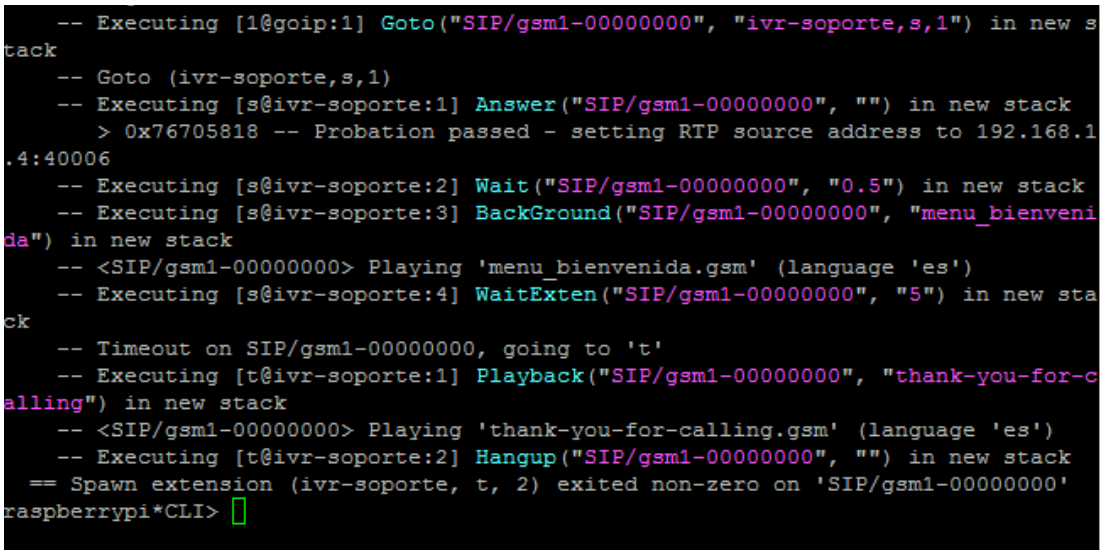


Figura 4.85: Prueba de tiempo máximo de respuesta al digitar una extensión.
Fuente: Investigador

Si se digita una extensión que no está disponible, la central telefónica reproduce un mensaje comunicando que la extensión marcada no está disponible y que lo intente nuevamente, a continuación se reproduce el mensaje de bienvenida para escuchar las extensiones que se encuentran disponibles.

Figura 4.86: Prueba de marcación a las extensiones del IVR-soporte.
Fuente: Investigador

a) Prueba de llamada al IVR-soporte

En el IVR-soporte al marcar la extensión "1", la central telefónica hace el llamado al archivo humedad_ambiente.php que contiene la programación necesaria para que la central telefónica interactúe con la base de datos e informe el último valor almacenado del sensor de humedad ambiental.

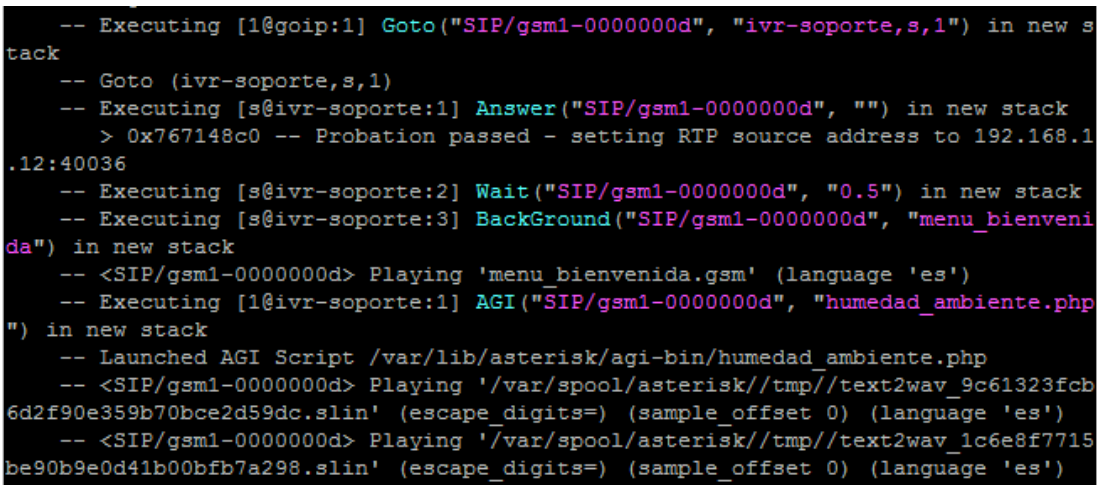


Figura 4.87: Prueba de marcación a la extensión "1", en el IVR-soporte.
Fuente: Investigador

En el IVR-soporte al marcar la extensión "2", la central telefónica hace el llamado al archivo iluminación.php que contiene la programación necesaria para que la central telefónica interactúe con la base de datos e informe el último valor almacenado del sensor de iluminación.

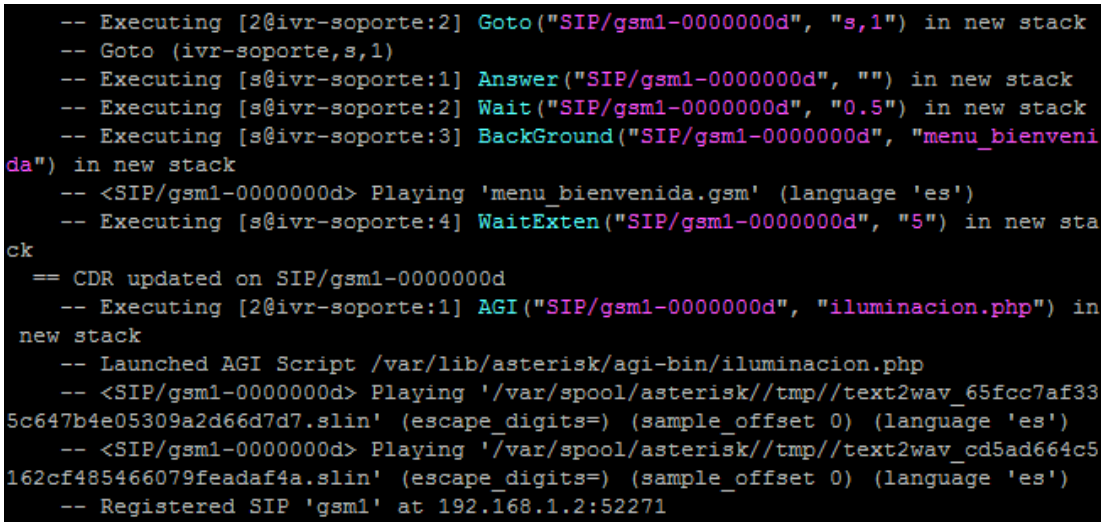


Figura 4.88: Prueba de marcación a la extensión "2", en el IVR-soporte.
Fuente: Investigador

En el IVR-soporte al marcar la extensión "3", la central telefónica hace el llamado al archivo nivel_pH.php que contiene la programación necesaria para que la central telefónica interactúe con la base de datos e informe el último valor almacenado del sensor de pH.

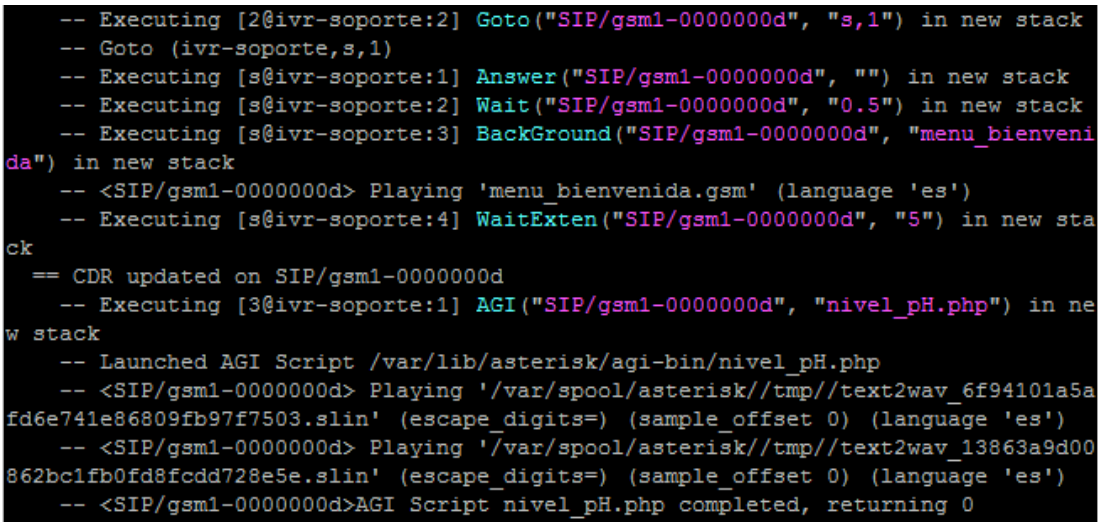


Figura 4.89: Prueba de marcación a la extensión "3", en el IVR-soporte.
Fuente: Investigador

En el IVR-soporte al marcar la extensión "4" la central telefónica hace el llamado al archivo temperatura_agua.php que contiene la programación necesaria para que la central telefónica interactúe con la base de datos e informe el último valor almacenado del sensor de temperatura del agua.

```
    -- Executing [3@ivr-soporte:2] Goto("SIP/gsm1-0000000d", "s,1") in new stack
    -- Goto (ivr-soporte,s,1)
    -- Executing [s@ivr-soporte:1] Answer("SIP/gsm1-0000000d", "") in new stack
    -- Executing [s@ivr-soporte:2] Wait("SIP/gsm1-0000000d", "0.5") in new stack
    -- Executing [s@ivr-soporte:3] BackGround("SIP/gsm1-0000000d", "menu_bienveni
da") in new stack
    -- <SIP/gsm1-0000000d> Playing 'menu_bienvenida.gsm' (language 'es')
    -- Executing [4@ivr-soporte:1] AGI("SIP/gsm1-0000000d", "temperatura_agua.php
") in new stack
    -- Launched AGI Script /var/lib/asterisk/agi-bin/temperatura_agua.php
    -- <SIP/gsm1-0000000d> Playing '/var/spool/asterisk//tmp//text2wav_1b21b4b046
1906c1ad01c8b8724498d3.slin' (escape_digits=) (sample_offset 0) (language 'es')
    -- <SIP/gsm1-0000000d> Playing '/var/spool/asterisk//tmp//text2wav_942ff3bc7f
a755deb5b5913e09946fe1.slin' (escape_digits=) (sample_offset 0) (language 'es')
    -- Registered SIP 'gsm1' at 192.168.1.12:52888
```

Figura 4.90: Prueba de marcación a la extensión "4", en el IVR-soporte.

Fuente: Investigador

En el IVR-soporte al marcar la extensión "5", la central telefónica hace el llamado al archivo temperatura_ambiente.php que contiene la programación necesaria para que la central telefónica interactúe con la base de datos e informe el último valor almacenado del sensor de temperatura ambiental.

```
    -- Executing [4@ivr-soporte:2] Goto("SIP/gsm1-0000000d", "s,1") in new stack
    -- Goto (ivr-soporte,s,1)
    -- Executing [s@ivr-soporte:1] Answer("SIP/gsm1-0000000d", "") in new stack
    -- Executing [s@ivr-soporte:2] Wait("SIP/gsm1-0000000d", "0.5") in new stack
    -- Executing [s@ivr-soporte:3] BackGround("SIP/gsm1-0000000d", "menu_bienveni
da") in new stack
    -- <SIP/gsm1-0000000d> Playing 'menu_bienvenida.gsm' (language 'es')
    -- Executing [5@ivr-soporte:1] AGI("SIP/gsm1-0000000d", "temperatura_ambiente
.php") in new stack
    -- Launched AGI Script /var/lib/asterisk/agi-bin/temperatura_ambiente.php
    -- <SIP/gsm1-0000000d> Playing '/var/spool/asterisk//tmp//text2wav_f2c0eaaa8d
259720c44bc23cb69687e0.slin' (escape_digits=) (sample_offset 0) (language 'es')
    -- <SIP/gsm1-0000000d> Playing '/var/spool/asterisk//tmp//text2wav_de3e19e824
d547a3094b9eb27e6144a9.slin' (escape_digits=) (sample_offset 0) (language 'es')
    -- <SIP/gsm1-0000000d>AGI Script temperatura_ambiente.php completed, returnin
```

Figura 4.91: Prueba de marcación a la extensión "5", en el IVR-soporte.
Fuente: Investigador

b) Prueba de llamada al IVR-reporte

Cuando existe un valor sensado que no se encuentra en el rango establecido se activa una alerta que permite realizar un control remoto del sistema, para conocer cuando una alerta se ha activado por última vez se marca "2" y se selecciona la extensión que permite conocer dicho dato dependiendo del sensor que se elija.

En el IVR-reporte al marcar la extensión "1", la central telefónica hace el llamado al archivo leerreporthumamb.php que contiene la programación necesaria para que la

central telefónica interactúe con la base de datos e informe el último valor almacenado fuera de rango del sensor de humedad ambiental.

Figura 4.92: Prueba de marcación a la extensión "1", en el IVR-reporte.
Fuente: Investigador

En el IVR-reporte al marcar la extensión "2", la central telefónica hace el llamado al archivo leerreportilum.php que contiene la programación necesaria para que la central telefónica interactúe con la base de datos e informe el último valor almacenado fuera de rango del sensor de iluminación.

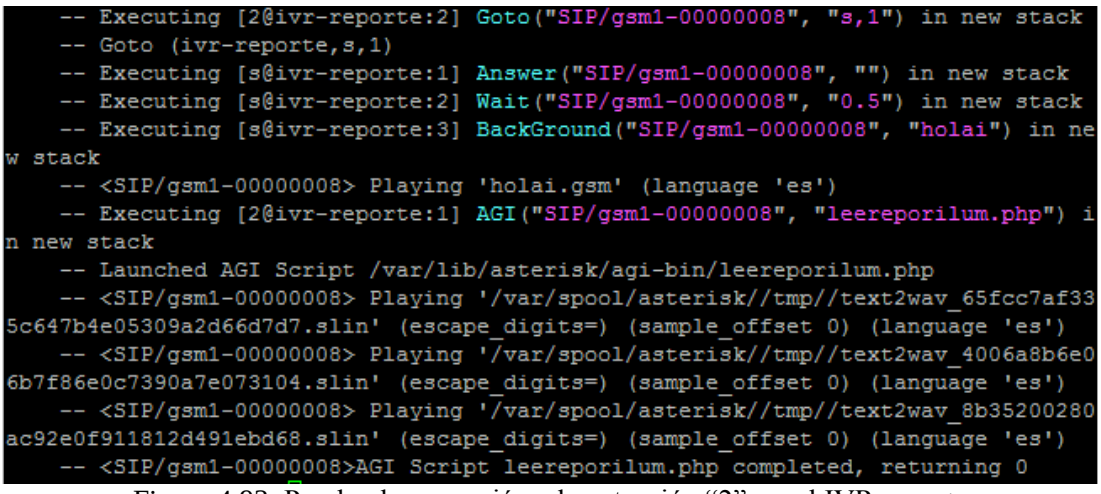


Figura 4.93: Prueba de marcación a la extensión "2", en el IVR-reporte
Fuente Investigador

En el IVR-reporte al marcar la extensión "3", la central telefónica hace el llamado al archivo leerreporph.php que contiene la programación necesaria para que la central telefónica interactúe con la base de datos e informe el último valor almacenado fuera de rango del sensor de pH.

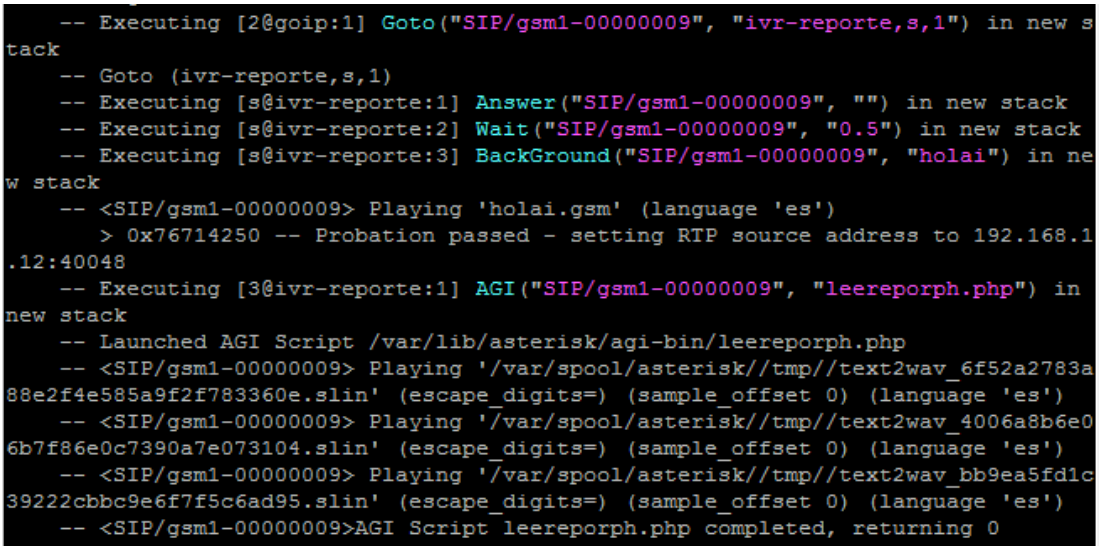


Figura 4.94: Prueba de marcación a la extensión "3", en el IVR-reporte.
Fuente: Investigador

En el IVR-reporte al marcar la extensión "4", la central telefónica hace el llamado al archivo leerreportempagua.php que contiene la programación necesaria para que la central telefónica interactúe con la base de datos e informe el último valor almacenado fuera de rango del sensor de temperatura del agua.

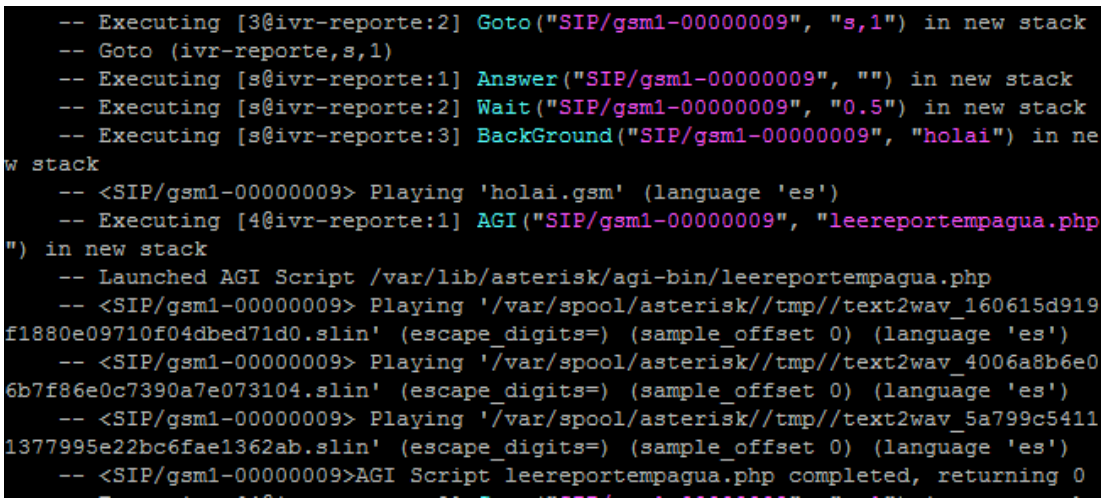


Figura 4.95: Prueba de marcación a la extensión "4", en el IVR-reporte.
Fuente: Investigador

En el IVR-reporte al marcar la extensión "5", la central telefónica hace el llamado al archivo leerreportempamb.php que contiene la programación necesaria para que la central telefónica interactúe con la base de datos e informe el último valor almacenado fuera de rango del sensor de temperatura ambiental.

```
    -- Executing [4@ivr-reporte:2] Goto("SIP/gsm1-00000009", "s,1") in new stack
    -- Goto (ivr-reporte,s,1)
    -- Executing [s@ivr-reporte:1] Answer("SIP/gsm1-00000009", "") in new stack
    -- Executing [s@ivr-reporte:2] Wait("SIP/gsm1-00000009", "0.5") in new stack
    -- Executing [s@ivr-reporte:3] BackGround("SIP/gsm1-00000009", "holai") in ne
w stack
    -- <SIP/gsm1-00000009> Playing 'holai.gsm' (language 'es')
    -- Executing [5@ivr-reporte:1] AGI("SIP/gsm1-00000009", "leereportempamb.php"
) in new stack
    -- Launched AGI Script /var/lib/asterisk/agi-bin/leereportempamb.php
    -- <SIP/gsm1-00000009> Playing '/var/spool/asterisk//tmp//text2wav_f2c0eaaa8d
259720c44bc23cb69687e0.slin' (escape_digits=) (sample_offset 0) (language 'es')
    -- <SIP/gsm1-00000009> Playing '/var/spool/asterisk//tmp//text2wav_4006a8b6e0
6b7f86e0c7390a7e073104.slin' (escape_digits=) (sample_offset 0) (language 'es')
    -- <SIP/gsm1-00000009> Playing '/var/spool/asterisk//tmp//text2wav_ec818f8feb
c9699fe927a439353addbc.slin' (escape_digits=) (sample_offset 0) (language 'es')
    -- <SIP/gsm1-00000009>AGI Script leereportempamb.php completed, returning 0
```

Figura 4.96: Prueba de marcación a la extensión “5”, en el IVR-reporte.
Fuente: Investigador

4.38.4 Prueba de activación de alertas

Pruebas de funcionamiento del ventilador

Cuando la temperatura ambiental es superior a 25°C se enciende el ventilador, como se muestra en la figura 4.97.

Figura 4.97: Prueba de funcionamiento del ventilador.
Fuente: Investigador

Pruebas de funcionamiento de la lámpara fluorescente

Cuando el sensor de luz detecta un nivel de lux menor a 25 y es entre las 7h00 y 18h00, se enciende la lámpara fluorescente, como se muestra en la figura 4.98.

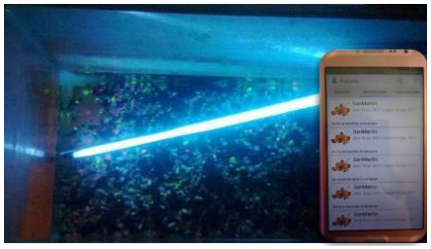


Figura 4.98: Prueba de funcionamiento de la lámpara fluorescente.
Fuente: Investigador

Pruebas de funcionamiento del calentador de agua

Cuando el sensor de temperatura de agua es menor a 28°C, se enciende el calentador de agua, como se muestra en la figura 4.99. Cuando la temperatura del agua alcanza el límite máximo que es 32°C el calentador de agua se apaga.

Figura 4.99: Prueba de funcionamiento del calentador de agua.
Fuente: Investigador

Pruebas de funcionamiento de la luz piloto

Cuando el sensor de pH detecta un nivel de pH menor a 6.5 y mayor a 7.5 se enciende una luz piloto, como se muestra en la figura 4.100.

Figura 4.100: Prueba de funcionamiento de la luz piloto.
Fuente: Investigador

4.38.5 Prueba de envió de alertas a través de Pushetta

Cuando la temperatura ambiental es superior a 25°C se activa el ventilador y por medio de la programación de Python envía una notificación a través de Pushetta informando que el ventilador se ha encendido, como se muestra en la figura 4.101.

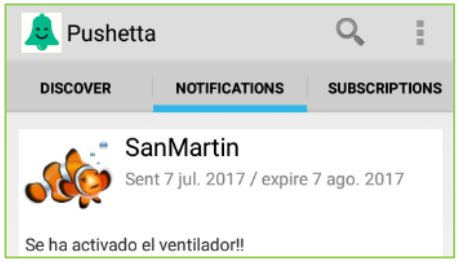


Figura 4.101: Notificación de Pushetta para activación del ventilador
Fuente: Investigador

Cuando el sensor BH1750 detecta un nivel de lux menor a 25 y es entre las 7h00 y 18h00, se enciende la lámpara y por medio de la programación de Python envía una notificación a través de Pushetta informando que se ha encendido la lámpara, como se muestra en la figura 4.102.

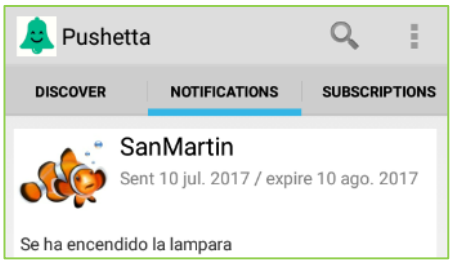


Figura 4.102: Notificación de Pushetta para activación de la lámpara.
Fuente: Investigador

Cuando el sensor de temperatura de agua es inferior a 28°C, se enciende el calentador de agua y cuando la temperatura es superior a 32°C, el calentador de agua se apaga. Por medio de la programación de Python envía una notificación a través de Pushetta, informando que se ha encendido el calentador de agua, como se muestra en la figura 4.103.

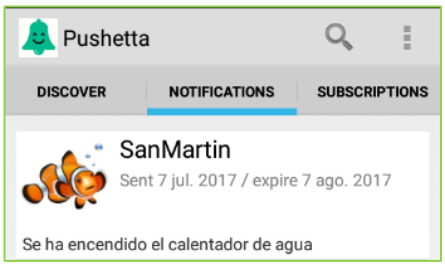


Figura 4.103: Notificación de Pushetta para activación del calentador de agua.
Fuente: Investigador

Cuando el sensor SEN0161 detecta un nivel de pH menor a 6.5 y mayor a 7.5 se enciende una luz piloto y por medio de la programación de Python envía una notificación a través de Pushetta informando que debe cambiar el agua de la pecera, como se muestra en la figura 4.104.

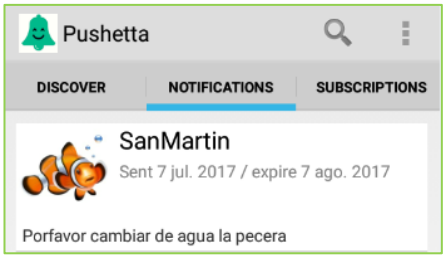


Figura 4.104: Notificación de Pushetta que pide cambiar de agua la pecera.
Fuente Investigador

4.39. Tablas de Resultados

a) Tabla de funcionamiento del Ventilador

En un inicio el ventilador no funcionaba, porque la lectura del último valor almacenado por el sensor de temperatura ambiente, no se estaba almacenando en la variable correspondiente asignada al ventilador, es decir que siempre se estaba obteniendo una temperatura ambiental de 0°C. En el momento que se colocó correctamente la variable de almacenamiento del último valor obtenido por el sensor, ya se obtuvieron valores de temperatura diferentes de cero, y se pudo comprobar el correcto funcionamiento del ventilador.

Tabla 4.9: Pruebas de funcionamiento del Ventilador.

Fecha	Hora	Funcionó bien	No funcionó
Lunes 26 de Junio del 2017	10.00 am		X
Miércoles 28 de Junio del 2017	13.50 pm		X
Jueves 29 de Junio del 2017	11:00 am	X	
Martes 04 de Julio del 2017	9:25 am	X	
Miércoles 05 de Julio del 2017	8:00 am	X	
Viernes 07 de Julio del 2017	10:30 am	X	
Martes 11 de Julio del 2017	14:00 pm	X	
Jueves 13 de Julio del 2017	16:00 pm	X	
Lunes 17 de Julio del 2017	7:40 am	X	
Miércoles 19 de Julio del 2017	9:50 am	X	

Fuente: Investigador

b) Tabla de funcionamiento de la lámpara fluorescente

En un inicio la lámpara no funcionaba correctamente, por la calibración del sensor BH1750, los valores que proporcionaba el sensor eran superiores a 100 lux, es decir que se daban valores de intensidad lumínica muy altos, y por ende no se requería del uso de la lámpara, al calibrar el sensor de luz se obtuvieron valores reales permitiendo encender la lámpara fluorescente. En la séptima prueba no se encendió la lámpara, porque la hora proporcionada por el servidor no coincidía con la hora actual correcta, por ello fue necesario modificar la hora en el servidor. De esta manera se logró corroborar el funcionamiento de la lámpara fluorescente.

Tabla 4.10: Pruebas de funcionamiento de la lámpara fluorescente.

Fecha	Hora	Funcionó bien	No funcionó
Lunes 26 de Junio del 2017	10.00 am		X
Miércoles 28 de Junio del 2017	13.50 pm		X
Jueves 29 de Junio del 2017	11:00 am		X
Martes 04 de Julio del 2017	9:25 am	X	
Miércoles 05 de Julio del 2017	8:00 am	X	
Viernes 07 de Julio del 2017	10:30 am	X	
Martes 11 de Julio del 2017	14:00 pm		X
Jueves 13 de Julio del 2017	16:00 pm	X	
Lunes 17 de Julio del 2017	7:40 am	X	
Miércoles 19 de Julio del 2017	9:50 am	X	

Fuente: Investigador

c) Tabla de funcionamiento del calentador de agua

En un inicio el calentador de agua no funcionaba correctamente cuando los niveles de temperatura eran superiores a 32°C, el agua se seguía calentando, es por ello que se adquirió un nuevo calentador de agua en el que era posible calibrar la temperatura máxima, es decir que cuando llegue a los 32°C se apague, y cuando los niveles de temperatura sean inferiores a 28°C se encienda nuevamente el calentador de agua. De esta manera se logró controlar que la temperatura se encuentre en un rango de 28°C a 32°C que se requiere para el pez Óscar.

Tabla 4.11: Pruebas de funcionamiento del calentador de agua.

Fecha	Hora	Funcionó bien	No funcionó
Lunes 26 de Junio del 2017	10.00 am		X
Miércoles 28 de Junio del 2017	13.50 pm		X
Jueves 29 de Junio del 2017	11:00 am		X
Martes 04 de Julio del 2017	9:25 am		X
Miércoles 05 de Julio del 2017	8:00 am	X	
Viernes 07 de Julio del 2017	10:30 am	X	
Martes 11 de Julio del 2017	14:00 pm	X	

Jueves 13 de Julio del 2017	16:00 pm	X	
Lunes 17 de Julio del 2017	7:40 am	X	
Miércoles 19 de Julio del 2017	9:50 am	X	

Fuente: Investigador

d) Tabla de funcionamiento de la luz piloto

Los datos adquiridos por el sensor de pH no tenían variación, proporcionaban un nivel de pH de 10.39, esto ocurrió porque no se utilizó correctamente el sensor, al introducir el sensor en cualquier tipo de agua siempre se obtenía el mismo valor de pH, porque no se había retiró el recubrimiento del sensor SEN0161 y por ende no se estaba midiendo ningún valor en el agua, excepto el nivel de líquido que viene por defecto en el sensor. Una vez retirado el recubrimiento se logró obtener variaciones en los valores de pH, con las siguientes pruebas únicamente se obtenían valores de pH ácidos es decir únicamente valores inferiores a 7, para corregir este error se realizó calibraciones en el sensor de pH y se logró obtener valores de pH neutro, alcalino y ácido. De esta manera se logró corroborar el funcionamiento de la activación de la Luz piloto cuando el agua se encontraba con impurezas.

Tabla 4.12: Pruebas de funcionamiento de la luz piloto.

Fecha	**Hora**	**Funcionó bien**	**No funcionó**
Lunes 26 de Junio del 2017	10.00 am		X
Miércoles 28 de Junio del 2017	13.50 pm		X
Jueves 29 de Junio del 2017	11:00 am	X	
Martes 04 de Julio del 2017	9:25 am	X	
Miércoles 05 de Julio del 2017	8:00 am		X
Viernes 07 de Julio del 2017	10:30 am		X
Martes 11 de Julio del 2017	14:00 pm	X	
Jueves 13 de Julio del 2017	16:00 pm	X	
Lunes 17 de Julio del 2017	7:40 am	X	
Miércoles 19 de Julio del 2017	9:50 am	X	

Fuente: Investigador

4.40 Recursos Económicos

Loa materiales fueron adquiridos en las diferentes electrónicas del país, excepto el GoIP que fue importado, a continuación se detalla todos los elementos utilizados en el desarrollo del presente proyecto con su respectivo costo incluido el IVA.

Tabla 4.13: Presupuesto.

Ítem	Detalle	Unidad	Cantidad	Costo Unitario	Costo Total
1	Raspberry PI3	c/u	1	$65,00	$65,00
2	Arduino Uno	c/u	1	$12,28	$12,28
3	Shield Ethernet	c/u	1	$12,72	$12,72
4	Cargador Raspberry Pi3	c/u	1	$6,00	$6,00
5	Micro SD	c/u	1	$23,00	$23,00
6	Sensor de temperatura de agua	c/u	1	$10,00	$10,00
7	Sensor temperatura ambiental	c/u	1	$8,77	$8,77
8	Sensor de pH	c/u	1	$ 60,53	$ 60,53
9	Sensor de Luz	c/u	1	$14,73	$14,73
10	Módulo Relé	c/u	1	$ 7,50	$ 7,50
11	Juego de cables macho	c/u	1	$2,50	$2,50
12	GoIP	c/u	1	$146,29	$146,29
13	Baquelita	c/u	1	$0,65	$0,65
14	Ácido férrico	c/u	1	$0,75	$0,75
15	Ventilador 5V	c/u	1	$2,10	$2,10
16	Ventilador 110V	c/u	1	$14,60	$14,60
17	Calentador de Agua	c/u	1	$12,00	$12,00
18	Lámpara Fluorescente	c/u	1	$8,50	$8,50
19	Luz piloto	c/u	1	$2,50	$2,50
20	Espadín hembra	c/u	2	$0,50	$1,00
21	Espadín macho	c/u	2	$0,50	$1,00
22	Bornera de 2	c/u	2	$0,25	$0,50
23	Cable de impresora	c/u	1	$6,00	$6,00
24	Switch de 8 puertos TP-Link	c/u	1	$15,00	$15,00

Ítem	Detalle	Unidad	Cantidad	Costo Unitario	Costo Total
25	Cable HDMI	c/u	1	$7,00	$7,00
26	Papel Couche	c/u	1	$1,00	$1,00
27	Impresiones	c/u	1	$15,90	$15,90
28	Silicona	c/u	1	$0,80	$0,80
29	Carcasa para circuito	c/u	1	$6,00	$6,00
30	Cables Ethernet	c/u	5	$0,60	$3,00
31	Cortapicos	c/u	1	$6,24	$6,24
	Total				$473,86

Fuente: Investigador

Para el desarrollo del presente proyecto, el presupuesto fue financiado en su totalidad por el investigador.

Para el cálculo del valor de horas de trabajo de un Ingeniero Electrónico, se tomó en consideración el salario obtenido de la página web del Instituto Ecuatoriano de Seguridad Social.

El sueldo mensual del Ingeniero Electrónico es de $1.676,00, el sueldo diario sería de $1.676,00/20 días= 83,80. Por lo tanto el sueldo diario sería de 10,475.

Tabla 4.14: Salario obtenido del Instituto Ecuatoriano de Seguridad Social.

Descripción	Valor
Sueldo Mensual	$1.676
Sueldo Diario	$83,84
Sueldo por hora	$10,48

Fuente: Investigador.

Para el desarrollo del presente proyecto se empleó 190 horas, por lo que el diseño estaría valorado en $1.991,20. Por lo tanto el costo total del proyecto es de $2.465,06.

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

Del desarrollo e implementación del presente proyecto, se puede llegar a las siguientes conclusiones:

- Se realizó el análisis de las variables físicas que intervienen en el Acuario "San Martín", llegando a la conclusión de que se requiere como mínimo, una intensidad lumínica de 25 lux, para que el acuario se encuentre completamente iluminado; la temperatura ambiental en el día varía entre 22°C y 28°C, donde fue necesario controlar que la temperatura no sea superior a 25°C, para brindar un ambiente agradable a los turistas; cuando el agua de la pecera se encontraba sucia, se observaba que se contaba con niveles alcalinos de pH entre 7.8 y 8; la temperatura del agua, requiere como mínimo una temperatura de 28°C y como máximo 32°C, para que el pez Óscar se desarrolle con normalidad.
- Al ser un acuario de grandes dimensiones, se llega a la conclusión de que se requiere el uso de dos calentadores de agua, para nivelar la temperatura en el acuario, colocando en la parte derecha e izquierda de la pecera, porque se obtuvo niveles de temperatura en el lado derecho de 29°C a 31°C y en el lado izquierdo de 27°C a 30°C, también es necesario el uso de dos lámparas fluorescentes de 20W con una eficiencia de 68 lm/watt, colocadas en la parte inferior frontal y en la parte inferior trasera del acuario, para que el acuario se encuentre iluminado en su totalidad.
- Con la implementación de la central telefónica en Asterisk se llega a la conclusión de que es necesario analizar y controlar todos los parámetros que se den al realizar una llamada telefónica, al realizar la llamada y no marcar ninguna extensión, la llamada sigue en curso hasta que el usuario cuelgue, por

ello se requiere limitar el tiempo de espera, colocando un rango de 0 a 5 segundos para marcar alguna extensión, caso contrario pasado los 5 segundos se cuelga la llamada, al marcar una extensión diferente del 1-5 se requiere un mensaje que notifique al usuario que dicha extensión no está disponible.

5.2 Recomendaciones

- Utilizar los sensores apropiados y calibrarlos correctamente, para que la información proporcionada por cada uno de ellos sea la correcta y de esta manera asegurarse que la información almacenada en la base de datos, es verídica, es recomendable cambiar el agua de la pecera, la primera vez que el sistema lo notifique, porque de esta manera se evita que el agua se contamine más rápido y se incremente la aparición de hongos y bacterias.
- Utilizar calentadores de agua, en los que sea posible limitar la temperatura máxima de 32°C, para que los peces no sufran cambios de temperatura que alteran su ciclo biológico natural, antes de encender los calentadores es recomendable introducirlos en el agua 5 minutos, para que nivelen su temperatura y al encenderlos no vayan a explotar, se recomienda utilizar lámparas fluorescentes, que son las indicadas para los acuarios, porque la radiación que emiten no es perjudicial, ni para los peces ni plantas del acuario, por el contrario permite que las plantas acuáticas realicen el proceso de fotosíntesis, así como también visualizar claramente las especies de peces.
- Es recomendable instalar la última versión de Asterisk disponible, porque viene con paquetes actualizados, que pueden ser aprovechados al máximo, logrando tener llamadas telefónicas exitosas, es recomendable colocar tiempos de espera para la marcación de extensiones y para escuchar nuevamente el menú, porque se puede dar el caso que no se escuchó bien el tipo de sensor que se requiere conocer el estado, también se recomienda instalar en Asterisk los paquetes de festival y la voz femenina de la Junta de Andalucía, porque es la voz más clara en español que se encuentra disponible.

Bibliografía

[1] W. A. Z. Aquariums, «Comunidad mundial de zoológicos y acuarios (WAZA),» [En línea]. Available: http://www.waza.org. [Último acceso: 14 Marzo 2017].

[2] Asociación Latinoamericana de Parques Zoológicos y Acuarios, «Estrategia Global de Acuarios,» [En línea]. Available: http://www.alpza.com/. [Último acceso: 14 Marzo 2017].

[3] C. Álvarez González y F. Moyano López, «Nutrición y Alimentación de peces marinos,» 2004. [En línea]. [Último acceso: 15 Marzo 2017].

[4] Soto Muñoz , «Acuario Marino, formación y mantenimiento,» [En línea]. Available: https://goo.gl/opCnRz .

[5] E. M. Glover, «La Era del Acuario,» [En línea]. Available: http://www.christianrosenkreuz.org/laeradeacuario.pdf. [Último acceso: 15 Marzo 2017].

[6] V. Lapuente Solórzano, «Control y monitorización de un acuario en tiempo real mediante tecnología Open Source,» Marzo 2015. [En línea]. Available: https://zaguan.unizar.es/record/37185/files/TAZ-PFC-2015-154.pdf. [Último acceso: 14 Marzo 2017].

[7] V. Valencia, «Diseño de un Sistema de control de acuarios.,» [En línea]. Available: http://bibdigital.epn.edu.ec/bitstream/15000/7144/1/CD-5339.pdf. [Último acceso: 16 Marzo 2017].

[8] N. De Juan Váazques, «SCADA para control de acuarios mediante arduino y VB.,» [En línea]. Available: https://goo.gl/ubbKmI. [Último acceso: 20 Mazo 2017].

[9] R. Almeida, «Diseño e implementación de un Sistema automatizado de control de cambio de agua y mantenimiento de acuarios medianos y pequeños.,» Febrero 2014. [En línea]. Available: http://bibdigital.epn.edu.ec/bitstream/15000/7144/1/CD-5339.pdf. [Último acceso: 22 Marzo 2017].

[10] P. Bos, «Control de acuario con CIAA,» Julio 2016. [En línea]. Available: https://goo.gl/lL46ne. [Último acceso: 27 Marzo 2017].

[11] A. Castro Snurmacher, «Controlador de acuarios por ordenador basado en arduino.,» Febrero 2014. [En línea]. Available: http://ciberdroide.com/AcuBioMed/wp-content/uploads/2014/09/PrimerasPaginasCAO1_Kindle.pdf. [Último acceso: 01 Abril 2017].

[12] C. Cusi y F. Sánchez, «Estudio y diseño de un prototipo para el monitoreo de acuarios utilizando tecnología WIFI (IEEE 802.11b/g/n) enfocado al IoT (Internet of things), mediante una plataforma raspberry pi y el sistema operativo Android.,» Noviembre

2016. [En línea]. Available: http://bibdigital.epn.edu.ec/bitstream/15000/16904/1/CD-7483.pdf. [Último acceso: 03 Abril 2017].

[13] F. P. J. C, «Acuariofilia, enfermedades y tratamientos de peces de acuario,» [En línea]. Available: https://botplusweb.portalfarma.com/Documentos/2017/5/19/115332.pdf. [Último acceso: 12 Mayo 2017].

[14] F. Lango Reynoso y M. Castañeda Chávez, «Laboratorio de Investigación de Recursos Acuáticos,» 2012. [En línea]. Available: http://www.scielo.cl/scielo.php?pid=S0718-560X2012000100002&script=sci_arttext. [Último acceso: 04 Abril 2017].

[15] Teton Jackes, «Guía Técnica de la Acuariofilia,» [En línea]. Available: https://goo.gl/bZkEMO . [Último acceso: 04 Abril 2017].

[16] Acuario, «Principales Componentes de un Acuario,» 2009. [En línea]. Available: https://goo.gl/64ReaX .

[17] R. GSM, «Historia de estándar GSM,» [En línea]. Available: https://goo.gl/ciov72. [Último acceso: 06 Abril 2017].

[18] D. Telecom, «User Manual,» [En línea]. Available: http://www.discoverytelecom.eu/upload/iblock/e87/voip2gsm.pdf. [Último acceso: 17 05 2017].

[19] L. M. J. V. M. Russell Bryant, «Asterisk "The Definitive Guide",» [En línea]. Available: http://asterisk-service.com/downloads/Asterisk-%20The%20Definitive%20Guide,%204th%20Edition.pdf. [Último acceso: 19 05 2017].

[20] J. A. Carballar, «VoIP, La telefonía de Internet,» [En línea]. Available: https://goo.gl/pKPG9u. [Último acceso: 08 Abril 2017].

[21] V. García Suárez, «Introducción a la Raspberry Pi,» [En línea]. Available: http://hacklabalmeria.net/recursos/intropi.pdf. [Último acceso: 09 Abril 2017].

[22] M. Francois, Á. Sánchez y A. Maestre, «Raspberry Pi 2,» Enero 2016. [En línea]. Available: https://goo.gl/uHyTBi . [Último acceso: 09 Abril 2017].

[23] R. Enríquez , «Guía de usuario de Arduino,» Noviembre 2009. [En línea]. Available: http://www.uco.es/aulasoftwarelibre/wp-content/uploads/2010/05/Arduino_user_manual_es.pdf. [Último acceso: 11 Abril 2017].

[24] J. M. Ruíz, «Arduino + Ethernet Shield,» Enero 2013. [En línea]. Available: http://unicarlos.com/_ARDUINO/Arduino%20+%20Ethernet%20Shield%20(1).pdf. [Último acceso: 12 Abril 2017].

[25] A. García , «Aprendiendo a utilizar el sensor de temperatura,» Enero 2014. [En línea]. Available: http://panamahitek.com/aprendiendo-utilizar-el-sensor-de-temperatura-ds18b20/. [Último acceso: 12 Abril 2107].

[26] Naylamp, «Tutorial: módulo sensor de luz,» [En línea]. Available: http://www.naylampmechatronics.com/blog/44_Tutorial-m%C3%B3dulo-sensor-de-luz-BH1750.html. [Último acceso: 13 Abril 2017].

[27] I. y. D. Electrónico, «Sensor de temperatura y humedad ambiental,» [En línea]. Available: https://electronilab.co/tienda/sensor-de-temperatura-y-humedad-dht22/. [Último acceso: 13 Abril 2017].

[28] E. Sigma, «Sensor de pH,» [En línea]. Available: http://www.sigmaelectronica.net/sen0161-p-2101.html. [Último acceso: 14 Abril 2017].

[29] Omron, «Módulos Relé de Seguridad,» [En línea]. Available: http://afly.co/tvf. [Último acceso: 14 04 2017].

[30] R. Camps y L. A. Casillas, «Base de Datos,» [En línea]. Available: http://www.uoc.edu/masters/oficiales/img/913.pdf. [Último acceso: 15 Abril 2017].

[31] E. Gómez y P. Martinez, «Base de Datos 1,» [En línea]. Available: https://rua.ua.es/dspace/bitstream/10045/2990/1/ApuntesBD1.pdf. [Último acceso: 16 Abril 2017].

[32] M. Mañas, «phpMyAdmin,» [En línea]. Available: http://personales.upv.es/moimacar/download/servidores/phpmyadmin.pdf. [Último acceso: 20 Abril 2017].

[33] E. Gonzales , «Servidores,» [En línea]. Available: https://goo.gl/2jGuHQ. [Último acceso: 25 Abril 2017].

[34] M. Babir, «Servidor Apache,» [En línea]. Available: https://goo.gl/JtrEnM . [Último acceso: 28 Abril 2017].

[35] A. Electronics, «Sensor DHT22,» [En línea]. Available: https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf. [Último acceso: 09 Mayo 2017].

[36] C. Electronic, «Sensor de humedad y tempertuta ambiental SHT11,» [En línea]. Available: http://www.tme.eu/es/details/sht11/sensores-de-humedad/sensirion/.

[37] N. Mechatronics, «Sensor de Luz BH1750,» [En línea]. Available: http://www.naylampmechatronics.com/sensores-luz-y-sonido/76-modulo-sensor-de-luz-digital-bh1750.html. [Último acceso: 08 Mayo 2017].

[38] T. CRN, «Transmisores para medir Iluminación, Luminancia, Irradiación.,» [En línea]. Available: http://www.crntecnopart.com/images/pdf/ESP/transmiluz.pdf.

[39] OpenHacks, «Sensor de pH SEN0161,» [En línea]. Available: https://www.openhacks.com/uploadsproductos/ph_meter_sku__sen0161__-_robot_wiki.pdf. [Último acceso: 09 Mayo 2017].

[40] I. Hanna, «Electrodo de pH,» [En línea]. Available: http://www.hannainst.es/catalogo-productos/electrodos/electrodos-de-ph/electrodo-de-ph-con-sensor-de-temperatura-para-usos-generales-hi-1217.

[41] D. Semiconductor, «Sensor de temperatura DS18B20,» [En línea]. Available: https://cdn.sparkfun.com/datasheets/Sensors/Temp/DS18B20.pdf. [Último acceso: 06 Mayo 2017].

[42] C. Sutron, «Sensor de Temperatura SDI-12 sumergible,» [En línea]. Available: https://www.sutron.com/wp-content/uploads/2015/01/aquatemp.pdf.

[43] A. Electronics, «Arduino Uno,» [En línea]. Available: http://digital.csic.es/bitstream/10261/127788/7/D-c-%20Arduino%20uno.pdf. [Último acceso: 02 Mayo 2017].

[44] R. B, «Raspberry Pi 3 Model B,» [En línea]. Available: http://docs-europe.electrocomponents.com/webdocs/14ba/0900766b814ba5fd.pdf. [Último acceso: 02 Mayo 2017].

[45] G. Coley, «BeagleBone Rev A6,» [En línea]. Available: http://beagleboard.org/static/Docs/Hardware/BONE_SRM.pdf.

[46] A. Products, «Compare Boards Specs,» [En línea]. Available: https://www.arduino.cc/en/Products/Compare.

[47] ElectronicLab, «Raspberry Pi,» [En línea]. Available: https://electronilab.co/tienda/raspberry-pi-3-modelo-b-armv8-1g-ram/.

[48] C. Italia, «Datasheet GoIP,» [En línea]. Available: http://www.contechitalia.com/download/GoIP1%20Datasheet.pdf. [Último acceso: 05 Mayo 2017].

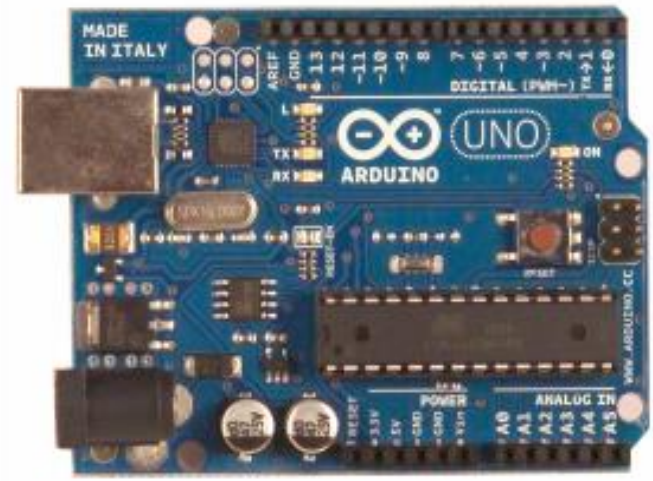
ANEXOS

Anexos

I. Anexo A: Parámetros Técnicos de los equipos empleados en el proyecto.

a) Arduino Uno

Arduino UNO



Product Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduno, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

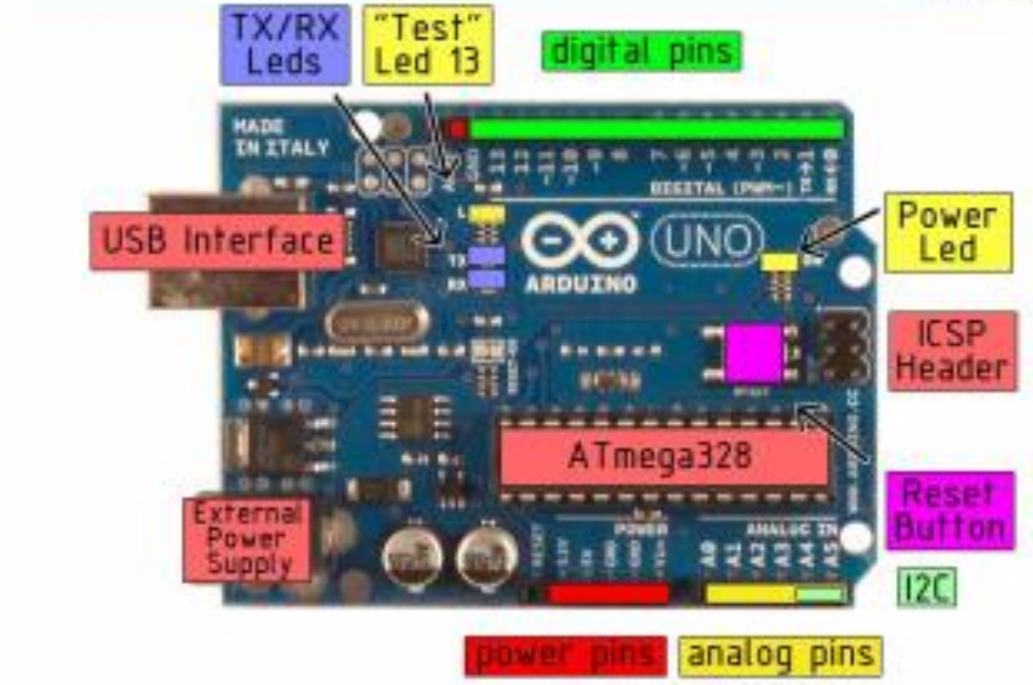
Technical Specification

EAGLE files: arduino-duemilanove-uno-design.zip Schematic: arduino-uno-schematic.pdf

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB of which 0.5 KB used by bootloader
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

the board



b) Arduino Ethernet Shield

Arduino Ethernet Shield

Overview

The Arduino Ethernet is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins, 6 analog inputs, a 16 MHz crystal oscillator, a RJ45 connection, a power jack, an ICSP header, and a reset button.

NB: Pins 10, 11, 12 and 13 are reserved for interfacing with the Ethernet module and should not be used otherwise. This reduces the number of available pins to 9, with 4 available as PWM outputs.

An optional Power over Ethernet module can be added to the board as well.

The Ethernet differs from other boards in that it does not have an onboard USB-to-serial driver chip, but has a Wiznet Ethernet interface. This is the same interface found on the Ethernet shield.

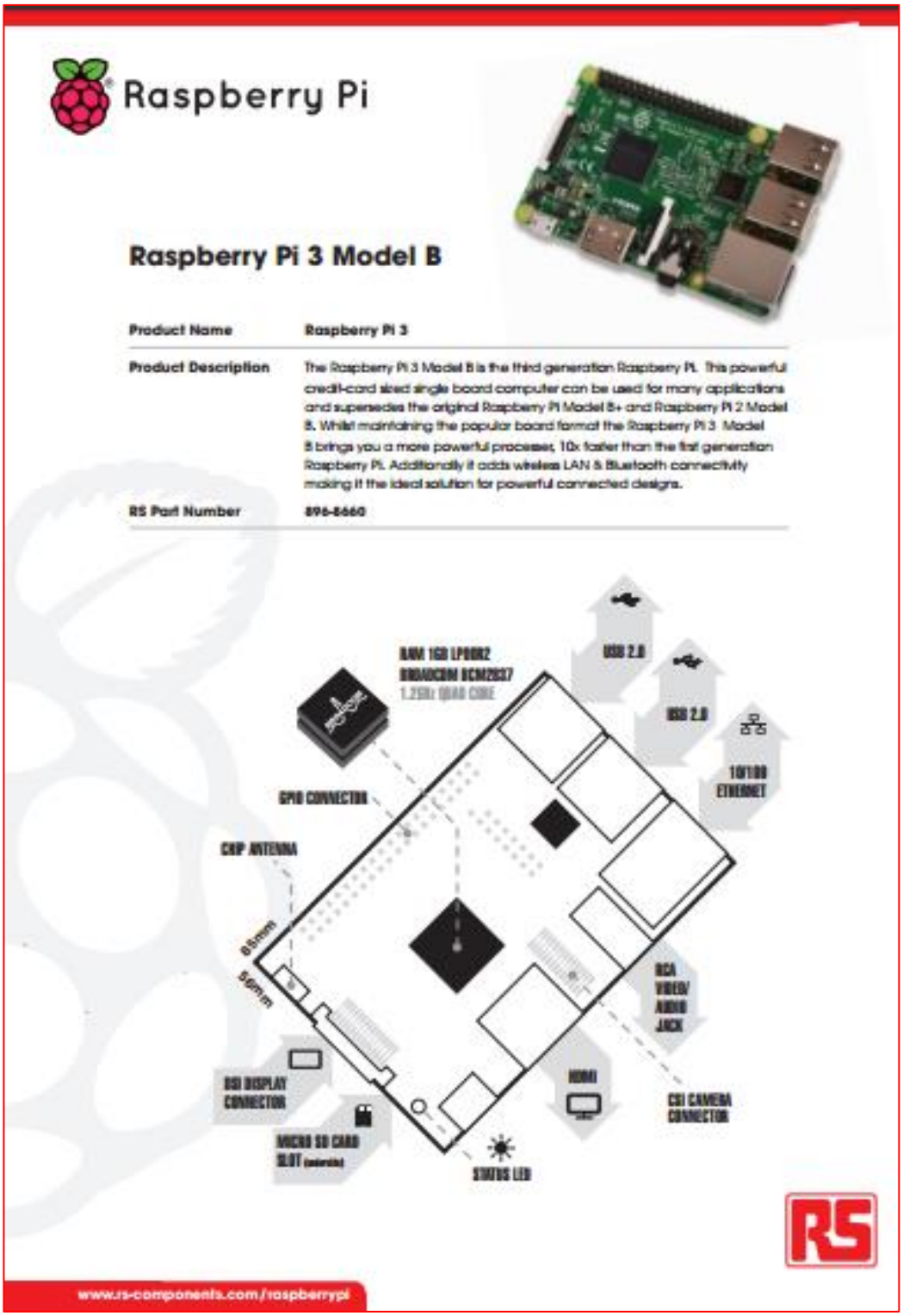
An onboard microSD card reader, which can be used to store files for serving over the network, is accessible through the SD Library. Pin 10 is reserved for the Wiznet interface, SS for the SD card is on Pin 4.

The 6-pin serial programming header is compatible with the USB Serial adapter and also with the FTDI USB cables or with Sparkfun and Adafruit FTDI-style basic USB-to-serial breakout boards. It features support for automatic reset, allowing sketches to be uploaded without pressing the reset button on the board. When plugged into a USB to Serial adapter, the Arduino Ethernet is powered from the adapter.

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage Plug(limits)	6-18V
Input Voltage PoE (limits)	36-57V
Digital I/O Pins	14 (of which 4 provide PWM output)
Arduino Pins reserved:	
	10 to 13 used for SPI
	4 used for SD card
	2 W5100 interrupt (when bridged)
Analog Input Pins	6
DC Current per I/O Pin	40 mA

c) Raspberry Pi



Raspberry Pi

Raspberry Pi 3 Model B

Specifications

Processor	Broadcom BCM2387 chipset. 1.2GHz Quad-Core ARM Cortex-A53 802.11 b/g/n Wireless LAN and Bluetooth 4.1 (Bluetooth Classic and LE)
GPU	Dual Core VideoCore IV® Multimedia Co-Processor. Provides Open GL ES 2.0, hardware-accelerated OpenVG, and 1080p30 H.264 high-profile decode. Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure
Memory	1GB LPDDR2
Operating System	Boots from Micro SD card, running a version of the Linux operating system or Windows 10 IoT
Dimensions	85 x 56 x 17mm
Power	Micro USB socket 5V1, 2.5A

Connectors:

Ethernet	10/100 BaseT Ethernet socket
Video Output	HDMI (rev 1.3 & 1.4 Composite RCA (PAL and NTSC)
Audio Output	Audio Output 3.5mm jack, HDMI USB 4 x USB 2.0 Connector
GPIO Connector	40-pin 2.54 mm (100 mil) expansion header: 2x20 strip Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines
Camera Connector	15-pin MIPI Camera Serial Interface (CSI-2)
Display Connector	Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane
Memory Card Slot	Push/pull Micro SDIO

Key Benefits

- Low cost
- Consistent board format
- 10x faster processing
- Added connectivity

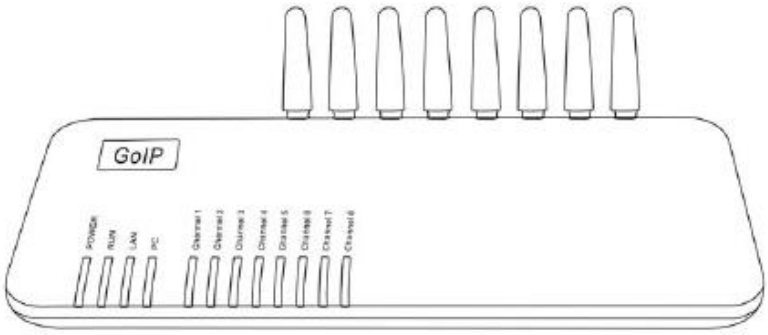
Key Applications

- Low cost PC/tablet/laptop
- IoT applications
- Media centre
- Robotics
- Industrial/Home automation
- Server/cloud server
- Print server
- Security monitoring
- Web camera
- Gaming
- Wireless access point
- Environmental sensing/monitoring (e.g. weather station)

RS

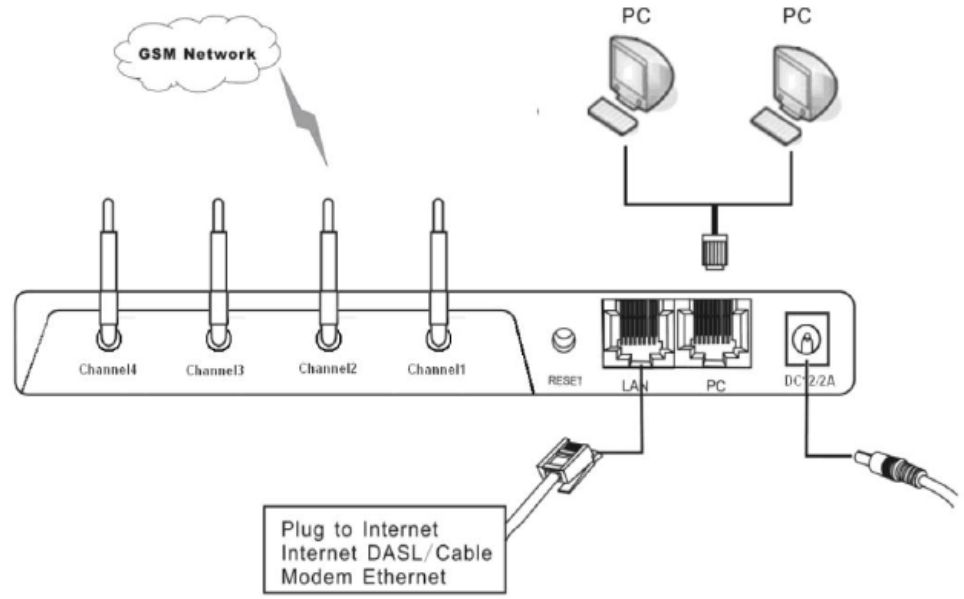
www.rs-components.com/raspberrypi

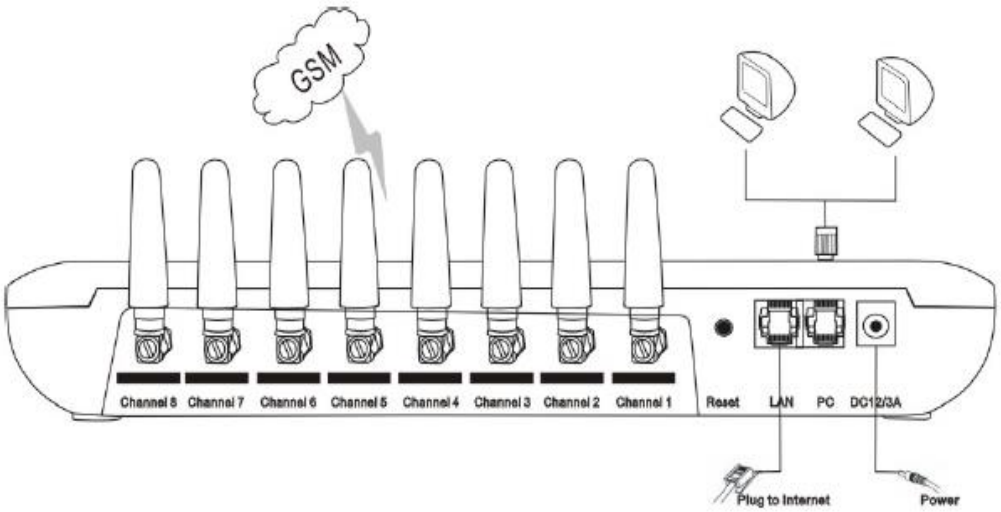
d) GoIP



The description of LED indicators is as follows.

LED	Description
Power indicator	This indicator is constantly ON after connected with power.
LAN indicator	This indicator is ON after connected with the network equipment and blinks during data transmission.
PC indicator	This indicator is ON after connected with the network equipment and blinks during data transmission.
RUN indicator	1. The RUN indicator blinks once every 100ms during startup. 2. When GoIP is connected with the server, the RUN indicator blinks once per second.
Channel indicator	1. When the GSM module of the GoIP logs onto the local GSM network, this indicator blinks once per second. 2. When this indicator blinks quickly, it indicates that the GoIP is trying to log onto the local GSM network. 3. When the GSM channel of the GoIP is activated, this indicator is normally ON.


GSM Network
PC
PC
Channel4
Channel3
Channel2
Channel1
RESET
LAN
PC
DC12/2A
Plug to Internet
Internet DASL/Cable
Modem Ethernet


GSM
Channel 8
Channel 7
Channel 6
Channel 5
Channel 4
Channel 3
Channel 2
Channel 1
Reset
LAN
PC
DC12/3A
Plug to Internet
Power


GoIP

II. Anexo B: Parámetros Técnicos de los sensores empleados en el proyecto.

a) Sensor de temperatura digital DS18B20

DALLAS SEMICONDUCTOR MAXIM

www.maxim-ic.com

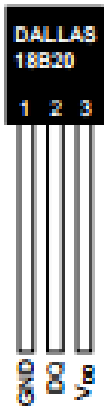
DS18B20

Programmable Resolution 1-Wire Digital Thermometer

FEATURES

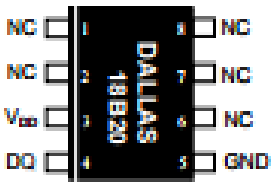
- Unique 1-Wire® interface requires only one port pin for communication
- Each device has a unique 64-bit serial code stored in an onboard ROM
- Multidrop capability simplifies distributed temperature sensing applications
- Requires no external components
- Can be powered from data line. Power supply range is 3.0V to 5.5V
- Measures temperatures from –55°C to +125°C (–67°F to +257°F)
- ±0.5°C accuracy from –10°C to +85°C
- Thermometer resolution is user-selectable from 9 to 12 bits
- Converts temperature to 12-bit digital word in 750ms (max.)
- User-definable nonvolatile (NV) alarm settings
- Alarm search command identifies and addresses devices whose temperature is outside of programmed limits (temperature alarm condition)
- Available in 8-pin SO (150mil), 8-pin µSOP, and 3-pin TO-92 packages
- Software compatible with the DS1822
- Applications include thermostatic controls, industrial systems, consumer products, thermometers, or any thermally sensitive system

PIN ASSIGNMENT

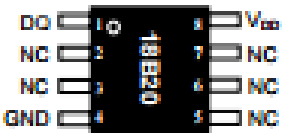


(BOTTOM VIEW)

TO-92
(DS18B20)



8-Pin 150mil SO
(DS18B20Z)



8-Pin µSOP
(DS18B20U)

PIN DESCRIPTION

GND - Ground
DQ - Data In/Out
V_{DD} - Power Supply Voltage
NC - No Connect

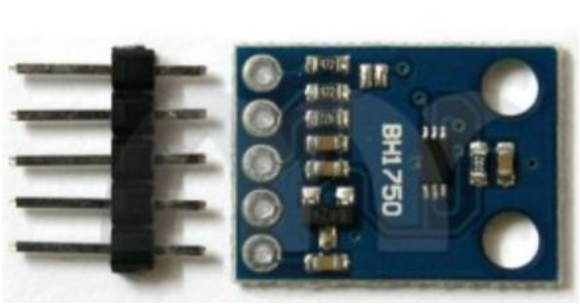
DESCRIPTION

The DS18B20 Digital Thermometer provides 9 to 12–bit centigrade temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. It has an operating temperature range of –55°C to +125°C and is accurate to ±0.5°C over the range of –10°C to +85°C. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1–wire bus; thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment or machinery, and process monitoring and control systems.

b) Sensor de luz BH1750

Sensor Digital De Luz BH1750



Descripción

El BH1750 es un sensor digital de intensidad de luz ambiente, tiene un conversor ADC de 16bits interno. Esta es una versión mejorada del típico sensor de luz a base de un LDR, el cual simplemente entrega un valor analógico. El BH1750, nos entrega automáticamente el valor En Lux (desde 1 lx hasta 65535 lx), y se comunica por I2C, con la posibilidad de seleccionar 2 Address. Este sensor nos permite medir la cantidad de luz por metro cuadrado que tenemos en alguna habitación y poder modificar automáticamente las luces de los alrededores.

Ejemplos de Lux:

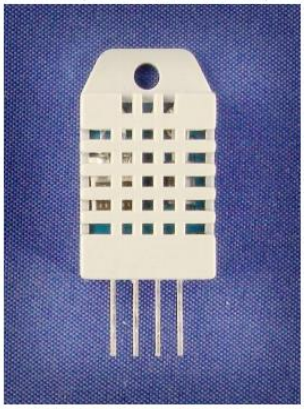
- Noche: 0.001—0.02
- Luz Lunar: 0.02—0.3
- Nublado Interior: 5—50
- Nublado Exterior: 50—500
- Soleado Interior: 100-1000
- Luz minima para la lectura: 50—60
- Intensidad estándar sistema de video hogareño: 1400

Caracteristicas Técnicas

- Interfaz digital a través de bus I2C con capacidad de seleccionar entre 2 direcciones
- Respuesta espectral similar a la del ojo humano
- Reliza mediciones de luz y convierte el resultado a una palabra digital
- Amplio rango de medición 1-65535 lux
- Modo de bajo consumo de energía
- Rechazo de ruido a 50/60 Hz
- Baja dependencia de la medición contra la fuente de luz: halógeno, led, incandescente, luz de día, etc.
- Es posible seleccionar dos direcciones de esclavo (I2C).
- La influencia del espectro infrarojo es baja.
- Voltaje 3.3v-5v

c) Sensor de humedad y temperatura ambiental DHT22

DHT22 (DHT22 also named as AM2302)



Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

1. Feature & Application:

* Full range temperature compensated * Relative humidity and temperature measurement
* Calibrated digital signal *Outstanding long-term stability *Extra components not needed
* Long transmission distance * Low power consumption *4 pins packaged and fully interchangeable

2. Description:

DHT22 output calibrated digital signal. It utilizes exclusive digital-signal-collecting-technique and humidity sensing technology, assuring its reliability and stability.Its sensing elements is connected with 8-bit single-chip computer.

Every sensor of this model is temperature compensated and calibrated in accurate calibration chamber and the calibration-coefficient is saved in type of programme in OTP memory, when the sensor is detecting, it will cite coefficient from memory.

Small size & low consumption & long transmission distance(20m) enable DHT22 to be suited in all kinds of harsh application occasions.

Single-row packaged with four pins, making the connection very convenient.

3. Technical Specification:

Model	DHT22
Power supply	3.3-6V DC
Output signal	digital signal via single-bus
Sensing element	Polymer capacitor
Operating range	humidity 0-100%RH; temperature -40~80Celsius
Accuracy	humidity +-2%RH(Max +-5%RH); temperature <+-0.5Celsius
Resolution or sensitivity	humidity 0.1%RH; temperature 0.1Celsius
Repeatability	humidity +-1%RH; temperature +-0.2Celsius
Humidity hysteresis	+-0.3%RH
Long-term Stability	+-0.5%RH/year
Sensing period	Average: 2s
Interchangeability	fully interchangeable
Dimensions	small size 14*18*5.5mm; big size 22*28*5mm

d) Sensor de pH SEN0161

PH meter(SKU: SEN0161)

From Robot Wiki

Contents

Analog pH Meter Kit

Introduction

Need to measure water quality and other parameters but haven't got any low cost pH meter? Find it difficult to use with Arduino?
Here comes an analog pH meter, specially designed for Arduino controllers and has built-in simple, convenient and practical connection and features. It has an LED which works as the Power Indicator, a BNC connector and PH2.0 sensor interface. To use it, just connect the pH sensor with BNC connector, and plug the PH2.0 interface into the analog input port of any Arduino controller. If pre-programmed, you will get the pH value easily. Comes in compact plastic box with foams for better mobile storage.
Attention:In order to ensure the accuracy of the pH probe, you need to use the standard solution to calibrate it regularly.Generally, the period is about half a year. If you meaure the dirty aqueous solution, you need to increase the frequency of calibration.

Applications

- Water quality testing
- Aquaculture

Specification

- Module Power : 5.00V
- Module Size : 43mm×32mm
- Measuring Range:0-14PH
- Measuring Temperature :0-60 ℃

III. Anexo C: Códigos de programación en Arduino y Raspberry Pi.

A) Código de programación en arduino

I. Programación de los sensores para la toma de información

```
//Se cargan las librerías necesarias para la Ethernet Shield
#include <Ethernet.h>
#include <SPI.h>
//Configuración de la Ethernet Shield
byte MAC[] = {0x00, 0xAA, 0xBB, 0xCC, 0xDE, 0x12}; //Dirección MAC
byte IP_CLIENTE[] = {10,10,0,3}; //Dirección IP del Arduino
byte IP_SERVIDOR[] = {10,10,0,2}; //Dirección IP del servidor
EthernetClient client;


#include <Wire.h>
#include <BH1750.h>
#include "DHT.h"

#define DHTPIN 2                  //Pin digital 2
#define DHTTYPE DHT22
#define pinpH A3                  //Se declara el pin donde se conecta el dato del sensor
#define pin18b20 8                //Se declara el pin donde se conecta el dato del sensor

#include <OneWire.h>              //Se cargan las librerías necesarias para el sensor
#include <DallasTemperature.h>


DHT dht(DHTPIN, DHTTYPE);
BH1750 lightMeter;
OneWire conex(pin18b20);          //Se define e inicializa las variables
DallasTemperature s18b20(&conex);     //Se define e inicializa las variables

unsigned int spH;                 //Se define e inicializa las variables
int k,guardar[10];
float pH;

void setup(){

  Serial.begin(9600);
  lightMeter.begin();             //Inicio sensor de luz
  dht.begin();                    //Inicio sensor de humedad
  s18b20.begin();                 //Inicio sensor de temperatura
  Serial.println("Test de sensores");
  Ethernet.begin(MAC, IP_CLIENTE);
  delay(2000);
}


void loop() {

  //Sensor Temperatura Agua
```

```
float Tagua= s18b20.getTempCByIndex(0);
s18b20.requestTemperatures();
Serial.print("Temperatura_Agua: ");
Serial.print(Tagua);
Serial.println(" *C");
Serial.println("");

 //Sensor pH
    for(k=0;k<10;k++)
{
guardar[k]=analogRead(pinpH);
delay (10);
}
spH=0;
for(k=0;k<10;k++) spH+=guardar[k];
pH=(float)spH*5/1024/10;//Convierte el valor analógico en mv
pH=3.5*pH;
//Convierte el valor de mv a unidades de pH
// Imprimir variables
Serial.print("pH: ");
Serial.print(pH);
Serial.println(" ");


//Sensor Luz
    uint16_t lux = lightMeter.readLightLevel();
  Serial.print("Light: ");
  Serial.print(lux);
  Serial.println(" lx");


  //Sensor Temperatura y humedad Ambiental

  float h = dht.readHumidity();
  float t = dht.readTemperature();

  Serial.print("Humedad Ambiental: ");
  Serial.print(h);
  Serial.println(" %\t");
  Serial.print("Temperatura Ambiental: ");
  Serial.print(t);
  Serial.println(" *C ");
   delay(2000);


// Proceso de envió de muestras al servidor

//HUMEDAD AMBIENTAL

if (client.connect(IP_SERVIDOR, 80)>0)
{
client.print("GET /acuario/humedad_ambiental.php?RH=");
client.print(h);
client.println();
```

```
Serial.println("Dato enviado Humedad_Ambiental");
}
else
{
Serial.println("Sin conexion");
}
client.stop();
client.flush();
delay(2000);



//TEMPERATURA AMBIENTAL
if (client.connect(IP_SERVIDOR, 80)>0)
{
client.print("GET /acuario/temperatura_ambiental.php?gradoC=");
client.print(t);
client.println();
Serial.println("Dato enviado Temperatura_Ambiental");
}
else
{
Serial.println("Sin conexion");
}
client.stop();
client.flush();
delay(2000);



//ILUMINACION
if (client.connect(IP_SERVIDOR, 80)>0)
{
client.print("GET /acuario/iluminacionn.php?lx=");
client.print(lux);
client.println();
Serial.println("Dato enviado Iluminacion");
}
else
{
Serial.println("Sin conexion");
}
client.stop();
client.flush();
delay(2000);



// TEMPERATURA AGUA
if (client.connect(IP_SERVIDOR, 80)>0)
{
client.print("GET /acuario/temperatura_aguaa.php?gradC=");
client.print(Tagua);
client.println();
```

```
Serial.println("Dato enviado Tagua");
}
else
{
Serial.println("Sin conexion");
}
client.stop();
client.flush();
delay(2000);



//PH
if (client.connect(IP_SERVIDOR, 80)>0)
{
client.print("GET /acuario/nivelph.php?pH=");
client.print(pH);
client.println();
Serial.println("Dato enviado pH");
}
else
{
Serial.println("Sin conexion");
}
client.stop();
client.flush();
delay(2000);

}
```

B) Código de programación en Raspberry Pi

Scripts para el envío de información a la base de datos

Se crean los scripts que permiten enviar la información recibida desde los diferentes sensores utilizados en el presente proyecto hacia la base de datos.

Sensor de Temperatura de Agua

Se crea el script temperatura_aguaa.php

Se crea el script que permiten enviar la información recibida desde el sensor de temperatura de agua a la base de datos.

```
<?php
// Se hace el llamado al script conexion.php
require("conexion.php");

// Se lee el dato de temperatura_agua  mediante GET
$datoagua = mysqli_real_escape_string($conex, $_GET['gradC']);

// Se guarda el valor para insertar en la base de datos
```

```
$insertartagua = "INSERT INTO temperatura_agua (gradC)
VALUES('".$datoagua."')";

// Se inserta el valor en la base de datos
mysqli_query($conex, $insertartagua);
mysqli_close($conex);
?>
```

Sensor de Temperatura Ambiental

Se crea el script temperatura_ambiental.php

Se crea el script que permiten enviar la información recibida desde el sensor de temperatura ambiental a la base de datos.

```
<?php
// Se hace el llamado al script conexion.php
require("conexion.php");

// Se lee el dato de temperatura_agua  mediante GET
$datotamb = mysqli_real_escape_string($conex, $_GET['gradoC']);

// Se guarda el valor para insertar en la base de datos
$insertartamb = "INSERT INTO temperatura_ambiente (gradoC)
VALUES('".$datotamb."')";

// Se inserta el valor en la base de datos
mysqli_query($conex, $insertartamb);
mysqli_close($conex);
?>
```

Sensor de Humedad Ambiental

Se crea el script humedad_ambiental.php

Se crea el script que permiten enviar la información recibida desde el sensor de humedad ambiental a la base de datos.

```
<?php
// Se hace el llamado al script conexion.php
require("conexion.php");

// Se lee el dato de temperatura_agua  mediante GET
$datohuma = mysqli_real_escape_string($conex, $_GET['RH']);

// Se guarda el valor para insertar en la base de datos
$insertarhuma = "INSERT INTO humedad_ambiente (RH)
VALUES('".$datohuma."')";

// Se inserta el valor en la base de datos
mysqli_query($conex, $insertarhuma);
mysqli_close($conex);
?>
```

Sensor de Luz

Se crea el script iluminacionn.php

Se crea el script que permiten enviar la información recibida desde el sensor de luz de agua a la base de datos.

```
<?php
// Se hace el llamado al script conexion.php
require("conexion.php");

// Se lee el dato de temperatura_agua  mediante GET
$datoilum = mysqli_real_escape_string($conex, $_GET['lx']);

// Se guarda el valor para insertar en la base de datos
$insertarilum= "INSERT INTO iluminacion (lx)
VALUES('".$datoilum."')";

// Se inserta el valor en la base de datos
mysqli_query($conex, $insertarilum);
mysqli_close($conex);
?>
```

Sensor de pH

Se crea el script nivelph.php

Se crean el script que permiten enviar la información recibida desde el sensor de pH de agua a la base de datos.

```
<?php
// Se hace el llamado al script conexion.php
require("conexion.php");

// Se lee el dato de temperatura_agua  mediante GET
$datoph = mysqli_real_escape_string($conex, $_GET['pH']);

// Se guarda el valor para insertar en la base de datos
$insertarph= "INSERT INTO nivel_pH (pH)
VALUES('".$datoph."')";

// Se inserta el valor en la base de datos
mysqli_query($conex, $insertarph);
mysqli_close($conex);
?>
```

C) Programación para la lectura del último valor sensado que se encuentra almacenado en la base de datos.

Sensor de Temperatura de Agua

```
cd /var/lib/asterisk/agi-bin
sudo nano temperatura_agua.php
sudo chmod +x /var/lib/asterisk/agi-bin/ temperatura_agua.php
```

```
#!/usr/bin/php -q
<?php
require('phpagi.php');
error_reporting(E_ALL);
$agi = new AGI();
$agi-> answer();

$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

$query="SELECT * FROM temperatura_agua order by fecha_tagua desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("sensor de temperatura de agua");
$query="SELECT * FROM temperatura_agua order by fecha_tagua desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("la temperatura del agua es $_re[gradC] grados... la fecha es $_re[fecha_tagua]");
$agi-> hangup();
?>
```

Sensor de Temperatura Ambiental

```
cd /var/lib/asterisk/agi-bin
sudo nano temperatura_ambiente.php
sudo chmod +x /var/lib/asterisk/agi-bin/ temperatura_ambiente.php

#!/usr/bin/php -q
<?php
require('phpagi.php');
error_reporting(E_ALL);
$agi = new AGI();
$agi-> answer();

$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

$query="SELECT * FROM temperatura_ambiente order by fecha_tamb desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("sensor de temperatura ambiental");
$query="SELECT * FROM temperatura_ambiente order by fecha_tamb desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("la temperatura ambiental es $_re[gradoC]... la fecha es $_re[fecha_tamb]");
$agi-> hangup();
?>
```

Sensor de Humedad Ambiental

```
cd /var/lib/asterisk/agi-bin
sudo nano humedad_ambiente.php
sudo chmod +x /var/lib/asterisk/agi-bin/ humedad_ambiente.php

#!/usr/bin/php -q
<?php
require('phpagi.php');
error_reporting(E_ALL);
$agi = new AGI();
$agi-> answer();

$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

$query="SELECT * FROM humedad_ambiente order by fecha_hum desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("sensor de humedad ambiental");
$query="SELECT * FROM humedad_ambiente order by fecha_hum desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("la humedad ambiental es $_re[RH]... la fecha es $_re[fecha_hum]");
$agi-> hangup();
?>
```

Sensor de Iluminación

```
cd /var/lib/asterisk/agi-bin
sudo nano iluminacion.php
sudo chmod +x /var/lib/asterisk/agi-bin/ iluminacion.php


#!/usr/bin/php -q
<?php
require('phpagi.php');
error_reporting(E_ALL);
$agi = new AGI();
$agi-> answer();

$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

$query="SELECT * FROM iluminacion order by fecha_lx desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("sensor de iluminacion");
$query="SELECT * FROM iluminacion order by fecha_lx desc";
```

```
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("la iluminacion es $_re[lx]lux... la fecha es $_re[fecha_lx]");
$agi-> hangup();
?>
```

Sensor de nivel de pH

```
cd /var/lib/asterisk/agi-bin
sudo nano nivel_pH.php
sudo chmod +x /var/lib/asterisk/agi-bin/ nivel_pH.php

#!/usr/bin/php -q
<?php
require('phpagi.php');
error_reporting(E_ALL);
$agi = new AGI();
$agi-> answer();

$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

$query="SELECT * FROM nivel_pH order by fecha_pH desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("sensor de nivel de ph");
$query="SELECT * FROM nivel_pH order by fecha_pH desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("el nivel de ph es $_re[pH]... la fecha es $_re[fecha_pH]");
$agi-> hangup();
?>
```

D) Programación para la lectura del último valor sensado que se encuentra fuera de rango, almacenado en la base de datos.

Sensor de Temperatura de Agua

```
cd /var/lib/asterisk/agi-bin
sudo nano leereportempagua.php
sudo chmod +x /var/lib/asterisk/agi-bin/ leereportempagua.php

#!/usr/bin/php -q
<?php
require('phpagi.php');
error_reporting(E_ALL);
$agi = new AGI();
$agi-> answer();
```

```
$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

$query="SELECT * FROM temperatura_agua WHERE gradC not BETWEEN 28 and 32
order by fecha_tagua desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("sensor de temperatura de agua");
$agi->text2wav("El ultimo valor sensado fuera de rango se detalla a continuacion");
$query="SELECT * FROM temperatura_agua WHERE gradC not BETWEEN 28 and 32
order by fecha_tagua desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("la temperatura del agua es $_re[gradC] grados... la fecha es
$_re[fecha_tagua]");
$agi-> hangup();
?>
```

Sensor de Temperatura Ambiental

```
cd /var/lib/asterisk/agi-bin
sudo nano leereportempamb.php
sudo chmod +x /var/lib/asterisk/agi-bin/ leereportempamb.php

#!/usr/bin/php -q
<?php
require('phpagi.php');
error_reporting(E_ALL);
$agi = new AGI();
$agi-> answer();

$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

$query="SELECT * FROM temperatura_ambiente WHERE gradoC not BETWEEN 1 and
25 order by fecha_tamb desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("sensor de temperatura ambiental");
$agi->text2wav("El ultimo valor sensado fuera de rango se detalla a continuacion");
$query="SELECT * FROM temperatura_ambiente WHERE gradoC not BETWEEN 1 and
25 order by fecha_tamb desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("la temperatura ambiental es $_re[gradoC]... la fecha es $_re[fecha_tamb]");
$agi-> hangup();
?>
```

Sensor de Humedad Ambiental

```
cd /var/lib/asterisk/agi-bin
sudo nano leereporhumamb.php
sudo chmod +x /var/lib/asterisk/agi-bin/ leereporhumamb.php

#!/usr/bin/php -q
<?php
require('phpagi.php');
error_reporting(E_ALL);
$agi = new AGI();
$agi-> answer();

$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

$query="SELECT * FROM humedad_ambiente WHERE RH not BETWEEN 25 and 65
order by fecha_hum desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("sensor de humedad ambiental");
$agi->text2wav("El ultimo valor sensado fuera de rango se detalla a continuacion");
$query="SELECT * FROM humedad_ambiente WHERE RH not BETWEEN 25 and 65
order by fecha_hum desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("la humedad ambiental es $_re[RH]... la fecha es $_re[fecha_hum]");
$agi-> hangup();
?>
```

Sensor de Iluminación

```
cd /var/lib/asterisk/agi-bin
sudo nano leereporilum.php
sudo chmod +x /var/lib/asterisk/agi-bin/ leereporilum.php



#!/usr/bin/php -q
<?php
require('phpagi.php');
error_reporting(E_ALL);
$agi = new AGI();
$agi-> answer();

$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

$query="SELECT * FROM iluminacion WHERE lx not BETWEEN 25 and 3000 order by
fecha_lx desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);
```

```
$agi->text2wav("sensor de iluminacion");
$agi->text2wav("El ultimo valor sensado fuera de rango se detalla a continuacion");
$query="SELECT * FROM iluminacion WHERE lx not BETWEEN 25 and 3000 order by
fecha_lx desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("la iluminacion es $_re[lx]lux... la fecha es $_re[fecha_lx]");
$agi-> hangup();
?>
```

Sensor de nivel de pH

```
cd /var/lib/asterisk/agi-bin
sudo nano leereporph.php
sudo chmod +x /var/lib/asterisk/agi-bin/ leereporph.php

#!/usr/bin/php -q
<?php
require('phpagi.php');
error_reporting(E_ALL);
$agi = new AGI();
$agi-> answer();c

$conexion = mysql_connect('localhost','root','godinmylife') or die (mysql_error());
mysql_select_db('sensores',$conexion);

$query="SELECT * FROM nivel_pH WHERE pH not BETWEEN 6.5 and 7.5 order by
fecha_pH desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("sensor de nivel de ph");
$agi->text2wav("El ultimo valor sensado fuera de rango se detalla a continuacion");
$query="SELECT * FROM nivel_pH WHERE pH not BETWEEN 6.5 and 7.5 order by
fecha_pH desc";
$_result=mysql_query($query,$conexion);
$_re=mysql_fetch_array($_result);

$agi->text2wav("el nivel de ph es $_re[pH]... la fecha es $_re[fecha_pH]");
$agi-> hangup();
?>
```

E) Programación para la obtención del último valor sensado que se encuentra almacenado en la base de datos.

Sensor de Temperatura de Agua

```
cd /var/www/html/acuario
```

sudo nano ultimo_temperatura_agua.php

```
<?php
$conex = mysql_connect('localhost','arduin','godinmylife');
mysql_select_db('sensores',$conex);
$selecciona = "SELECT * FROM temperatura_agua order by fecha_tagua desc";
$valortagua = mysql_query($selecciona);
$recupera=mysql_fetch_object($valortagua);
echo $recupera->gradC;
?>
```

Sensor de Temperatura Ambiental

cd /var/www/html/acuario
sudo nano ultimo_temperatura_ambiental.php

```
<?php
$conex = mysql_connect('localhost','arduin','godinmylife');
mysql_select_db('sensores',$conex);
$selecciona = "SELECT * FROM temperatura_ambiente order by fecha_tamb desc";
$valortamb = mysql_query($selecciona);
$recupera=mysql_fetch_object($valortamb);
echo $recupera->gradoC;
?>
```

Sensor de Humedad Ambiental

cd /var/www/html/acuario
sudo nano ultimo_humedad_ambiental.php

```
<?php
$conex = mysql_connect('localhost','arduin','godinmylife');
mysql_select_db('sensores',$conex);
$selecciona = "SELECT * FROM humedad_ambiente order by fecha_hum desc";
$valorhamb = mysql_query($selecciona);
$recupera=mysql_fetch_object($valorhamb);
echo $recupera->RH;
?>
```

Sensor de Iluminación

cd /var/www/html/acuario
sudo nano ultimo_iluminacionn.php

```
<?php
$conex = mysql_connect('localhost','arduin','godinmylife');
mysql_select_db('sensores',$conex);
$selecciona = "SELECT * FROM iluminacion order by fecha_lx desc";
$valorlx = mysql_query($selecciona);
$recupera=mysql_fetch_object($valorlx);
echo $recupera->lx;
?>
```

Sensor de pH

```
cd /var/www/html/acuario
sudo nano ultimo_ph.php

<?php
$conex = mysql_connect('localhost','arduin','godinmylife');
mysql_select_db('sensores',$conex);
$selecciona = "SELECT * FROM nivel_pH order by fecha_pH desc";
$valorph = mysql_query($selecciona);
$recupera=mysql_fetch_object($valorph);
echo $recupera->pH;
?>
```

F) Programación en Python para la activación de alertas.

```
#!/usr/bin/python
import RPi.GPIO as GPIO
import time
from urllib2 import urlopen
from datetime import datetime, time
import urllib2
import json

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

GPIO.setup(18,GPIO.OUT)
GPIO.setup(17,GPIO.OUT)
GPIO.setup(27,GPIO.OUT)
GPIO.setup(22,GPIO.OUT)

# Funcion que usa el script para enviar las notificaciones a Pushetta
def sendNotification(token, channel, message):
    data = {
        "body" : message,
        "message_type" : "text/plain"
    }

    req = urllib2.Request('http://api.pushetta.com/api/pushes/{0}/'.format(channel))
    req.add_header('Content-Type', 'application/json')
    req.add_header('Authorization', 'Token {0}'.format(token))

    response = urllib2.urlopen(req, json.dumps(data))

a = 0
b = 0
c = 0
d = 0
while True :

  #Control para el sensor de humedad y temperatura ambiental
  sensordht22 = urlopen('http://192.168.1.6:80/acuario/ultimo_temperatura_ambiental.php')
```

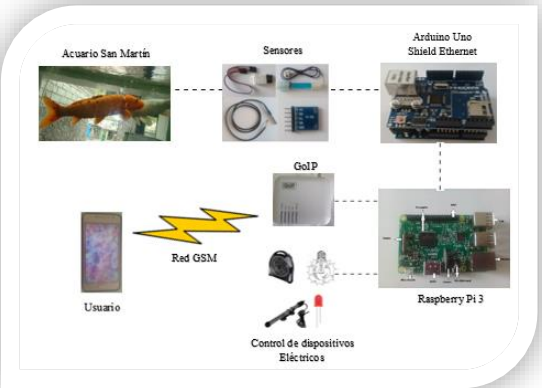
```
  tempdht22= float(sensordht22.read())
  if ( tempdht22 > 25 ) :
      GPIO.output(18, GPIO.LOW)
      a += 1
      if ( a == 1) :
          print("Alarma activada DHT22")
          sendNotification("4c744d705a1d416c522789577b496a177327329b", "SanMartin",
"Se ha activado el ventilador!!")
   else:
      GPIO.output(18, GPIO.HIGH)
      a = 0
      print("Alarma desactivada DHT22")


  #Control para el sensor de luz
  sensorluz = urlopen('http://192.168.1.6:80/acuario/ ultimo_iluminacionn.php')
  sensor1750 = float(sensorluz.read())
  tiempo = datetime.now()
  if ( sensor1750 < 25 and ( tiempo.hour >= 6 or tiempo.hour <= 18 ) ) :
      GPIO.output(22, GPIO.LOW)
      b += 1
      if ( b == 1) :
          print("Alarma activada LUZ")
          sendNotification("4c744d705a1d416c522789577b496a177327329b", "SanMartin",
"Se ha encendido la lampara")
  else:
      GPIO.output(22, GPIO.HIGH)
      b = 0
      print("Alarma desactivada LUZ")


  #Control para el sensor de temperatura de agua
  sensoragua = urlopen('http://192.168.1.6:80/acuario/ultimo_temperatura_agua.php')
  sensor18b20 = float(sensoragua.read())
  if ( sensor18b20 < 28  ) :
      GPIO.output(27, GPIO.LOW)
      c += 1
      if ( c == 1) :
          print("Alarma activada TEMP AGUA")
          sendNotification("4c744d705a1d416c522789577b496a177327329b", "SanMartin",
"Se ha encendido el calentador de agua")
  else:
      GPIO.output(27, GPIO.HIGH)
      c = 0
      print("Alarma desactivada TEMP AGUA")

  #Control para el sensor de pH
  sensorph = urlopen('http://192.168.1.6:80/acuario/ ultimo_ph.php')
  ph = float(sensorph.read())
  if ( (ph < 6.5) or (ph > 7.5)) :
      GPIO.output(17, GPIO.LOW)
      d += 1
      if ( d == 1) :
          print("Alarma activada ph")
```

```
            sendNotification("4c744d705a1d416c522789577b496a177327329b", "SanMartin",
"Porfavor cambiar de agua la pecera")
    else:
        GPIO.output(17, GPIO.HIGH)
        d = 0
        print("Alarma desactivada ph")
```

IV. Anexo D: Manual de uso e instalación

ACUARIO SERPENTARIO Y AVES EXÓTICAS "SAN MARTÍN"

Sistema de Control Electrónico de Acuarios utilizando Tecnología GSM y VoIP



Manual de instalación

1. Descargar la imagen de Sistema Operativo Raspbian de la página oficial de Raspberry Pi: https://www.raspberrypi.org/downloads/noobs/
2. Formatear la tarjeta micro SD, y cargar la imagen del Sistema Operativo.
3. Insertar la tarjeta micro SD en la Raspberry Pi y encenderla.
4. Iniciar con el proceso de Instalación, eligiendo el Sistema Operativo Raspbian, el idioma y teclado en español.
5. Una vez culminada la instalación se procede a actualizar los paquetes y versiones disponibles, con los comandos sudo apt-get update y sudo apt-get upgrade.
6. Asignar una dirección IP estática a la Raspberry Pi.
7. Instalar PHP, servidor Apache, base de datos MySQL, PhpMyAdmin, Servidor FTP y Arduino.
8. Ingresar a PhpMyAdmin y crear la base de datos "sensores".
9. Crear las tablas para cada sensor en la base de datos "sensores".
10. Programar en arduino para lectura de los sensores y envío de información a la base de datos "sensores".
11. Crear los scripts para la conexión con PhpMyAdmin, declarando el nombre del usuario "arduin", la clave "godinmylife" y el nombre de la base de datos "sensores".

12. Crear los scripts para el envío de la información de cada sensor a la base de datos "sensores".
13. Conectar Arduino Uno con la Shield Ethernet para el envío de información, a la base de datos "sensores".
14. Instalar Asterisk, Festival, Voces en español para Asterisk y Festival, AGI y las librerías GPIO.
15. Crear los scripts para la lectura de la información almacenada de cada sensor que se encuentra en la base de datos "sensores" mediante Asterisk.
16. Crear los scripts para la lectura de información almacenada de cada sensor que se encuentra en la base de datos "sensores" mediante Asterisk, cuando un valor se encuentra fuera de rango.
17. Configurar todo lo relacionado con el protocolo SIP y el dialplan en el archivo de configuración sip.conf y extensions.conf, respectivamente.
18. Crear el IVR soporte, asignando extensiones para la comunicación y obtención de la información almacenada por cada en sensor en la base de datos "sensores".
19. Crear el IVR reporte, asignando extensiones para la comunicación y obtención de la información almacenada por cada en sensor que se encuentra fuera de rango, en la base de datos "sensores".
20. Configurar el GoIP para comunicar la red de teléfono móvil, con la red de telefonía VoIP.
21. Crear una cuenta y un canal en la aplicación Pushetta para la notificación de alertas.
22. Programar en Python para la activación de alertas, mediante la activación de puertos GPIO en la Raspberry Pi.
23. Conectar el módulo Relé para activar el ventilador, el calentador de agua, la lámpara fluorescente o la luz piloto, cuando el sistema lo requiera.
24. Diseñar una placa, con el circuito que permite integrar todo el Sistema de Control Electrónico para el Acuario "San Martín".

Manual de uso

---

1. Llamar al número celular "098790XXXX".
2. Para consultar el último valor almacenado que se encuentra en la base de datos "sensores", marcar "1".
3. Elegir el sensor que se desea consultar:
    - "1" para el sensor de humedad ambiental
    - "2" para el sensor de iluminación

- “3” para el sensor de pH
- “4” para el sensor de temperatura del agua
- “5” para el sensor de temperatura ambiental

4. Llamar al número celular “098790XXXX”.
5. Para consultar el último valor almacenado que se encuentra en la base de datos “sensores”, que se encuentra fuera de rango, marcar “2”.
6. Elegir el sensor que se desea consultar:
    - “1” para el sensor de humedad ambiental
    - “2” para el sensor de iluminación
    - “3” para el sensor de pH
    - “4” para el sensor de temperatura del agua
    - “5” para el sensor de temperatura ambiental.
7. Ingresar a Pushetta y verificar la recepción de las notificaciones cuando el sistema de alertas se active.

V. Anexo E: Acuario Serpentario “SAN MARTÍN”

a) Carta de compromiso del Acuario Serpentario “SAN MARTÍN”

ACUARIO SERPENTARIO
ANIMALES EXÓTICOS “SAN MARTIN”

CARTA DE COMPROMISO

Baños 01 de Febrero de 2017

Ingeniera Mg.
Pilar Urrutia
DECANA
Facultad de Ingeniería en Sistemas, Electrónica e Industrial
Presente.

Señora Decana:

En mi calidad de propietario del Acuario Serpentario y Animales exóticos “San Martín”, por medio de la presente manifiesto a usted, que la Srta Jessica Andrea Vivanco Correa, portadora de la cédula de ciudanía N° 1719383943, estudiante de Décimo nivel en el período académico Octubre 2016 – Marzo 2017 de la Carrera de Ingeniería en Electrónica y Comunicaciones de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, tiene mi autorización y respaldo para desarrollar en la empresa que represento el Trabajo de Titulación denominado “Sistema de control electrónico para el Acuario Serpentario San Martín del cantón Baños de la provincia de Tungurahua utilizando tecnologías GSM y VoIP ”, para lo cual designo como Tutor Empresarial al Ing. William Germán Vega Merino.

Con estos antecedentes informo que la realización de este Trabajo de Titulación es de gran importancia para la empresa, el estudiante tiene total apoyo para su desarrollo y ejecución, por lo expuesto anteriormente solicito se apruebe y se proceda con el trámite correspondiente.

Atentamente,

ZOOACUARIO SERPENTARIO AVES EXÓTICAS BAÑOS - TUNGURAHUA SAN MARTIN

REPRESENTANTE DE LA INSTITUCION
LEONARDO VEGA MARIÑO
PROPIETARIO

Vía Lligua Sector San Martín
Teléfono 2740994

b) Finalización del proyecto en el Acuario Serpentario "SAN MARTÍN"

ACUARIO SERPENTARIO
ANIMALES EXÓTICOS "SAN MARTIN"

Baños, 26 de julio de 2017

Ingeniera M.Sc.
Pilar Urrutia U.
DECANA
Facultad de Ingeniería en Sistemas, Electrónica e Industrial
Presente

Señora Decana:

Por medio del presente, en calidad de representante legal de esta empresa certifico que el trabajo de investigación: "SISTEMA DE CONTROL ELECTRÓNICO PARA ACUARIOS UTILIZANDO TECNOLOGÍAS GSM Y VOIP", desarrollado por la señorita: Jessica Andrea Vivanco Correa, ha sido concluido de conformidad a los intereses de la Empresa.

Por la atención que se sirva dar al presente, me suscribo de usted.

Atentamente,

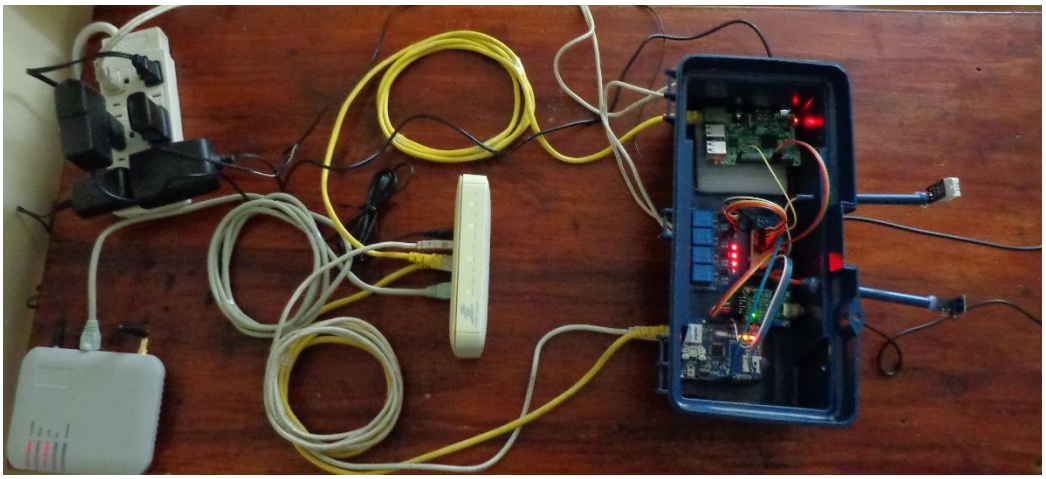
ZOOACUARIO SERPENTARIO AVES EXOTICAS "SAN MARTIN" BAÑOS - TUNGURAHUA

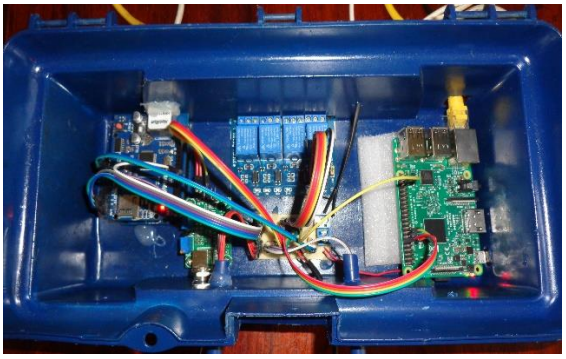
Ing. William German Vega Merino
Representante

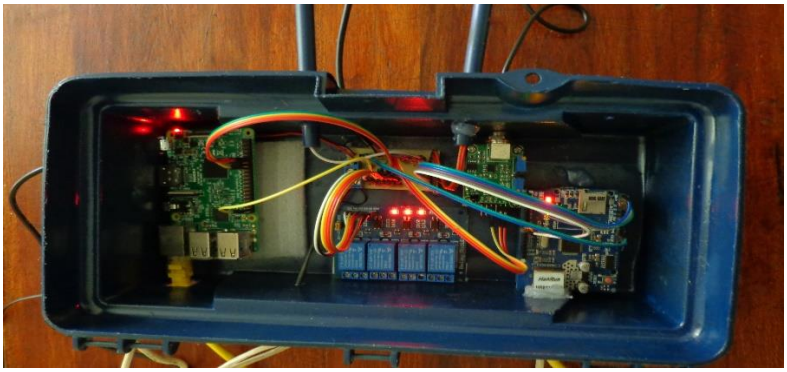
Vía Lligua Sector San Martín

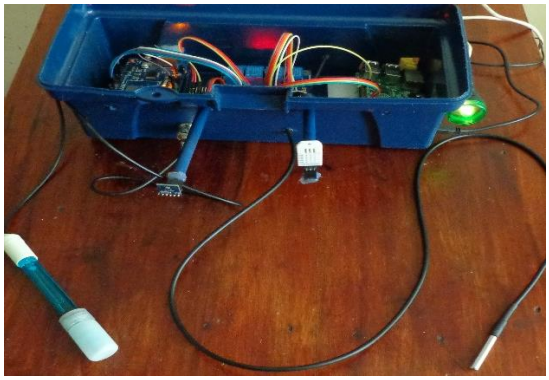
Teléfono 2740994

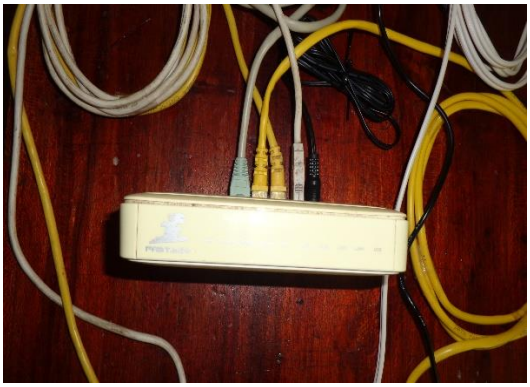
VI. Anexo F: Fotografías de la instalación

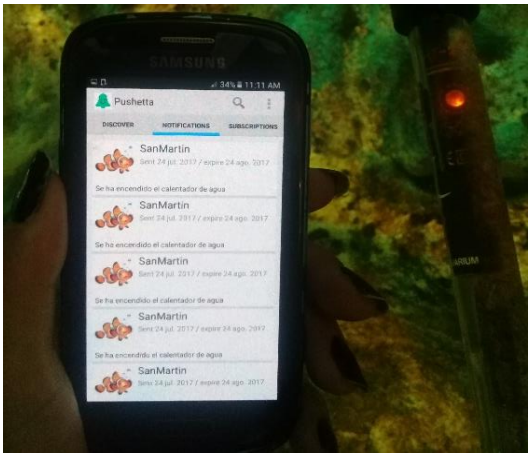


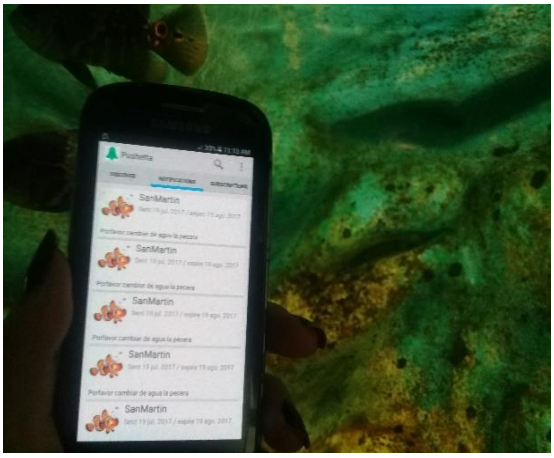


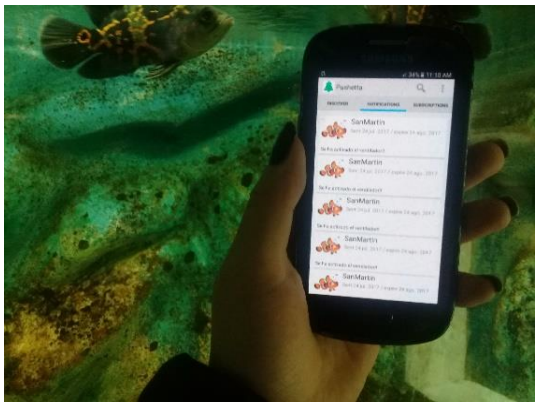


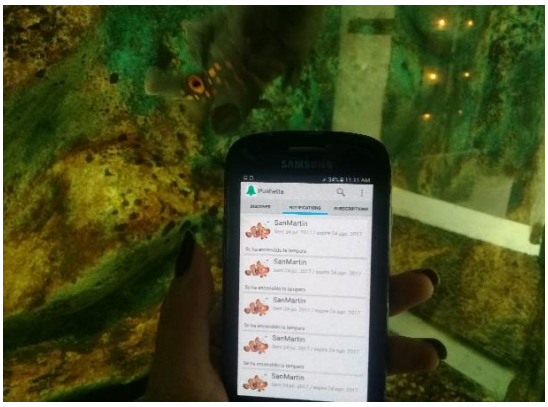


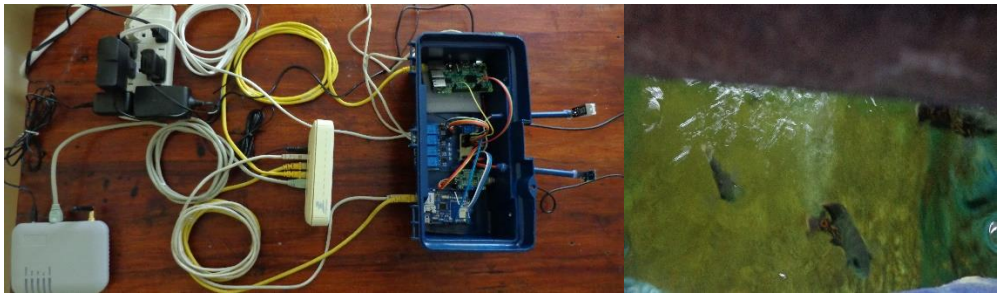



Pushetta
DISCOVER
NOTIFICATIONS
SUBSCRIPTIONS
SanMartin
Se ha encendido el calentador de agua










Preservar
Cuidar
Proteger
VIDA EXOTICA
OSCAR
(Astronotus ocellatus)
Omnívoro
max 36cm
Perú, Colombia, Brasil