UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Internet de las Cosas (IoT) aplicado a un caso de Agricultura Urbana

TESIS

Que para obtener el título de

Ingeniero en Telecomunicaciones

P R E S E N T A

Salas González Marco Antonio

DIRECTOR DE TESIS

M.I. Jesús Reyes García

Ciudad Universitaria, Cd. Mx., 2023

A Bety, quien siempre se ha esforzado por darnos todo.

Contenido

Introducción y objetivos del proyecto

Al ser la ciudad más grande e importante del país, la Ciudad de México cuenta con una gran cantidad de servicios que ofrecerles a sus habitantes para mejorar su calidad de vida. La Ciudad de México es famosa en el país y en algunas regiones de Latinoamérica en temas de salud, educación, empleo y seguridad. Razón por la cual es bastante atractiva y cada año miles de personas se mudan temporal o permanentemente a la capital del país.

Teniendo en cuenta los millones de personas que habitan la capital y los miles que llegan a vivir a ella cada año, se vuelve un reto importante el abasto de alimentos. Desafortunadamente México, al igual que la mayoría de países subdesarrollados, tiene una dependencia importante de la agricultura tradicional para abastecerse de alimentos. Lo anterior implicaría que, con el paso de los años al aumentar la población en la Ciudad de México se tendría que aumentar la producción de alimentos en los campos agrícolas tradicionales.

Para aumentar la producción y satisfacer la demanda de la población, la solución más económica y directa que las autoridades podrían implementar seria aumentar la cantidad de hectáreas de tierra destinadas a la agricultura, pero en la actualidad hay suficientes investigaciones para confirmar que esta medida trae más problemas que beneficios debido al daño ecológico que ocasiona el cambio de suelo. La solución más viable, pero no más económica, sería la implementación de técnicas agrícolas modernas que hacen uso de la tecnología, logrando una mayor producción.

El crecimiento acelerado de la población en la Ciudad de México hace necesario que se busquen estrategias que logren que al menos una parte de la población pueda autoabastecerse de alimentos. Estas estrategias deben ser creadas basándose en las características de la ciudad y el espacio disponible, principal razón por la cual se descarta el uso de agricultura tradicional debido a las grandes cantidades de tierra necesarias para su implementación.

La capital no es la única ciudad con problemas de abasto de alimentos. Alrededor del mundo muchas ciudades tienen el mismo problema y han buscado diferentes soluciones, algunas más eficientes que otras. Basándose en la experiencia de las diferentes soluciones que se le han dado a este problema, una de las técnicas más fiables es la implementación de la Hidroponía, un método agrícola milenario que permite el cultivo en agua o cualquier tipo de sustrato, por lo que el suelo deja de ser necesario, pudiendo implementar esta técnica en recipientes colocados en espacios pequeños dentro de los hogares **[1]**.

La palabra hidroponía nace del griego antiguo, que en español se traduce como “trabajo en agua”. La composición de la palabra da la falsa idea de que la hidroponía es una técnica agrícola que solo utiliza agua para el desarrollo de un cultivo, pero también son necesarios otros elementos y el control de parámetros como el nivel de pH, la temperatura y la humedad que garantizan el crecimiento de una planta **[1, 2, 3, 4, l]**.

El desarrollo de un cultivo hidropónico requiere de tiempo y dedicación para conseguir que los parámetros necesarios para el desarrollo de una planta se mantengan estables, porque de lo contrario el cultivo moriría **[3, 4]**. Debido a que la mayoría de la población de la Ciudad de México cuenta con poco tiempo libre, se puede hacer uso de la tecnología para el monitoreo de los parámetros sin la necesidad de intervención humana. Una de las mejores tecnologías para esta actividad es el Internet de las Cosas

(*Internet of Things*, IoT), que en estos últimos años ha tenido un crecimiento exponencial debido a la gran cantidad de aplicaciones que se le han encontrado **[5, II]**.

El IoT consiste en una red de dispositivos interconectados capaces de dar inteligencia a aparatos y objetos de la vida cotidiana, esto con la finalidad de monitorear el entorno y brindar al usuario final servicios personalizados que le generen una mejor calidad de vida. Los sensores y actuadores son dos piezas fundamentales del Internet de las Cosas debido a que a través de ellos la tecnología puede interactuar con el entorno. Otra pieza clave, y la que da el nombre a esta tecnología, es la conexión a través de la red, que permite la interacción entre diferentes sistemas a través de un direccionamiento IP **[5, 6, 7, 8, 9, 10, II, III]**.

La gran cantidad de sensores disponibles hoy en día hacen posible registrar cualquier tipo de parámetro y con la información obtenida un sistema electrónico puede ser capaz de modificar dichos parámetros a través de actuadores. Utilizando estos dos elementos clave, es posible la creación de un sistema inteligente capaz de monitorear y controlar los parámetros necesarios para mantener un cultivo hidropónico. Además de sensores y actuadores, también son necesarios dispositivos de conexión a la red, placas electrónicas, memorias, microprocesadores, etc., cuya función será explicada más adelante en este proyecto.

El objetivo de este trabajo consiste en la creación de un sistema inteligente, capaz de monitorear y mantener estables los parámetros necesarios de un cultivo hidropónico que pueda ser implementado fácilmente en los hogares de la Ciudad de México. Se busca crear un sistema barato y fácil de manejar, haciéndolo accesible para la mayoría de la población. El sistema constará de un invernadero que al crear un entorno cerrado facilitará la manipulación de los parámetros, los cuales serán registrados por sensores que se comunicaran a una tarjeta *Raspberry* (RB), posteriormente la RB será capaz de activar o desactivar actuadores que mantendrán estables los parámetros. Además del control del entorno, el sistema contará con autorriego, reduciendo de esta manera la interacción humana, por lo que el tiempo que los usuarios dedicaran a este sistema será mínimo. El enrutamiento a la red se realizará a través de la Raspberry, que cuenta con conexión a Internet vía Wifi, de esta manera será posible notificar por correo electrónico el estado del sistema o alguna falla que pudiera generarse, también estará a disposición una página web local donde se podrá consultar el estado del invernadero.

Capítulo 1: IoT –Internet de las cosas

1.1 ¿Qué es IoT?

A lo largo de la historia, los seres humanos hemos buscado facilitar nuestras tareas y actividades diarias a través de la invención de dispositivos, mecanismos y herramientas. Ciertos inventos y descubrimientos lograron sobresalir del resto debido al cambio tan drástico que ocasionaron en la humanidad. Aunque existen decenas de inventos que han cambiado nuestras vidas, los historiadores consideran que solo en cuatro ocasiones se logró un cambio importante como para considerarlo una Revolución Industrial, la palabra “Industrial” fue agregada al nombre debido a que la industria fue la más beneficiada con estos cambios. La primera revolución industrial surgió en Inglaterra en el siglo XVIII con el invento de la máquina de vapor, esta revolución logró que a través de la energía emanada del vapor se lograra una mayor producción en las fábricas debido a la utilización de máquinas y la invención de transportes más grandes como la locomotora y el barco de vapor. La segunda gran revolución industrial surgió con la invención de la electricidad y el motor eléctrico, inventos que aceleraron el cambio en la industria debido a que la electricidad, junto con el petróleo, comenzaron a ser las principales fuentes de energía, y el motor permitió la invención de nuevas máquinas, como el automóvil. La tercera revolución industrial nació junto con la computadora, el Internet, la electrónica y los primeros sistemas de automatización, lo que permitió elevar mucho más la producción gracias a la configuración de sistemas automáticos y al monitoreo a distancia. En la actualidad, nos encontramos viviendo la cuarta revolución industrial, caracterizada por el Big Data, la Nube, y el Internet de las Cosas que, a pesar de que aún se encuentran en desarrollo, ya ofrecen sus primeros servicios, como las reuniones virtuales, el seguimiento en tiempo real, la reproducción de videos vía Streaming, automóviles inteligentes, la telemedicina, etc., que hacen que la cuarta revolución industrial sea más prometedora que las tres anteriores debido a que puede ser implementada en todos los campos.

De la 4ta Revolución Industrial, el Internet de las Cosas será uno de los temas de mayor impacto debido a que será clave en el desarrollo de nuevas tecnologías que facilitaran la vida de los seres humanos al darle inteligencia a objetos cotidianos que normalmente no la tienen. La inteligencia es dada a través de un conjunto de dispositivos que se comunican entre sí para intercambiar información principalmente del estado del entorno **[5, 7, 9, 11, II, III]**.

Dentro de las primeras aplicaciones IoT que ya se han comenzado a comercializar se encuentran los dispositivos llamados *Wearables*, traducidos al español como *Tecnología Vestible*, que son accesorios diarios como ropa, relojes, joyería, etc., que cuentan con diferentes tipos de sensores para llevar un control de diferentes parámetros como la temperatura corporal, la frecuencia cardiaca, el nivel de oxígeno en la sangre, las calorías quemadas, y más, los cuales son enviados a servidores en la nube o a algún almacenamiento temporal para ser consultados desde el celular o cualquier otro dispositivo compatible.

Entre la *Tecnología Vestible* más famosa se encuentran los relojes inteligentes de *Huawei*, que cuentan con uno de los sistemas de sensores más fiables y completos. Recientemente se anunció un nuevo reloj capaz de medir la presión arterial a través de sensores instalados en la pulsera del reloj. Este ejemplo, junto con otros modelos de *Wearables*, como la ropa inteligente y los collares con GPS demuestran que el IoT cada vez se adentra más a los objetos cotidianos haciendo que ofrezcan más servicios a los usuarios.

Figura 1.1.1: Huawei Watch D. Tomado de https://consumer.huawei.com/mx/wearables/watch-d/

Los *Wearables* son actualmente los dispositivos basados en IoT más populares debido a sus bajos costos, pero el Internet de las Cosas ha entrado en todos los campos, desde el hogar hasta la industria, más delante se hablará de sus aplicaciones **[5]**.

Aunque las investigaciones y el desarrollo de aplicaciones del Internet de las Cosas ya se encuentran muy avanzadas y no es una tecnología nueva, aún no han logrado llegar a un acuerdo para la definición universal del IoT debido a que se puede interpretar de diferentes maneras según el ángulo desde el que se vea esta tecnología **[12]**. Visto desde un ángulo de aplicación se puede definir al Internet de las Cosas como un conjunto de sensores y actuadores que intercambia información a través de redes bidireccionales para dar inteligencia objetos cotidianos ofreciendo diferentes servicios, a través de un sistema, a usuarios finales. Dichos servicios pueden ser un monitoreo, un control, una optimización, una autonomía de funcionamiento y/o la toma de decisiones inteligentes de un sistema **[9, 10, 12]**. Por su parte, CISCO define al Internet de las Cosas como el momento del tiempo en el que el número de dispositivos conectados a Internet fue mayor al número de personas en el mundo. Según los estudios realizados por CISCO, este momento se alcanzó entre los años 2008 y 2009 **[5]**.

Tomando la definición de IoT vista desde un ángulo de aplicación, es necesario mencionar más a profundidad las características técnicas de un sistema basado en esta tecnología.

Jordi Salazar, en su trabajo llamado *Internet de las Cosas* publicado por *Erasmus*+, define a los sistemas IoT como la fusión de tres elementos clave, los Sensores y Actuadores, la Conectividad y, los Procesos y Usuarios **[9]**:

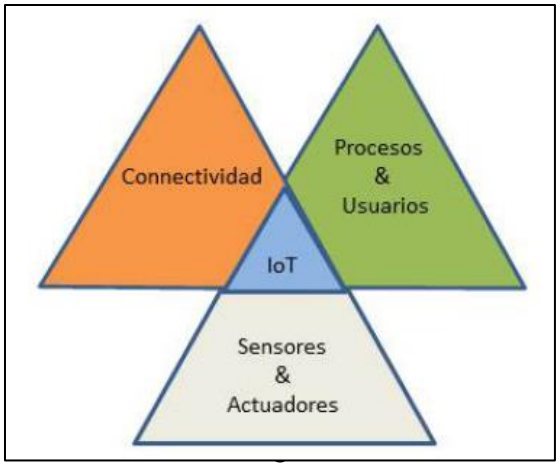
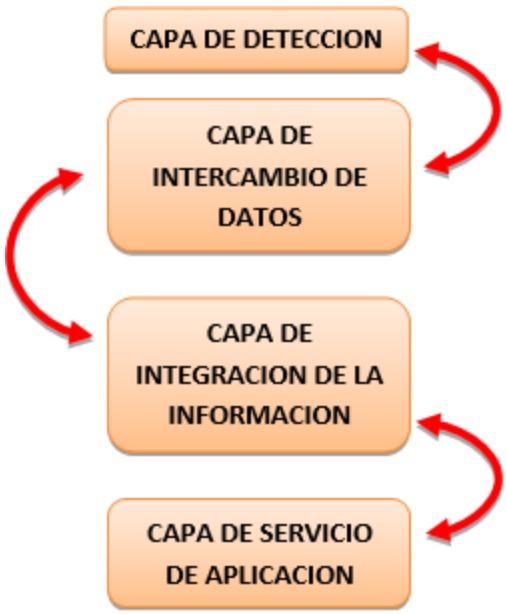


Figura 1.1.2: Concepto IoT. Nota. Adaptado de "Internet de las Cosas" (p. 14), por Jordi Salazar, Erasmus+

Basado en la imagen anterior, elaboró una arquitectura para sistemas IoT conformada por cuatro capas:



La arquitectura de 4 capas es muy similar a las arquitecturas TCP/IP y OSI de redes de datos. Cada capa tiene una tarea asignada y solo se puede comunicar con sus capas adyacentes para compartir información del sistema. A continuación, se describen las tareas realizadas por cada una de las capas.

1° CAPA DE DETECCION: Esta capa contiene algunos elementos físicos del sistema basado en IoT como sensores, cables de conexiones, tarjetas, microcontroladores, módulos, etc., que recolectaran e intercambiaran datos.

Los sensores son los elementos más importantes de esta capa debido a que son los encargados de la detección de las magnitudes físicas. Actualmente el número de sensores es elevado, por lo que para clasificarlos es necesario algún tipo de criterio **[13]**. A continuación, se hace una recopilación de los principales tipos de detectores tomando en cuenta diferentes criterios **[14]**:

1) Según el aporte de energía:
   - Activos: Son aquellos en los cuales la señal de salida depende de una fuente de alimentación externa.
   - Pasivos: La señal de salida depende solo de la fuente de alimentación de la entrada.
2) Según el tipo de salida:
   - Analógicos: La señal de salida presenta valores de amplitud continuos.
   - Digitales: Los valores de la señal de salida son discretos.
3) Según el modo de funcionamiento:
   - De deflexión: En este caso, la magnitud física medida produce algún efecto físico sobre el sensor, el cual, produce un efecto similar en magnitud, pero opuesto al producido por la magnitud física, con la finalidad de cancelar dicho efecto.

- De comparación: Este tipo de sensores comparan el efecto producido por la magnitud física con valores ya conocidos.

4) Según la magnitud medida:
   - De temperatura,
   - De humedad,
   - De presión,
   - De posición, etc.
5) Según el parámetro variable:
   - Resistivo,
   - Capacitivo,
   - Inductivo.

2° CAPA DE INTERCAMBIO DE DATOS: En esta capa se lleva a cabo la transmisión de los datos recopilados por los sensores. La transmisión es transparente, es decir, que la información se envía a una base de datos, o directamente a un procesador, tal cual es obtenida por los sensores. En este punto, el sistema no es capaz de reconocer si la información contiene errores o si es coherente. La recolección e intercambio de la información son realizados a través del medio de transmisión y el protocolo que mejor se adapte a las necesidades del sistema. Para el caso de sistemas pequeños la información puede ser transportada por diferentes medios de transmisión, los cuales serán enlistados junto a sus principales características **[15]**:

1) Cable multifilar:
   - Distancia máxima promedio: > 1 [km]
   - Velocidad máxima promedio: Pocos [Mbps]
   - Aplicaciones: Audio, Redes Caseras, Electrónica, Antenas, etc.
2) Par trenzado:
   - Distancia máxima promedio: 100 [m]
   - Velocidad máxima promedio: 100 [Mbps]
   - Aplicaciones: Telefonía, Internet, Redes LAN, CCTV, etc.
3) Cable Coaxial:
   - Distancia máxima promedio: 1 [km]
   - Velocidad máxima promedio: 10 [Mbps]
   - Aplicaciones: Telefonía, Internet, Redes LAN, TV, Aparatos de Medición, etc.
4) Wifi:
   - Distancia máxima promedio: 15 [m]
   - Velocidad máxima promedio: 2 [Gbps]
   - Aplicaciones: Conexión a Internet, Redes LAN.
5) Bluetooth:
   - Distancia máxima promedio: 10 [m]
   - Velocidad máxima promedio: 25 [Mbps]
   - Aplicaciones: Redes PAM, Audio, Internet, Envío de Archivos.

Como fue mencionado anteriormente, los datos tienen dos opciones, ser almacenados para ser usados posteriormente, o ser utilizados directamente por el sistema. La manera usual de almacenar información es a través de bases de datos, siendo las más comunes:

- SQL,
- Oracle,
- DB2,
- Microsoft Access, etc.

3° CAPA DE INTEGRACION DE LA INFORMACION: Es la capa más importante del sistema. Aquí es donde comienza el filtrado de la información recolectada para decidir si es útil, si está libre de errores o si es necesario solicitar nuevamente una recolección de información. La información que pasa el primer filtro se convierte en información útil y es analizada por el programa alojado en el microcontrolador para que el sistema tome las mejores decisiones basadas en la información recopilada por las capas anteriores.

La inteligencia del sistema es otorgada por el lenguaje de programación compatible con el microcontrolador, siendo los más comunes C, C++, Ensamblador y Python. Los cuales son capaces de manipular dispositivos físicos, lo que los hace ideales para proyectos de electrónica. Además de esto, algunos lenguajes, como Python, pueden enviar y recibir información a través de la red, lo que es de bastante utilidad para desarrollar proyectos IoT **[10]**.

4° CAPA DE SERVICIO DE APLICACIÓN: Es la capa final, y para el usuario es la capa más importante. Esta capa ofrece el servicio final y los resultados entregados al usuario dependen de las decisiones tomadas en la capa de integración de la información.

En caso de que el servicio final esté relacionado con la automatización y control de parámetros, los sistemas utilizan actuadores, que son los elementos capaces de intervenir en los parámetros que se requiere controlar o manipular, es decir, los actuadores son capaces de convertir energía, principalmente eléctrica, en otro tipo de energía, principalmente mecánica, para controlar parámetros como temperatura y humedad **[13]**. Los actuadores se clasifican en:

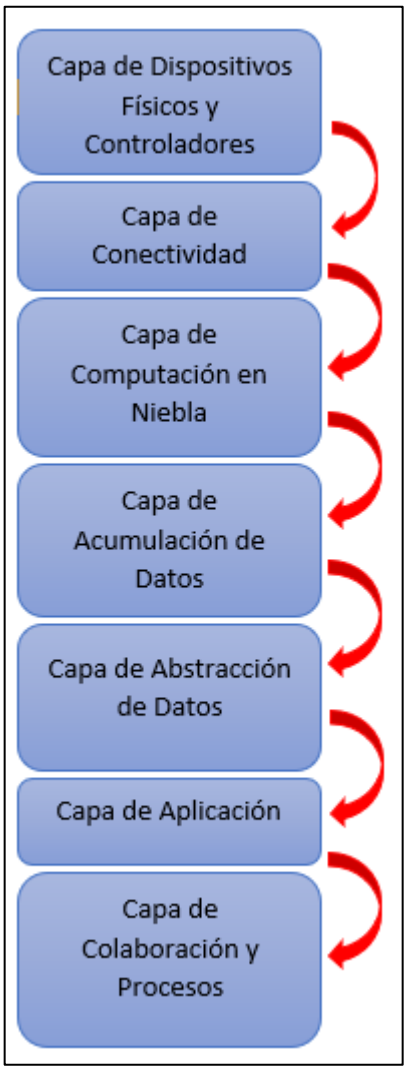
- Eléctricos,
- Hidráulicos, y
- Neumáticos.

El tipo de actuador a utilizar depende de la aplicación.

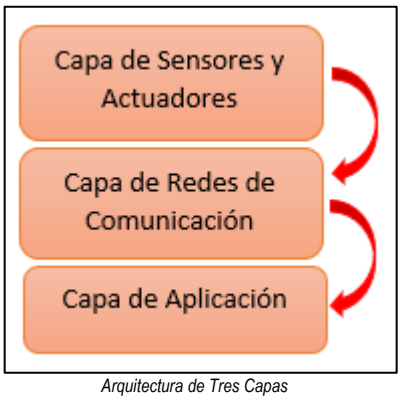
Al terminar de diseñar esta capa se deben realizar pruebas de funcionamiento del sistema para garantizar que realiza el servicio para el cual fue diseñado, así mismo, las pruebas ayudaran a encontrar fallas y anomalías que no se observaron en las capas inferiores.

Como se mencionó anteriormente, existen distintas arquitecturas para los sistemas basados en IoT, la mayoría de tres o cuatro capas. El número de capas se relaciona con la complejidad de las aplicaciones, debido a que el número depende de la cantidad de tareas realizadas por el sistema que se está diseñando.

Un ejemplo de una arquitectura diferente es CISCO, quien cuenta con dos tipos de arquitectura, una basada en siete capas, que es utilizada en sistemas avanzados que necesitan un diseño exhaustivo. Y una arquitectura simplificada que solo cuenta con tres capas, que son suficientes para sistemas pequeños **[16]**.



Arquitectura de Siete Capas



Arquitectura de Tres Capas

Aunque las arquitecturas fueron diferentes en la mayoría de fuentes consultadas, el principio de funcionamiento es el mismo en todas ellas. En este proyecto se optó por utilizar la primera arquitectura mostrada, de cuatro capas, debido a que fue la más utilizada en las fuentes consultadas.

Dentro de los servicios ofrecidos por un sistema IoT se encuentran **[9]**:

- Monitorización: Este servicio entrega al sistema la información pasada y/o en tiempo real de algún parámetro del entorno. La información es obtenida a través de una red de sensores. Su función solo es informar el estado del entorno. Para este servicio es necesario contar con un medio de almacenamiento para guardar la información para futuras consultas.
- Control: En un sistema IoT el control sirve para alterar los parámetros del entorno a través de actuadores que se activan o desactivan según la necesidad. Un ejemplo de este servicio seria la manipulación de la temperatura activando o desactivando sistemas de ventilación o calentamiento.
- Optimización: El servicio de optimización tiene como objetivo principal el ahorro de energía. En este servicio se utilizan sensores para conocer las condiciones del sistema y determinar si algún dispositivo debe ser accionado. Un ejemplo de este servicio es el encendido de iluminación, para

este caso un sensor de luz podría ser instalado, y dependiendo de la cantidad de luz natural en el lugar el sistema podría ser capaz de determinar el mejor momento para encender la luz artificial, evitando encender las luces cuando la iluminación natural aun es buena.
- Autonomía: El servicio de autonomía es uno de los principales servicios del IoT debido a que se espera que todos los sistemas, o la mayoría, sean independientes del ser humano permitiéndole dedicarse a otras actividades con la seguridad de que los sistemas son capaces de funcionar sin supervisión.
- Toma eficiente de decisiones: Para que un sistema sea considerado inteligente es necesario contar con el servicio de toma de decisiones, de esta manera se garantiza que el sistema sabrá actuar en todas las situaciones.

Todos estos servicios son independientes uno de otro y puede darse el caso de que un sistema IoT no cuente con alguno de estos servicios.

1.2 Aplicaciones de IoT

El Internet de las Cosas está impactando significativamente debido a que es posible aplicarlo en todos los campos. Los campos donde actualmente se están desarrollando aplicaciones son:

- Hogar,
- Salud,
- Ciudades inteligentes y transporte,
- Educación,
- Electrónica de consumo,
- Automoción (Seguridad de los vehículos),
- Agricultura y medio ambiente,
- Servicios de energía,
- Conectividad inteligente,
- Industria y producción, y,
- Comercio.

A través del IoT es posible monitorear, mediante sensores, los signos vitales de pacientes que se encuentran en sus hogares, y que, en caso de presentar anomalías, los sistemas IoT notifican a familiares y servicios de emergencia para brindarles asistencia. También, es posible monitorear remotamente cualquier lugar mediante un CCTV conectado a Internet, y que con una red de sensores de movimiento sea capaz de alertar sobre la entrada de intrusos a zonas restringidas **[9]**.

Se espera que, en el futuro, cuando la red 5G comience a ofrecer sus servicios, más aplicaciones comiencen a ser una realidad en la población, como el uso de automóviles totalmente autónomos que sean capaces de determinar la mejor ruta, que cuenten con sensores de proximidad para evitar accidentes y que ofrezcan servicios personalizados al cliente basados en sus gustos. Otra de las aplicaciones bastante prometedora es la telemedicina, mediante la cual será posible realizar operaciones y procedimientos médicos vía remota, donde el cirujano se encontrará manipulando los aparatos médicos probablemente a miles de kilómetros de distancia mediante un sistema IoT. Para poder gozar de estas y muchas otras aplicaciones será necesario modificar la red actual y sus componentes debido a la cantidad de información que deberá ser procesada, la cual sobrepasa los límites de los sistemas actuales **[12]**.

La agricultura ha sido uno de los campos en los que la tecnología siempre ha intervenido buscando optimizar los procesos, reducir los costos y obtener mayores ganancias, entre muchos otros objetivos. En este caso, el IoT se ha centrado principalmente en cinco aplicaciones **[8]**:

1° Sistemas de Información Geográfica (SIG) y GPS: A través de un sistema IoT, es posible que, con una red de sensores conectados a un vehículo aéreo o terrestre, un SIG y un GPS se realice el mapeo de zonas agrícolas, de esta manera se conocería el área de trabajo exacta, la latitud, la altura respecto al nivel del mar, el declive, la temperatura, la humedad, el tipo de terreno, etc. Todos estos elementos permitirían crear estrategias previas al cultivo como la optimización del espacio, un buen diseño del sistema de riego, entre otras.

2° Información Satelital mediante espectros: La información satelital puede ser muy valiosa para el IoT debido a que es posible llevar un control en tiempo real de las amenazas mediante sensores colocados en el satélite, los cuales son capaces de medir parámetros con diferentes espectros, como el infrarrojo, que mide temperatura. Entre la información que es posible obtener desde un satélite se encuentra la cantidad de vegetación en un cultivo, la cantidad de plantas marchitas, la cantidad de nitrógeno en el suelo y la cantidad de nutrientes, que mediante su análisis es posible encontrar y solucionar amenazas latentes.

3° Drones: En la actualidad los drones se han convertido en una herramienta importante para el ser humano, permitiendo el monitoreo aéreo de zonas de difícil acceso y realizando actividades agrícolas de una manera eficiente; como pude ser el riego de fertilizantes, el esparcimiento de semillas y la fumigación.

4° Softwares agrícolas y datos en línea: En la agricultura se ha comenzado el desarrollo de softwares y aplicaciones que tienen como objetivo facilitar el monitoreo, capacitar a los agricultores y acceder de manera remota a los parámetros involucrados en un cultivo. Con una red de sensores es posible medir los parámetros que pueden ser direccionados a Internet donde son accesibles a quien los requiera.

5° Conjunto de Datos: Uno de los objetivos del IoT es la comunicación de los dispositivos (M2M) y los sistemas. Un conjunto de Datos se refiere a la interacción de diferentes sistemas que realizan las mismas tareas, con la finalidad de compararse entre ellos y determinar en qué puntos un sistema es más eficiente que otro, además de compartirse información del estado de sus sistemas y alertas de posibles amenazas. Para el caso de la agricultura un conjunto de datos sería importante para que un agricultor conociera el estado de los cultivos de sus vecinos y determinar si su cultivo está dentro del promedio. También, podría tomar precauciones en caso de que los sistemas informaran de plagas dentro del conjunto.

Capítulo 2: Invernaderos

2.1 Invernaderos Convencionales e Invernaderos Hidropónicos

A pesar de que la agricultura ha sido parte de la vida de los seres humanos por miles de años, en la mayoría de las poblaciones aún no se logra llegar a una producción autosuficiente que logre abastecer las necesidades de la población. El principal problema es el uso de la agricultura clásica, una técnica que desafortunadamente depende de los factores ambientales, el clima y los tipos de suelo. Por ello, se han buscado alternativas y nuevas técnicas que permitan una producción más eficiente que logre resolver los problemas de abasto principalmente en países pobres o densamente poblados.

La agricultura tradicional es una técnica dependiente del clima y tipo de suelo. Esta técnica está basada en las estaciones del año. Los agricultores tienen fechas específicas para sembrar y para cosechar, las fechas son definidas según el inicio de la temporada de lluvias y la temperatura. Si la temporada de lluvias no es buena, o la temperatura cambia drásticamente, se corre el riesgo de perder la producción, ocasionando pérdidas económicas y un desabasto de alimentos.

Los invernaderos se han convertido en una solución a las limitantes de la agricultura clásica. Consiente en una estructura sólida cerrada cubierta de algún material traslucido, principalmente plástico, que permite tener un control al interior, de parámetros como la temperatura, la humedad, la fuerza del viento, la iluminación, etc. Al poder manipular los parámetros al interior del invernadero es posible pasar de una agricultura temporal a una agricultura permanente, sin importar las condiciones meteorológicas al exterior, logrando producir alimentos durante todo el año **[IV, V, VI]**.

Existen diferentes maneras de clasificar los tipos de invernaderos, se pueden clasificar según su estructura en **[IV, V, VI]**:

- **Invernaderos sencillos:** Son aquellos invernaderos hechos en casa con material fácil de conseguir, son económicos y en la mayoría de casos son temporales debido a la calidad de sus materiales y a que su elaboración está destinada a un pasatiempo.
- **Invernaderos permanentes:** Estos invernaderos suelen ser construidos por empresas especializadas, los materiales empleados y su diseño tienen como finalidad una larga vida y una manipulación precisa de los parámetros al interior del invernadero debido a que su fin es comercial.

Según su diseño estructural:

- **Planos o tipo Parral:** Su construcción se basa en paredes y un techo plano, similar a un prisma rectangular, su diseño es económico y fácil de implementar, pero tiene la desventaja de una mala circulación del aire en su interior. También, en zonas lluviosas se corre el riesgo de hundimiento del techo si el agua comienza a acumularse, poniendo en riesgo toda la estructura del invernadero, por lo que solo es recomendable para zonas áridas o de poca lluvia.

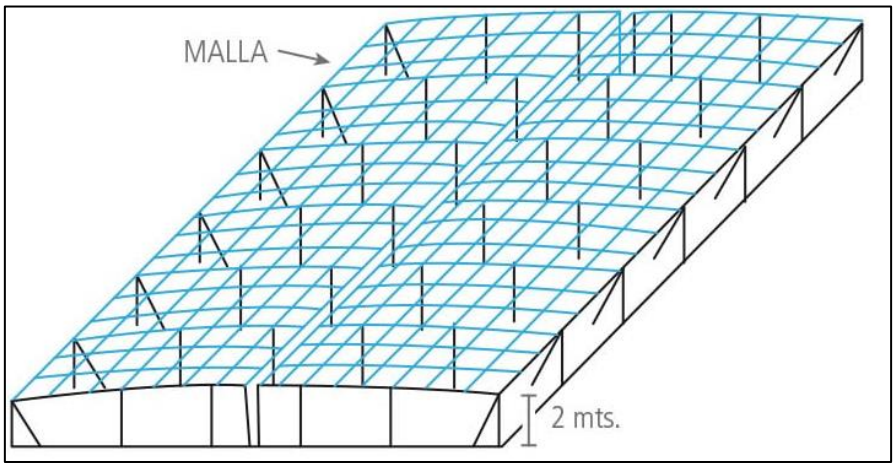


Figura 2.1.1: Invernadero tipo Parral. Tomado de "Principales Tipos de Invernaderos" por Horticultivos, https://www.horticultivos.com/featured/principales-tipos-invernaderos/

- **Tipo Raspa:** Es un invernadero de paredes planas, cuenta con una raspa en el techo, que evita la acumulación del agua de lluvia. Su implementación es barata y puede ser utilizado en zonas con lluvia abundante.

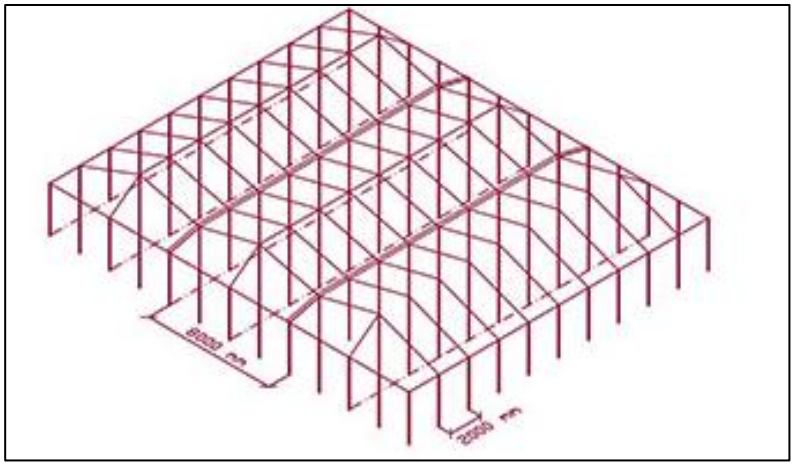


Figura 2.1.2. Invernadero tipo Raspa. Tomado de "Invernadero Raspa y Amagado" por NOVAGRIC, https://www.novagric.com/es/venta-invernaderos-novedades/tipos-de-invernaderos/invernaderos-raspa-amagado

- **Capilla y Doble Capilla**: Este invernadero cuenta con una estructura de paredes planas y una cubierta arcada, es ideal para climas fríos y con lluvia extrema, esto debido a que el arco evita la acumulación de agua y nieve en el techo del invernadero.

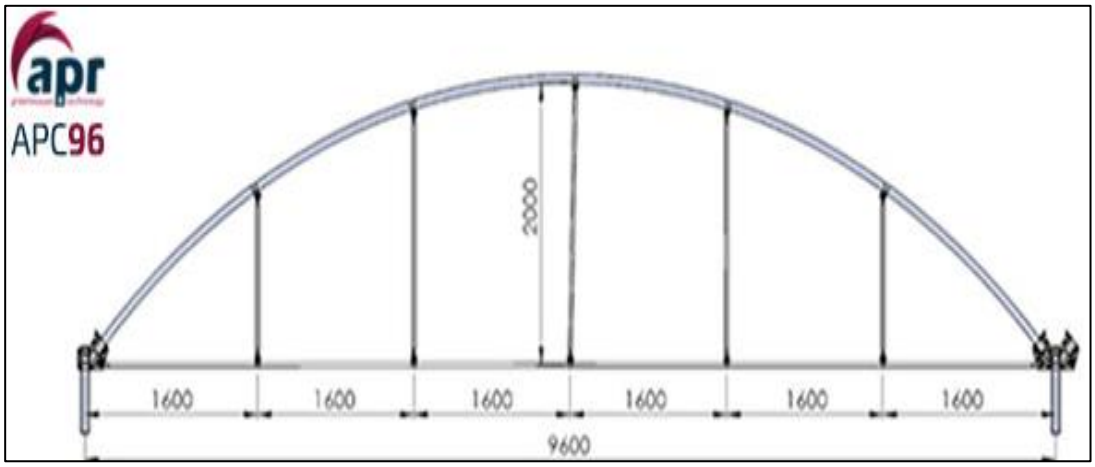


Figura 2.1.3. Invernadero tipo Capilla. Tomado de "Invernaderos Capilla" por NOVAGRIC, https://www.novagric.com/es/venta-invernaderos-novedades/tipos-de-invernaderos/invernadero-capilla

- **Tipo túnel o semicilíndrico:** Es el invernadero con más ventajas que desventajas, consiste en un semicírculo, por lo que sus paredes y su cubierta son curvas. Es ideal para climas extremos y su diseño permite una buena iluminación y una buena ventilación.

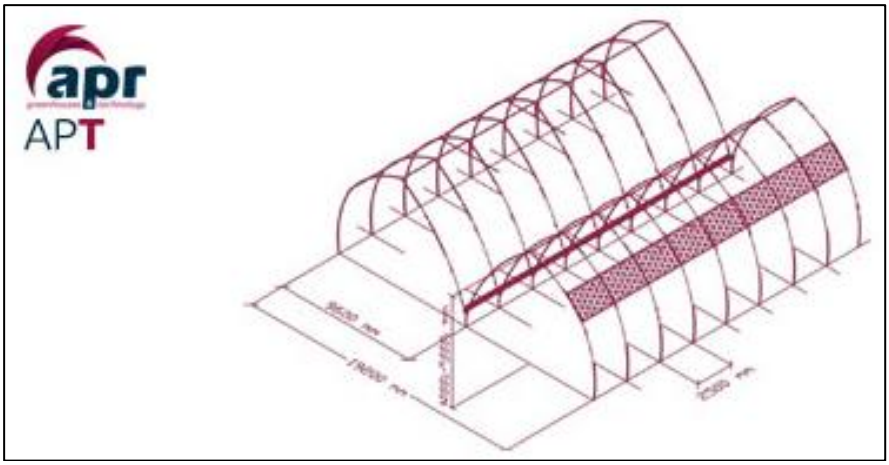


Figura 2.4. Invernadero tipo Túnel. Tomado de "Invernadero Túnel" por NOVAGRIC, https://www.novagric.com/es/venta-invernaderos-novedades/tipos-de-invernaderos/invernadero-tunel

Según el tipo de cultivo:

- **Enarenado:** Cultivo tradicional y más conocido. Consiste en la agricultura tradicional sobre tierra.
- **Aeropónico:** Es una de las técnicas de cultivo más reciente, en esta técnica no es necesaria la tierra o algún otro tipo de material de soporte para la planta debido a que la raíz se encuentra suspendida en el aire. La planta es colocada en los orificios de una caja oscura en donde las hojas se encuentran en el exterior y la raíz en el interior. Para alimentar a la planta, la caja cuenta con aspersores que son activados y rocían la raíz con agua preparada que lleva todos los nutrientes necesarios. Al estar en completa oscuridad, la raíz no sufre los daños causados en cultivos tradicionales debido a la radiación solar, además de que las algas y algunos tipos de enfermedades no logran desarrollarse debido a la falta de luz.
- **Hidropónico:** Para este proyecto, el invernadero con cultivo hidropónico será el más importante. Es una técnica antigua, pero debido a sus exigencias no se había empleado con frecuencia hasta hace unas décadas. La hidroponía consiste en la sustitución del suelo por agua, o algún sustituto sólido, como medio de soporte para la planta en desarrollo. Según las primeras investigaciones científicas en este tema, se llegó a la conclusión de que el crecimiento exitoso de una planta no dependía de su medio de soporte, sino de los nutrientes que el medio poseía, por lo cual es posible eliminar el suelo y proporcionar a la planta los nutrientes necesarios a través de otros medios que más adelante se mencionaran.

2.2 Técnicas de hidroponía

La hidroponía es una de las técnicas más complicadas y productivas que existen en la agricultura. Es complicada porque se necesitan parámetros y condiciones muy precisas para lograr un buen desarrollo de las plantas, y es una de las técnicas más productivas debido a que en poco espacio es posible lograr una cantidad de cosecha mucho mayor que en la agricultura tradicional **[1, 2]**.

Las condiciones que se deben cumplir para una buena producción agrícola hidropónica están ligadas a la eliminación del suelo como medio de soporte para la planta. Como ya se mencionó, la tierra en el suelo

cumple dos funciones en el cultivo tradicional, una es darle soporte al cuerpo de una planta para que pueda crecer libremente y la otra razón, la más importante, es alimentar a la planta a través de los minerales que se encuentran en la tierra y que son absorbidos por la raíz. De acuerdo a lo anterior, los investigadores concluyeron que era posible un buen desarrollo de una planta si se sustituía el suelo por alguna otra fuente de nutrientes y si se le proporcionaba a la planta algún otro medio de soporte artificial, y así fue como nació la hidroponía.

En las investigaciones relacionadas a la hidroponía fue necesario un estudio a profundidad sobre los componentes del suelo para determinar qué minerales son proporcionados a una planta y se descubrieron dos grupos importantes, el grupo de los macroelementos, que son aquellos elementos químicos que se encuentran presentes en grandes cantidades y que son de vital importancia para un correcto crecimiento, en este grupo se encuentran el Carbono, el Calcio, el Hidrogeno, el Oxígeno, el Nitrógeno, el Fosforo, el Potasio, el Magnesio y el Azufre. Al otro grupo pertenecen los microelementos que, aunque se encuentran presentes en pequeñas cantidades, su ausencia significaría un mal desarrollo en las plantas, un fruto de poca calidad e inclusive la muerte. A este grupo pertenecen el Hierro, el Manganeso, el Zinc, el Cobre, el Molibdeno y el Boro. Con esta información fue posible buscar fuentes adecuadas de nutrientes para lograr alimentar un cultivo sin suelo. A esta fuente alternativa de nutrientes se le llama solución nutritiva y la manera más común de encontrarlo es en una mezcla liquida de agua y sales que contienen los elementos necesarios para sustituir al suelo como fuente de nutrición **[1, 2]**.

Son tres las técnicas que se han empleado en la hidroponía, las cuales se describen a continuación **[1, 2, 4, I, VII]**

1° NFT o Técnica del Flujo Laminar de Nutriente:

Consiste en una serie de canales hechos con tubos de PVC, estos canales contienen orificios los cuales permitirán que la planta salga a la superficie conforme va creciendo.

La técnica es llamada “del flujo laminar” debido a que una ligera capa (o película) de la solución nutritiva circula por los canales logrando humedecer a las raíces de las plantas, las cuales se encuentran al interior de los tubos protegidas de la luz solar que ocasiona algas y bacterias. En esta técnica es importante la circulación de la solución nutritiva a lo largo de los canales debido a que su movimiento permite la oxigenación, que es un factor importante para que una planta se mantenga con vida. En pocas palabras, la circulación consiste en vaciar la solución nutritiva por un extremo del canal, al cual se le da una ligera pendiente para que la gravedad haga que la solución nutritiva pueda correr fácilmente y salga por un desagüe colocado en el otro extremo del canal. Al salir, la solución nutritiva es recolectada y se hace circular nuevamente por el canal. El proceso se repite por varios minutos para lograr una óptima oxigenación y al final se deja una ligera capa de solución nutritiva, lo suficientemente alta para cubrir las raíces del cultivo. La circulación de solución nutritiva en los sistemas NTF puede ser realizada manual o automáticamente, y el número de veces en las que se hace la circulación de la solución nutritiva depende de la temperatura del lugar y la etapa del desarrollo de las plantas. Para un cultivo con diferentes tipos de plantas se suele recomendar una circulación por hora con una duración de 5 minutos.

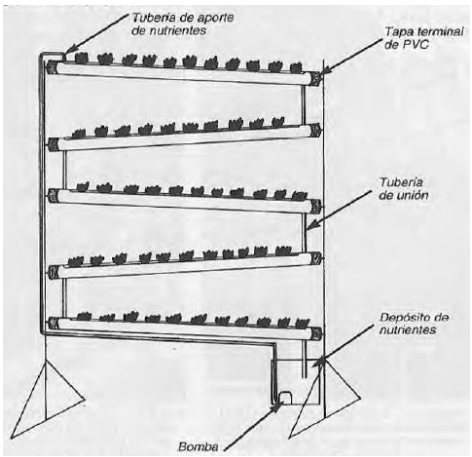


Figura 2.2.1: Diagrama NFT. Nota. Adaptado de "Cultivos Hidropónicos" (p. 174), por J.M. Resh, 2002, Ediciones Mundi-Prensa

2° Raíz flotante:

Esta técnica no suele ser tan elaborada como la primera debido a que la cantidad de material empleado no es tan grande. En este caso, además de la solución nutritiva, solo se necesita un contenedor de agua, el cual este cubierto en su totalidad para evitar la entrada de luz, un unicel con la misma área que el contenedor de agua, debido a que funcionara como tapa del contenedor, y esponja.

Aquí el procedimiento es sencillo, primero se verifica que el unicel entre correctamente al contenedor y que pueda subir y bajar sin dificultad, después se le hacen algunos orificios al unicel, en este caso el tamaño y separación los orificios dependerá del tipo de cosecha que se desea. En los orificios se debe introducir la esponja, que fungirá como medio de soporte para cada uno de los brotes de la planta que entraran a presión en los orificios, la suavidad de la esponja permitirá que la planta crezca libremente. Para finalizar, el contenedor se llena de solución nutritiva y se coloca el unicel sobre la solución. Las raíces se cubrirán con la solución nutritiva y se irán alimentando, por lo que la solución se ira terminando y el unicel que flota sobre la solución ira bajando al fondo del contenedor.

En esta técnica, al igual que en NFT, es necesaria la oxigenación de la solución. Para esta técnica existen dos formas de darle oxígeno a la solución. La primera es manualmente quitando el unicel del contenedor y agitando con las manos la solución por varios minutos. La segunda opción, y la más recomendable, es introducir una manguera conectada a una bomba de aire, la cual producirá burbujas al interior de la solución logrando oxigenarla. Esta bomba puede estar programada para activarse y desactivarse según las necesidades del cultivo.

Figura 2.2.2: Siembra hidropónica "Raíz Flotante". Nota. Adaptado de "Hidroponia Simplificada. Cartilla de Capacitación" (p. 36), por Oficina Regional de la FAO para América Latina y el Caribe, 2003, FAO

3° En sustrato:

El sustrato es el medio de soporte para una planta, en el caso de la agricultura tradicional el sustrato es la tierra en el suelo. Para la hidroponía, al eliminar el suelo como sustrato, se suele utilizar aquel material que cumpla ciertas condiciones, algunas de ellas son, que no sea toxico tanto para la planta como para el ser humano, que tenga buena retención de humedad, que sea económico, que sea lavable, que no se desintegre, etc. Algunos de los materiales que cumplen estas condiciones y que son muy populares en este tipo de cultivo son el tezontle, la agrolita, la cascara de arroz, la vermiculita, entre otros.

Este tipo de cultivo fue diseñado para plantas de gran tamaño y que necesitan de un buen soporte para su crecimiento. El diseño es muy similar al tradicional, con la diferencia de que en este caso el sustrato es colocado en algún recipiente o material que evite la filtración de la solución nutritiva. La planta se siembra en el sustrato, similar a una en tierra en un cultivo tradicional, y se riega con la solución nutritiva. A diferencia del cultivo tradicional, en este caso la solución se quedará retenida una parte en el sustrato y la otra en el recipiente que contiene al sustrato, por lo que la humedad durará más tiempo que en el cultivo tradicional logrando disminuir la cantidad de agua utilizada.

Figura 2.2.3: Siembra en Sustrato Nota. Adaptado de "Hidroponía Simplificada. Cartilla de Capacitación" (p. 37), por Oficina Regional de la FAO para América Latina y el Caribe, 2003, FAO

Dentro de esta técnica se encuentran las muy conocidas “Camas Capilares”, que fueron diseñadas para lograr cualquier tipo de cultivo, pero son mayormente recomendadas para cultivos subterráneos, como la papa y la zanahoria, que no pueden realizarse en las dos técnicas mencionadas anteriormente debido a que el exceso de agua provocaría la muerte de los frutos. Esta técnica también fue diseñada con el propósito de alargar el tiempo entre los riegos y ahorrar agua, ya que su diseño le permite almacenar grandes cantidades de agua, y ser consumidas según las necesidades.

Las camas hidropónicas son recipientes impermeables que cuentan dos capas de materiales. La primera capa consiste en graba o tezontle que funciona como depósito de agua en la parte inferior del recipiente, la graba es utilizada debido a que su porosidad permite una buena retención de a humedad y evita que el agua se mezcle con la tierra. La segunda capa consta de tierra de cultivo, que es el lugar en el que se realiza la siembra. Ambas capas se separan por una malla que impide el paso de la tierra a la capa inferior, a la vez que permite que el agua sea absorbida por las raíces de las plantas. El riego de la cama capilar se realiza a través de un tubo de PVC enterrado en la capa inferior que cuenta con un extremo vertical que

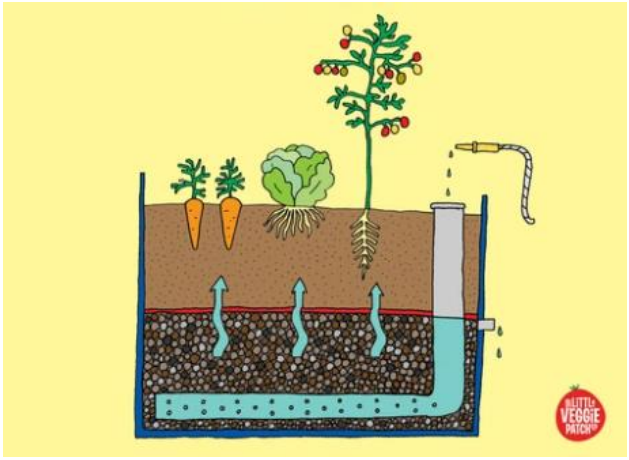


Figura 2.2.4. Cama de Cultivo Capilar. Tomado de “How to Build a Wicking Bed” por The Little Veggie Patch, https://littleveggiepatchco.com.au/blogs/news/building-a-wicking-bed

sale a la superficie, y es por donde entra el agua. Al llenar el tubo con agua, una parte será absorbida por la capa de graba, mientras el resto permanecerá en el tubo hasta que las plantas la requieran.

2.3 Ventajas e inconvenientes de cultivos hidropónicos

El éxito de los cultivos hidropónicos radica en que las ventajas son mayores comparadas con los inconvenientes o desventajas. La primera ventaja, que ya se mencionó es que los sistemas hidropónicos no dependen de tipo de suelo, y en caso de un cultivo en invernadero, también el tipo de clima no tiene relevancia. Por lo que fácilmente se puede utilizar en cualquier región del mundo, inclusive en un barco en altamar, permitiendo llevar alimentos a lugares pobres o de difícil acceso. También, otra de sus principales ventajas es la cantidad de producción en áreas pequeñas debido a que la mayoría de sus sistemas suelen ser verticales y tan altos como se desee.

Descritas ya las dos principales ventajas, a continuación, se enlista una serie de ventajas adicionales:

- Reducción de los costos de producción.
- La siembra y cosecha se puede dar en cualquier momento del año debido a que no depende del clima de la región.
- Ahorro en el consumo de agua.
- Cosecha de mejor calidad.
- Acorta el tiempo de cosecha.
- Evita la destrucción y corrosión de los suelos.
- Acorta el tiempo de cosecha.
- Mejora el medio ambiente.
- Se convierte en una opción de ingresos para personas de escasos recursos.

Para el caso de los inconvenientes, el principal es el gasto inicial debido a que es necesaria la compra del material para la infraestructura del sistema hidropónico, los químicos para la realización de la solución nutritiva, el material para el invernadero, los sensores, cables, y todo lo relacionado con la automatización del sistema, etc. También, otra desventaja es el monitoreo permanente de los parámetros para la obtención de una buena producción debido a que si alguno falla se corre el riesgo de perder la cosecha, además de que es necesario un mantenimiento periódico del sistema para garantizar una buena limpieza y de esta manera evitar la contaminación de la solución nutritiva, la cual es bastante sensible y los cambios en sus propiedades pueden alterar la producción.

2.4 Monitoreo y control de un invernadero hidropónico como una aplicación IoT

Anteriormente se mencionó que el propósito de implementar un sistema IoT en un cultivo hidropónico es mantener un monitoreo autónomo de los parámetros del entorno del cultivo. También, se mencionó la arquitectura y los puntos importantes de cualquier sistema basado en el Internet de las Cosas, que serán tomados como referencia para el diseño del sistema hidropónico.

El primer paso consiste en desplegar un grupo de sensores, al menos uno por cada parámetro que se desea monitorear. Los parámetros más comunes suelen ser la temperatura, la humedad relativa, la iluminación y el nivel de pH **[2]**. Los sensores son los encargados de convertir fenómenos físicos en algún tipo de energía, principalmente eléctrica o magnética, las cuales son transformadas en datos binarios para ser utilizados en los procesos de automatización **[14]**.

La información recolectada por los sensores debe ser enviada para ser almacenada en una memoria o base de datos en espera de ser utilizada. Puede ser empleado cualquier tipo de medio de transmisión siempre y cuando garantice una entrega fiable a los dispositivos que la soliciten. En este punto es donde normalmente entra el Protocolo de Internet (IP) debido a que es muy común que la información sea almacenada en bases de datos en la nube.

Un sistema hidropónico IoT debe contar con un cerebro que lleve el control de los procesos y decisiones, en este caso el cerebro será una tarjeta Raspberry Pi. La cual puede ser el centro de control de los sensores utilizados. A ella llega la información captada por los sensores, la toma para utilizarla y al mismo tiempo la envía a una base de datos para que se encuentre disponible en caso de que algún usuario desee consultar el estado los parámetros del sistema.

El siguiente paso consiste en convertir los datos obtenidos por los sensores en información útil. En este caso, cuando la Raspberry obtiene datos de los sensores los envía al programa previamente diseñado y ejecutado, el cual hace uso de las memorias, el procesador y demás elementos de la Raspberry para procesar la información de entrada y decidir si cumple o no los parámetros establecidos en el diseño del programa. Dependiendo del veredicto final del sistema, es posible que la Raspberry habilite sus pines de salida para activar o desactivar algún tipo de actuador conectado en ella. El objetivo de un actuador es que mediante una señal eléctrica es capaz de cambiar el estado de un dispositivo electromecánico. Por ejemplo, si en algún momento el sistema programado determina que la temperatura dentro de un invernadero es más alta que la definida en el diseño, enviará una señal a un actuador, y al recibir dicha señal, el actuador activará la ventilación del invernadero hasta que la temperatura descienda al valor definido previamente.

El ejemplo anterior puede ser aplicado a cualquier otra variable que se desee monitorear y controlar en un sistema IoT, en este caso un cultivo hidropónico. La información obtenida por los sensores funge como datos de entrada para el programa del sistema, el cual analiza la información y según su decisión, activa o desactiva a los actuadores que, a su vez, habilitan o deshabilitan los dispositivos electromecánicos que controlan el entorno y el valor de sus parámetros.

Todos estos datos deben ser enrutados por alguna red para que el sistema pueda ser llamado IoT, pero además de los datos de los sensores, también es posible enrutar los dispositivos electromecánicos para que su estado pueda ser modificado a distancia. Por ejemplo, los ventiladores podrían ser activados remotamente con algún dispositivo con conexión a Internet.

Lo anterior es una explicación resumida de la manera en que el Internet de las Cosas puede ser implementado a un sistema hidropónico. Mas adelante, se explicará detalladamente el proceso.

Capítulo 3: Parámetros a observar en un cultivo hidropónico

A continuación, se presenta una breve descripción de los principales parámetros a monitorear en un invernadero con cultivos hidropónicos.

3.1 Temperatura y humedad ambiente

En un cultivo en general, no solo hidropónico, la temperatura y la humedad ambiente son dos medidas relacionadas con el entorno. La temperatura ambiental se define como la temperatura, energía calorífica, registrada en un lugar en un momento determinado. Mientras que la humedad ambiente se define como la cantidad de vapor de agua. Ambas medidas están relacionadas entre sí, y su medición exacta debe ser realizada con sensores debido a que pueden ser engañosas para la percepción humana. La cantidad de vapor de agua en el aire suele determinar la sensación térmica, que es una reacción del cuerpo que nos hace sentir más frio o más calor que el registrado en un termómetro. Cuando se tiene la misma cantidad de humedad, pero se aumenta la temperatura la sensación térmica hace que el cuerpo humano sienta más calor, por el contrario, cuando se mantiene constante la temperatura y se comienza a aumentar la humedad es común sentir más frio. Debido a esta falsa sensación es importante contar con sensores y termómetros que permitan conocer la temperatura y la humedad ambiente reales para poder tomar buenas decisiones que permitan un funcionamiento óptimo del cultivo hidropónico **[1, 2, 4]**.

3.2 Humedad del terreno

En el caso de la humedad ambiental, las mediciones se realizan en el aire, pero para el caso de la humedad del terreno las mediciones son en el suelo. La humedad de terreno se define como la cantidad de agua atrapada en el suelo **[1, 2, 4]**.

La humedad del terreno es otro de los parámetros importantes que se debe tener en cuenta al momento de diseñar un cultivo debido a que diferentes tipos de suelo absorberán diferentes cantidades de agua, que al evaporarse se mezclará con el aire influyendo en la humedad del ambiente.

3.3 Luminosidad

Es importante recordar que algunas plantas necesitan luz para poder realizar la fotosíntesis, que es la manera en la que se alimentan las plantas. Por lo que es necesario, en caso de que el tipo de cultivo lo necesite, tener un sistema que aproveche al máximo la luz solar o proporcionarle al cultivo iluminación artificial. En la actualidad existe información relacionada al diseño de la estructura, el material utilizado y la orientación del invernadero para aprovechar al máximo la luz solar, además de técnicas que permiten atraer una mayor cantidad de luz al invernadero **[1, 2, 4, VIII, IX]**.

3.4 PH en el agua de riego

La medición del nivel de pH corresponde a la medición de hidrogeno en una sustancia y el conocerla es importante debido a que con ello se puede determinar si una sustancia es acida (del nivel 0 al 6), neutra (nivel 7) o base (del nivel 8 al 14). Para el caso de la hidroponía, la medición se realiza en la solución nutritiva, ya que, dependiendo de su nivel de pH las raíces del cultivo podrán o no absorber los nutrientes adecuadamente **[1, 2, 3]**.

3.5 Nivel de agua en depósito de suministro

El nivel de agua adecuado en un sistema hidropónico depende de cuál de las tres técnicas está siendo empleada. En este proyecto se han decidido utilizar las técnicas NFT y en Sustrato, por lo cual se procede a hablar de los niveles de agua para esta técnica **[1]**.

- **NFT:** En esta técnica los canales proporcionan el soporte para las plantas y el medio por el cual correrá la sustancia nutritiva. Los depósitos de suministros de solución nutritiva serán dos, los canales, en los que la solución nutritiva debe de ser lo suficientemente alta como para tocar las raíces de la planta, y a la vez procurando dejar una capa de aire entre la superficie de la solución nutritiva y la parte superior del tubo de PVC, de esta manera, la capa de aire proporcionará oxigeno adicional a las raíces. El otro depósito de suministro corresponde al recipiente que almacenará la solución nutritiva que saldrá por los desagües de los canales para ser utilizada nuevamente generando la recirculación. Es importante mencionar que el recipiente que actuará como depósito de suministro estará cubierto totalmente para evitar el paso de la luz solar que facilita la creación de algas en lugares húmedos. La solución nutritiva es llevaba desde el almacén de agua hasta los canales a través de una bomba **[1, 2]**.

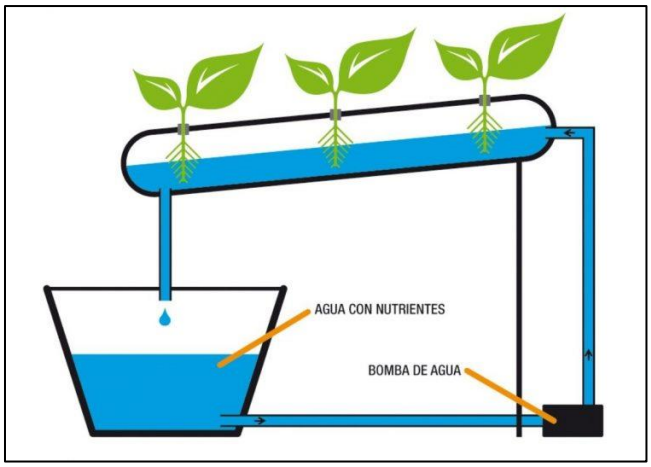


Figura 3.5.1: Niveles de agua para NFT. Tomado de "NFT" por Hidroponía24, https://hidroponia24.com/sistemas-hidroponicos/nft/

En Sustrato: El nivel de agua depende del tipo de sustrato y el tipo de cultivo. En especial, para las camas de cultivo capilares la cantidad de solución nutritiva dependerá de las dimensiones de la cama, el largo y ancho de la cama son opcionales, pero la altura del depósito de agua no deberá pasar los 20 cm de alto. El depósito de las camas capilares no está totalmente lleno de agua, porque como ya se mencionó, en esta

capa se agrega otro tipo de material que ayude a la retención de la humedad. Además del depósito de agua, también debe haber solución nutritiva en la tubería de PVC que suministra el riego, y es donde se observa si el cultivo aun cuenta con solución nutritiva **[1, 2]**.

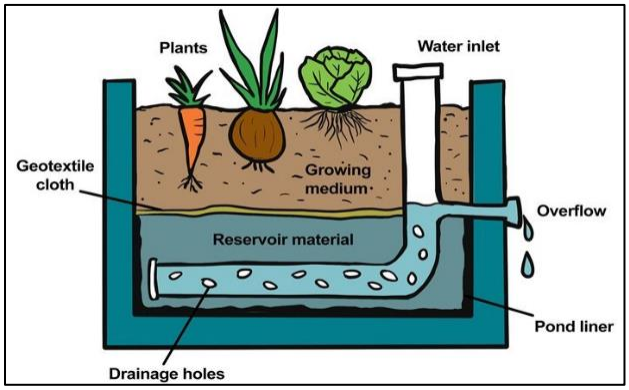


Figura 3.5.2: Niveles de agua para Camas de Cultivo Hidropónico. Tomado de "Building a Wicking Bed" por The Royal Botanic Garden, https://www.rbgsyd.nsw.gov.au/Learn/Living-learning/Gardening-at-Home/Building-a-Wicking-Bed

3.6 Sistema de riego y recirculación de agua

Los sistemas de riego son parte fundamental de un cultivo hidropónico ya que serán los responsables de llevar la solución nutritiva por todo el cultivo. Un mal diseño de estos sistemas podría ocasionar que la solución no llegue correctamente a todas las plantas, ocasionando un crecimiento desigual en el cultivo, e incluso la muerte de algunas plantas si la solución nutritiva es muy poca o nula **[1, 2]**.

En este caso, el tipo de sistema de riego también depende de la técnica de cultivo empleada, y la recirculación de agua solo es posible en NTF. A continuación, se describen los sistemas de riego para las técnicas que se van a emplear.

- **NTF:** De nuevo se menciona que los canales hechos con tubo de PVC están conectados entre ellos para logran que un solo flujo recorra todo el cultivo. Los canales están conectados en zigzag, cada canal con una pequeña inclinación y un canal sobre otro creando un sistema vertical de pocos metros de alto. Debajo de los canales, preferentemente al nivel del piso, se encuentra el recipiente que sirve como depósito de solución nutritiva. Conectada al depósito, se encuentra una bomba que sube la solución al canal más alto y la solución comienza a bajar por todos los canales logrando la oxigenación de las raíces. En el canal más bajo del sistema se encuentra un desagüe que se conecta al depósito para que la solución nutritiva llegue al mismo punto donde salió y de nuevo vuelva a subir creando una recirculación en el sistema **[1, 2]**.

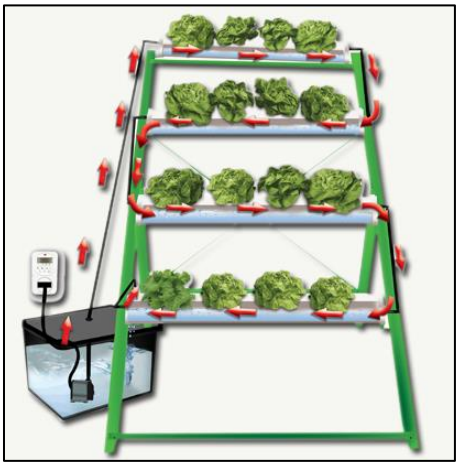


Figura 3.6.1: Recirculación en NFT. Tomado de "Cultivos alternativos" por Hidroponía, https://hidroponiablog.wordpress.com/

- **En sustrato:** Para esta técnica no es posible la recirculación. En cuanto al riego, se utiliza un tubo de PVC en forma de L, la parte horizontal es perforada con pequeños orificios para permitir la salida de agua, y enterrada en la capa de grava en el suministro de agua. La parte vertical sale a la superficie atravesando las capas de grava y la de tierra. El agua, o la solución nutritiva, es introducida por el orificio de la parte vertical que sale por la superficie. En caso de agregar demasiada solución nutritiva existe la posibilidad de que se inunde la capa de tierra, lo cual arruinaría el cultivo, para evitar esto se agrega un drenaje entre la capa de graba y tierra, que evitara inundaciones **[1, 2]**.

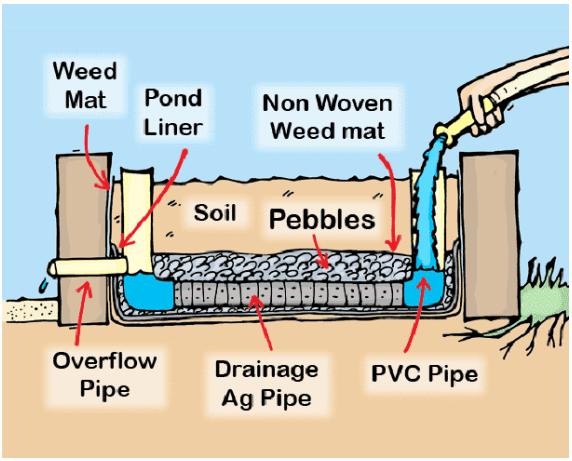


Figura 3.6.2: Sistema de Riego de una Cama de Cultivo Capilar. Tomado de "Wicking Beds" por Local Food Connect, https://localfoodconnect.org.au/community-gardening/wicking-beds/

3.8 Sistema de ventilación

El principio básico de un invernadero es el permitir la entrada de radiación solar, la cual, al tratar de escapar comienza a rebotar con las paredes interiores del invernadero provocando un calentamiento del aire al interior. Esta es la razón por la que el interior de un invernadero tiene una temperatura mayor que el exterior. En días calurosos o cuando un invernadero lleva mucho tiempo cerrado, es posible que las temperaturas al interior sean extremadamente altas debido al calor acumulado, por eso es importante contar con un sistema de ventilación que permita renovar el aire dentro del invernadero para tener una temperatura ideal **[IV, V, VI]**.

La técnica más utilizada es la ventilación natural, consiste abrir orificios en partes estratégicas del invernadero para lograr un flujo de aire al interior. Para una ventilación natural se abren orificios en la parte alta y en la parte baja de las paredes del invernadero. Las partes altas permiten la salida del aire caliente, mientras que por las partes bajas entra aire fresco del exterior. Si se desea controlar la cantidad de aire que entra y sale se busca la manera de ajustar el tamaño de los orificios o de cerrarlos completamente si no se desea ventilar el cultivo. En los últimos años se ha venido modernizando la técnica de ventilación natural y se ha llegado a una ventilación mecánica, que es bastante similar a la natural. La diferencia entre ambas técnicas es que en los orificios de la ventilación natural son colocados unos ventiladores que hacen que el flujo de aire este mejor controlado. Los ventiladores estarán colocados de tal manera que en la parte baja del invernadero permitan la entrada de aire, y en la parte alta permitan la salida, similar a la ventilación natural **[IV, V, VI]**.

Estas dos técnicas son las más empleadas en los cultivos comerciales, en ambas se ha evitado hablar de las dimensiones de los orificios o el tamaño de los ventiladores debido a que esto depende de las dimensiones del invernadero, pero se define que el tamaño de la ventilación debe ser de aproximadamente el 10% del área de la pared donde es colocada.

Capítulo 4: Análisis de los parámetros de un cultivo en hidropónico

4.1 Límites dentro de los cuales debe pueden variar los parámetros de un cultivo

En la agricultura, los límites de variación de los parámetros suelen ser diferentes, esto debido a que cada especie de cultivo tendrá sus propios parámetros. La idea del proyecto es implementar un invernadero hidropónico con diferentes especies, por lo que los parámetros a utilizar serán los definidos en los libros como “globales”, con la finalidad de poder producir la mayor cantidad de especies dentro de un mismo invernadero.

En los libros se presentan los siguientes valores, que son los valores promedio en los cuales una buena cantidad de especies de plantas se pueden desarrollar adecuadamente **[1, 2, I]**:

Parámetro:	**Rango global:**
pH	5.5 – 6.5
Temperatura	10 – 25 [°C]
Humedad ambiental	50 – 75 [%]
Conductividad Eléctrica	1 – 3 [mS/cm]

Estos rangos tienen que ser acompañados de mínimo 6 horas de luz solar y vientos que no dañen las plantas. Todos estos factores y parámetros permitirán un cultivo variable **[2]**.

A continuación, se presenta una tabla de diferentes especies de plantas y hortalizas que se encuentran dentro de los rangos de los parámetros de la tabla anterior. Estas especies serán candidatas a ser producidas dentro del sistema, el objetivo será producir la mayor cantidad de especies que aparecen en la tabla **[X]**.

Nombre	**Temperatura [°C]**	**Humedad [%]**	**pH**	**CE [mS/cm]**
Acelga	15 – 25	60 – 90	6.0 – 6.5	0 – 2.0
Ajo	10 – 20	70 – 75	5.0 – 7.5	1.9
Apio	9 – 25	60 – 80	6.0 – 7.2	1.8
Brócoli	6 – 30	60 – 75	4.3 – 8.0	2.8
Calabacita	10 – 30	65 – 80	4.5 – 8.2	1.2
Cebolla blanca	15 – 20	70 – 75	6.0 – 7.5	1.2
Cebolla cambray	7 – 30	70	6.0 – 6.8	1.2
Cebolla morada	7 – 24	70	6.0 – 7.0	1.2
Chilaca	18 – 32	60 – 75	5.5 – 6.8	2.5
Cilantro	10 – 30	75	5.5 – 7.0	1.5
Clavel	18 – 25	60 – 70	6.5 – 7.5	1.0 – 2.0
Col blanca	15 – 18	60 – 80	6.0 – 6.5	2.8
Col morada	15 – 18	60 – 80	6.0 – 6.5	2.8
Crisantemo	18 – 25	60 – 70	5.5 – 6.5	2.0
Epazote	15 – 22	75	5.2 – 8.3	1.5 – 2.0
Espinacas	13 – 20	60 – 90	6.0 – 7.5	2.0

Fresas	10 – 35	60 – 80	5.2 – 7.0	1.0 – 1.5
Gerberas	15 – 25	70 – 90	5.5 – 6.7	1.5
Gladiola	10 – 25	60 – 70	6.5 – 7.0	2.5
Hierbabuena	15 – 25	70 – 80	5.5 – 8.3	1.5
Jalapeño	16 – 32	50 – 70	6.5 – 7.5	2.5
Jitomate	12 – 30	65 – 70	5.0 – 7.0	2.0
Laurel	15 – 30	70	5.0 – 7.0	1.5
Lechuga	16 – 20	60 – 80	5.8 – 6.8	1.25
Manzanilla	15 – 23	70	6.0 – 6.8	1.3
Margaritas	15 – 25	65 – 70	4.5 – 7.5	1.5
Mejorana	12 – 26	60 – 80	5.5 – 7.5	1.6
Melón	10 – 35	70	6.0 – 8.0	1.7
Menta	15 – 30	70 – 80	6.5 – 7.0	1.5
Mirasol	14 – 25	50 – 75	5.5 – 6.8	2.5
Morrón	15 – 25	50 – 70	5.8 – 6.6	1.5
Papa	15 – 20	60 – 80	4.8 – 7.0	1.7
Pepino	10 – 32	60 – 90	5.5 – 6.8	2.0
Perejil	11 – 20	80	6.0 – 7.5	1.5
Rábanos	15 – 18	70	4.3 – 8.3	1.2
Romero	9 – 25	50 – 60	6.0 – 7.5	1.5
Rosas	17 – 25	70 – 75	6 – 7.5	2.0 – 2.2
Serrano	14 – 25	50 – 70	6.0 – 6.5	2.5
Tomate	15 – 30	65 – 80	5.9 – 7.0	2.0
Zanahoria	16 – 21	70 – 80	6.0 – 7.5	1.0

4.2 Procesamiento de los parámetros

Para realizar un procesamiento de un parámetro es necesario obtener información de dicho parámetro. Las mediciones son tomadas por sensores y enviadas a un microcontrolador, en este caso al de una Raspberry. Los microcontroladores se definen como dispositivos electrónicos que ejecutan tareas específicas programadas en una memoria. La tarea que cada microcontrolador realiza se basa en la información que recibe.

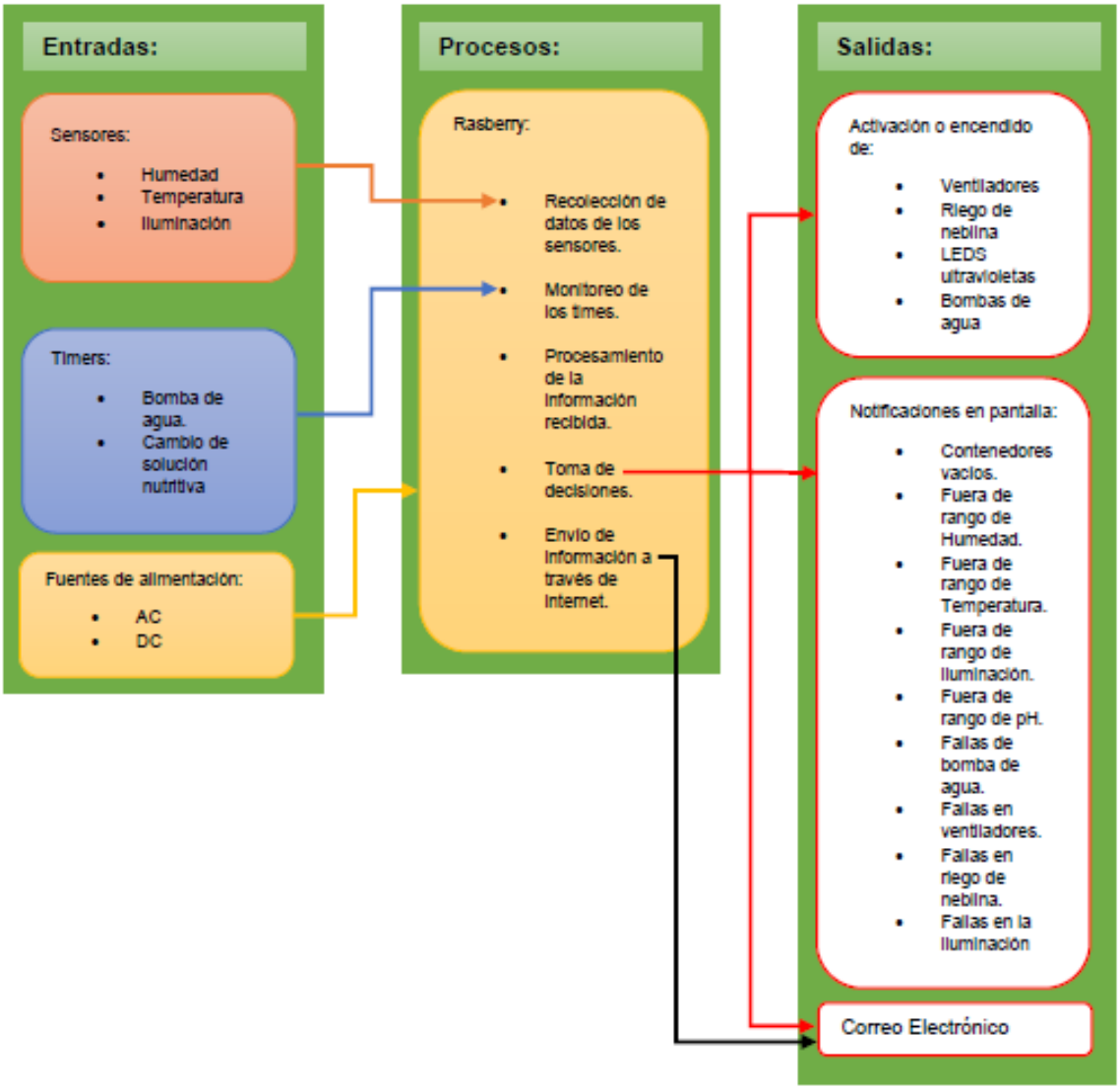
La Raspberry Pi 3 tendrá almacenado un código programado en Python, en el código se especificará cada uno de los parámetros y los niveles dentro de los cuales pueden variar. Se pretende que la Raspberry solicite a todos los sensores realizar una medición cada cierto tiempo, esta información es registrada por los pines de entrada de la tarjeta, y posteriormente son procesados convirtiéndolos en información útil para el sistema. El sistema que funciona gracias al programa creado en Python, será capaz de determinar si los parámetros cumplen las condiciones especificadas en el código fuente. La manera en que lo hace es comparando las entradas reales con las especificadas.

4.3 Niveles de alertas

Los niveles de alerta son aquellos valores que se encuentran fuera del rango de los parámetros establecidos al momento de diseñar el sistema. Cuando un sensor comience a medir valores fuera de este

rango la Raspberry será capaz de notificar al usuario que las condiciones no son óptimas, además de que se accionaran los actuadores necesarios para que los valores vuelvan a la normalidad. Una vez que el sistema comience a registrar valores dentro del rango los actuadores se apagarán.

Capítulo 5: Estructura física general del sistema de monitoreo y control



El diagrama de bloques anterior corresponde al diagrama del proyecto, consiste de tres bloques principales, y a continuación, serán explicados:

1° BLOQUE DE ENTRADAS:

En este bloque se encuentran todas las entradas al sistema como los datos de los sensores, los conteos de los timers y las fuentes de alimentación necesarios para que el sistema funcione y tome decisiones basadas en las condiciones. En este bloque se encuentra la primera capa de un sistema IoT, la Capa de Detección, formada por la red de sensores que capturaran la información

del entorno. También se encuentra la segunda capa, de Intercambio de Datos, que une tanto al Bloque de Entradas y al Bloque de Procesos, que en resumen envía los datos recibidos por los sensores a la Raspberry.

2° BLOQUE DE PROCESOS:

En este bloque se encuentra la parte más importante del sistema, la tarjeta Raspberry recibirá los datos capturados por los sensores, los procesará y convertirá en información útil para el programa del sistema, que igual se encuentra en este bloque. El programa estará diseñado para ser capaz de analizar la información recibida y procesada no solo de los sensores, también de los timers, y con base en ello tomar decisiones que afecten directamente al sistema, ofreciendo un servicio final al usuario del invernadero. En el Bloque de Procesos existen dos capas IoT, la Capa de Intercambio de Datos, que ya se explicó, y la Capa de Integración de la Información. La tercera capa, o de Integración de la Información, es la capa más importante para el funcionamiento del sistema, en ella se encuentra el código con las instrucciones y acciones a realizar para que el sistema siempre se encuentre funcionando en condiciones óptimas. Cuando el código se ejecuta y analiza las condiciones del entorno toma una decisión, la cual es enviada al bloque de salidas.

3° BLOQUE DE SALIDAS:

Al recibir las decisiones tomadas por el programa del sistema, en el bloque de salida se activarán o desactivaran los actuadores de los dispositivos electrónicos que se encargaran de mantener estables los parámetros del invernadero. Además, el usuario recibirá el servicio IoT contratado, es decir, la automatización de su invernadero hidropónico y una notificación vía correo electrónico sobre algún evento sobresaliente en el sistema, como alguna falla, fechas importantes en el cultivo, recordatorios, etc. Lo anterior es parte de la cuarta capa IoT, conocida como Capa de Servicio de Aplicación, que es la etapa final de un sistema IoT, donde las personas pueden interactuar con los servicios IoT.

Capítulo 6: Tecnologías a emplear

6.1 Rapsberry Pi

La tarjeta Raspberry es definida por sus creadores como una minicomputadora capaz de realizar las actividades básicas de una computadora normal además de proyectos de electrónica, programación y robótica. Fue desarrollada en 2012 en el Reino Unido por la fundación Raspberry Pi con el objetivo de acercar a los estudiantes a la programación a través de un mini ordenador de bajo costo **[10, 17]**.

La Raspberry cuenta con la mayoría de elementos de una computadora clásica, como un SoC, una memoria RAM, un reloj, módulos Bluetooth y Wifi, puertos de entrada y salida, puerto Ethernet, etc. Para su manipulación es necesario un monitor, un teclado y un mouse. Además, es necesario instalar un sistema operativo a través de una memoria Micro SD que se introduce a la tarjeta y debe permanecer todo el tiempo dentro debido a que funciona como memoria ROM de la Raspberry almacenando el sistema operativo, los programas, archivos y cualquier otra información no volátil que se necesite **[10, 17]**.

Debido al éxito de la tarjeta fue necesaria una actualización que cumpliera con las exigencias de las nuevas aplicaciones que se le encontraban por lo que Raspberry Pi sacó a la venta una versión llamada *B*, que se caracterizó por ser más eficiente que la primera versión, llamada *A*, además de contener más puertos y requerir más energía debido a sus nuevas funciones. Hasta la fecha se han lanzado cuatro generaciones de Raspberry, cada una con sus versiones A y B. En este proyecto se utilizará la Raspberry Pi 3 Versión B, por lo que a continuación se especifican sus características más importantes **[10, 17]**:

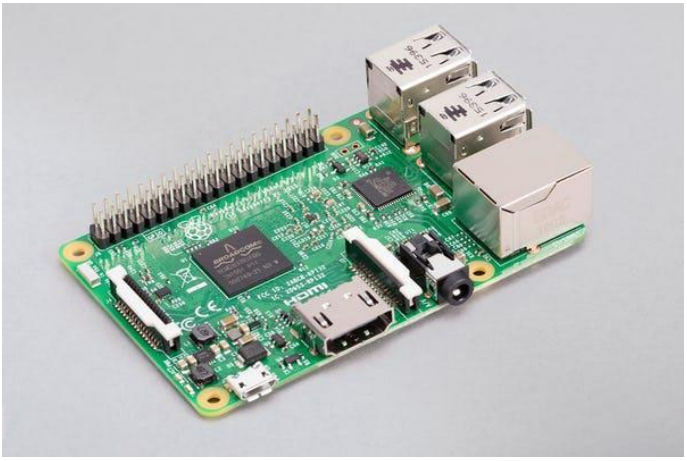


Figura 6.1.1: Raspberry Pi 3 Versión B. Tomado de "Raspberry Pi 3 Model B" por Local Raspberry Pi, https://www.raspberrypi.com/products/raspberry-pi-3-model-b/

Procesador:	Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
RAM:	1 GB
Módulos:	BCM43438 (LAN inalámbrica y Bluetooth)
GPU:	VideoCore IV Dual Core 400 MHz
Ethernet:	100BaseEthernet
Wifi:	Wifi 802.11 b/g/n

Bluetooth:	4.1
No. Pines GPIO:	40
No. Puertos USB:	4
Alimentación:	5 VCC, 2.5 A
Puertos:	• Estéreo • HDMI • CSI (Cámara) • DSI (Pantalla)

6.2 Conexiones de Raspberry Pi 3 con Sensores y Actuadores

El módulo GPIO (General Purpose Input/Output) es la parte de la placa Raspberry que funge como interfaz entre el software del sistema y el mundo real. Para la versión 3B el módulo consta de 40 pines que pueden ser configurados como entradas o salidas del sistema según las necesidades. En las entradas normalmente se conectan sensores o cualquier otro dispositivo capaz de recopilar información, mientras que en las salidas se conectan dispositivos electrónicos que reaccionan cuando se ejecuta el programa del sistema. Aunque la tarjeta cuenta con 40 pines GPIO, no todos pueden ser utilizados como pines de entrada y salida debido a que algunos de ellos están configurados como fuentes de voltaje de 3.3 [V] y 5 [V], y como tierra. Algunos otros pines están preconfigurados para ser utilizados con ciertos protocolos para generar señales o enviar y recibir información de tarjetas de expansión de pines, la diferencia entre los pines preconfigurados y los pines de fuentes de voltaje es que los pines preconfigurados si pueden ser utilizados como pines de entrada y salida **[10, 17]**.

A continuación, se presenta el mapa de los 40 pines GPIO de la tarjeta Raspberry Pi 3B:

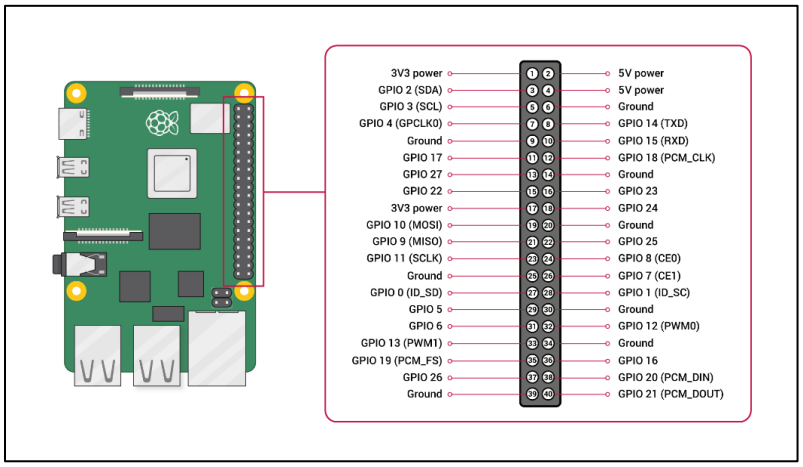


Figura 6.2.1: Mapa de pines GPIO. Tomado de "Raspberry Pi Documentation" por Raspberry Pi, https://www.raspberrypi.com/documentation/computers/os.html#gpio-and-the-40-pin-header

En la Figura 6.2.1 se observan los 40 pines, los 10 GPIO que funcionan como entradas o salidas, y los 30 restantes denominados pines especiales, los cuales están diseñados para diferentes funciones que serán explicadas brevemente, estas funciones están precargadas y necesitan ser configuradas completamente en caso de que se quieran utilizar **[10, 17]**.

- **Pines GROUND:** Están configurados para ser utilizados como tierra en los circuitos utilizados, es necesario conectar todas las tierras de los circuitos a los pines GPIO de tierra para crear una tierra común.
- **Pines 5V y 3V3:** Este tipo de pines entrega voltajes de 5 [V] y 3.3 [V] respectivamente, por lo que pueden ser utilizados para conectar dispositivos electrónicos que requieran voltajes de estas magnitudes, y así evitar el uso de baterías o fuentes de alimentación externas.
- **Pines seriales:** Son los pines Tx y Rx, su objetivo es comunicar a Raspberry con alguna otra tarjeta o con una expansión de la misma. Estos pines realizan la Transmisión y la Recepción de la información entre las tarjetas.
- **Pines PWM:** Estos pines entregan un tren de pulsos con ancho configurable.
- **Pines de interfaz:** El resto de los pines sirven para comunicar a la tarjeta con otros dispositivos que se comunican a través de interfaces especiales como UART, I2C y SPI.

Capítulo 7: Selección de sensores y sus características

Este capítulo tiene como objetivo introducir los sensores y actuadores que serán utilizados en el proyecto, por lo que a continuación se presentará una explicación a detalle de su funcionamiento y diseño, así como una tabla con las principales características de cada dispositivo.

7.1 Sensor de temperatura y humedad DHT22

El sensor DHT22 es un sensor dos en uno que mide la temperatura del aire con un termistor y la humedad ambiental con ayuda de un capacitor.

El termistor es un dispositivo electrónico capaz de detectar los cambios de temperatura a través del valor de su resistencia, la resistencia del termistor es inversamente proporcional al valor de la temperatura. Cuando aumenta la temperatura del aire la resistencia del termistor disminuye, mientras que, si la temperatura disminuye, la resistencia aumenta. El termistor es ampliamente utilizado en los sensores de temperatura debido a que es capaz de detectar variaciones mínimas en la temperatura y reflejarlas con cambios considerables en su resistencia **[XI, XII]**.

El capacitor utilizado para la medición de la humedad ambiental tiene un comportamiento similar al del termistor. Cuando el porcentaje de humedad cambia ocasiona un cambio en el valor de la capacitancia, logrando el funcionamiento del sensor **[XI, XII]**.

Ya explicado los principios de funcionamiento del sensor, a continuación, se presenta una fotografía del mismo:

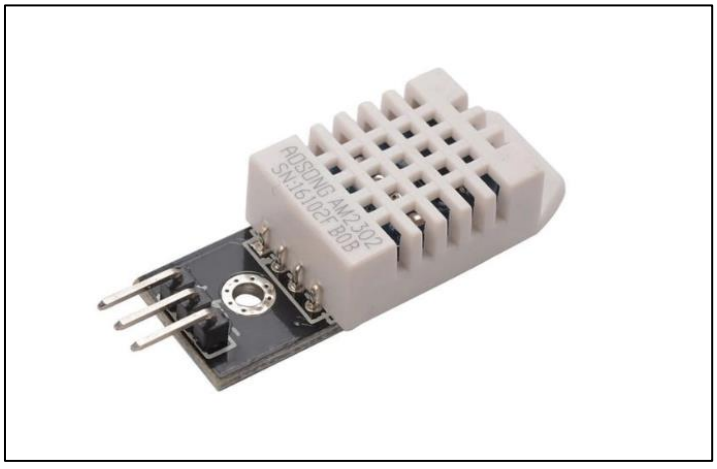


Figura 7.1.1: Sensor DHT22. Tomado de "DHT22: El sensor de temperatura y humedad de precisión " por Hardwarelibre, https://www.hwlibre.com/dht22/?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed%3A+hwlibreweb+%28Hardware+libre%29

En la imagen se puede observar el sensor cubierto con un protector color blanco, también puede ser azul. Cuenta con tres pines, dos para su alimentación y el tercer pin para enviar y recibir información del circuito con el que esté trabajando.

En la siguiente tabla se encuentran los datos de operación y las principales características de funcionamiento del sensor **[XI, XII]**:

Voltaje de alimentación:	3.3 – 6 [V]
Consumo de corriente:	2.5 [mA]
Tipo de señal:	Digital
Rango de temperatura:	-40 – 125 [°C]
Precisión de temperatura:	± 0.5 [°C]
Resolución de temperatura:	0.1 [°C]
Rango de humedad:	0 – 100 [%]
Precisión de humedad:	± 2 [%]
Resolución de humedad:	0.1 [%]
Frecuencia de muestreo:	2 [Hz]

7.2 Módulo con sensor de luz LDR

El módulo de detección de luz se basa en el funcionamiento de las resistencias LDR, *Light Dependent Resistor*. Estas resistencias son fabricadas de materiales semiconductores capaces de cambiar el valor de su resistencia dependiendo de la cantidad de luz que incide sobre ellas. Las resistencias LDR ofrecen una resistencia alta cuando la oscuridad es total, y la resistencia va disminuyendo con forme la cantidad de luz va aumentando hasta llegar a su valor más bajo cuando la luz solar incide sobre su superficie **[XIII, XIV]**.

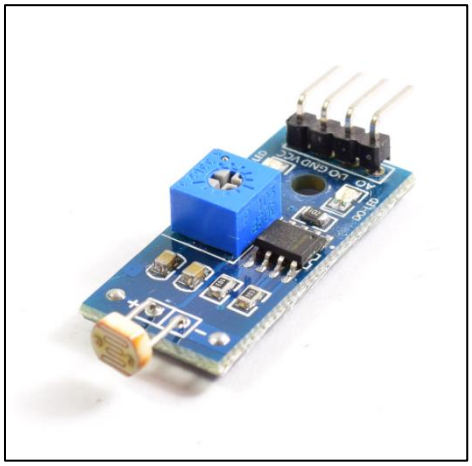


Figura 7.2.1: Modulo de Sensor de Luz. Tomado de "MODULO SENSOR LDR" por NayLamp Mechatronics, https://naylampmechatronics.com/sensores-luz-y-sonido/135-modulo-ldr.html

El módulo de sensor de luz que se utilizará en el proyecto cuenta con una resistencia LDR, también, cuenta con un circuito integrado programado para entregar datos analógicos y digitales. Para este módulo la salida será 0 cuando se exceda un valor umbral fijado a través del potenciómetro incluido en el módulo, y 1 cuando el valor leído por el sensor se encuentre por debajo del umbral **[XIII, XIV]**.

La salida digital es incluida en este módulo principalmente para ser conectada a algún relevador que active o desactive un componente electrónico.

A continuación, se presenta la tabla con las características técnicas para el funcionamiento del módulo **[XIII]**:

Voltaje de alimentación:	3.3 – 5 [V]
Tipo de señal:	Digital y Analógica
Rango de resistencia:	50 [Ω] – 1 [MΩ]
Valor de umbral:	Ajustable
Numero de pines:	4
Amplificador operacional:	LM393

7.3 Actuador de iluminación

En electrónica, un actuador es aquel dispositivo que logra accionar un mecanismo o circuito para realizar alguna tarea. Para el caso de la iluminación, el actuador más utilizado se basa en el relé, o relevador, que es un dispositivo electromecánico capaz de permitir o negar el paso de corriente en circuitos de bajo y alto voltaje **[10, 13, 14, 16]**.

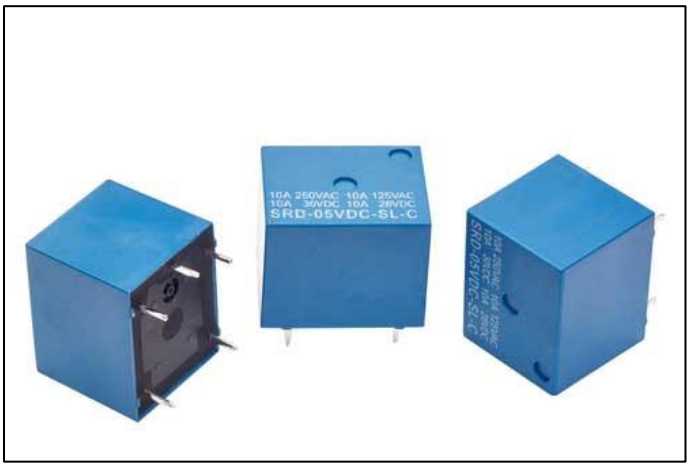


Figura 7.3.1: Relé. Tomado de "Relé con Arduino y ESP8266 para crear una lampara inteligente" por ProgramarFacil, https://programarfacil.com/blog/arduino-blog/rele-con-arduino-lampara/

Su funcionamiento es similar al de un interruptor, abre y cierra los circuitos permitiendo el paso de la corriente. Los relés se dividen en dos partes, la parte de control y la parte de interruptor. La parte de interruptor cuentan con tres placas, dos fijas y una móvil. Cuando el relé no tiene voltaje la placa móvil se encuentra en contacto con una de las placas fijas. Al aplicar voltaje se energiza un inductor que se encuentra dentro del relevador produciendo un campo magnético que atrae a la placa móvil hacia la otra placa fija, si al relevador se le deja de aplicar voltaje el electroimán se desactiva y la placa móvil regresa con la placa fija inicial. Este fenómeno es el que permite que los relés puedan cambiar de un estado a otro permitiendo o evitando el paso de la corriente en un circuito **[10, 13, 16]**. El voltaje que activa y desactiva el inductor es aplicado en la parte de control, y proviene de una señal de activación enviada por la Raspberry. Mientras que la corriente manipulada, a la cual se le permite o no el paso, se encuentra en la parte de interruptor y es una corriente diferente a la que circula por el inductor. La corriente controlada pertenece al circuito que se desea controlar.

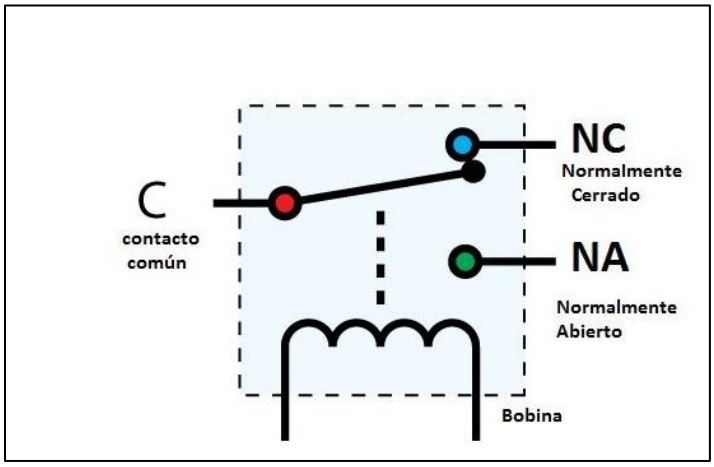


Figura 7.3.2: Componentes de un Relé. Tomado de "Relés" por ÁreaTecnología, https://www.areatecnologia.com/electricidad/rele.html

Debido a su diseño, los relevadores cuentan con inductores que funcionan con bajos voltajes controlando circuitos de bajo y alto voltaje. Un relé sencillo puede activar su electroimán con un voltaje de 5 [V] y controlar circuitos de hasta 220 [V], razón por la cual son ampliamente utilizados en la domótica y la industria.

Actualmente existen módulos con relés integrados que facilitan el armado de circuitos para control automático. Los módulos cuentan con un circuito impreso y los dispositivos electrónicos necesarios para que el relé funcione adecuadamente. Además, para el caso de la placa de Arduino, cuenta con seis pines repartidos tres en la parte de control y tres en la parte de interruptor.

Figura 7.3.3: Pines de un Relé. Tomado de "Relé con Arduino y ESP8266 para crear una lampara inteligente" por ProgramarFacil, https://programarfacil.com/blog/arduino-blog/rele-con-arduino-lampara/

Los pines de control son VCC, GND y IN. Tanto VCC como GND sirven para alimentar al módulo, mientras que IN recibe la señal que activa o desactiva al inductor dentro del relé. En los pines de interruptor se encuentran NC, COM y NO. NC y NO son las dos placas fijas llamadas *Normalmente Cerrada* y *Normalmente Abierta* respectivamente. La placa móvil es la denominada COM. Normalmente abierto o normalmente cerrado hace referencia a la posición de la placa móvil cuando no está alimentado el relé. A la placa fija que está en contacto con la placa móvil recibe el nombre de NC debido a que si se utilizara este pin su estado inicial seria de circuito cerrado, mientras que la placa que se encuentra separada de la placa fija recibe el nombre de NO, haciendo alusión a un circuito abierto. En un circuito se utiliza NC o NO dependiendo de la aplicación que se desee automatizar.

En un circuito de automatización de luz se suele utilizar un relé junto a un sensor LDR conectados a una tarjeta, en esta tesis será una tarjeta Raspberry. El sensor capta la intensidad de la luz, si la intensidad es

menor a un valor de umbral definido en el programa del sistema la tarjeta envía una señal de activación al inductor, el cual atraerá a la placa móvil creando un circuito cerrado que permitirá el paso de la corriente que enciende la iluminación artificial. Mientras la intensidad de la luz del sol no exceda el umbral definido la Raspberry seguirá enviando una señal al relevador que mantendrá activado al inductor permitiendo el paso de corriente que mantendrá encendida la iluminación artificial.

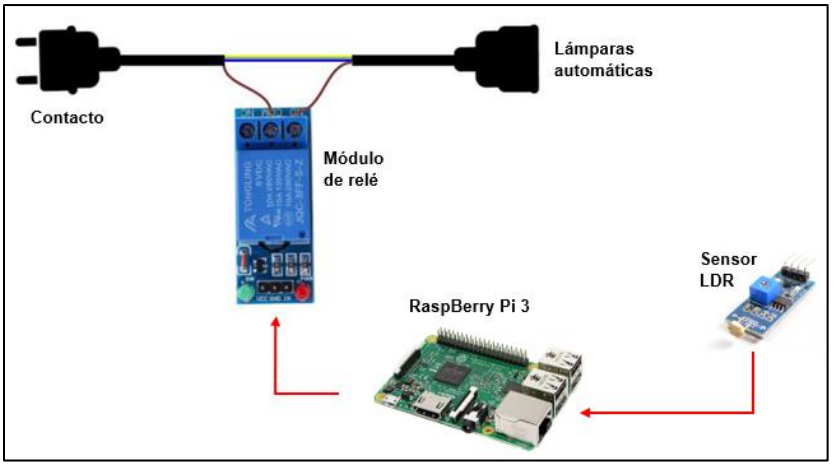


Figura 7.3.4: Diagrama de un actuador de iluminación.

7.4 Actuador para bomba de agua

Para la activación automática de una bomba de agua existen muchas alternativas, cada una depende de diversos factores como el tipo y tamaño de la bomba, la potencia de trabajo, la capacidad, etc. Las alternativas más comunes suelen ser los transistores MOSFET y los relés. La desventaja con los transistores es la poca precisión en su funcionamiento y el uso de más elementos eléctricos, como el disipador de calor cuando la potencia es alta **[XV]**. Como ya se mencionó en la sección 7.3, trabajar con relés es relativamente fácil debido a su flexibilidad al momento de trabajar con altos y bajos voltajes, además de facilitar el armado de circuitos cuando se utilizan los módulos de relés. Por tal motivo, se ha decidido utilizar un relé como actuador para la activación de la bomba de agua.

El relé estará trabajando conjuntamente con un código en Python diseñado para cambiar los estados del relé según la hora del día, lo que permitirá activar y desactivar la bomba de agua. Para trabajar con horas en tiempo real es necesario utilizar el módulo de Python *Datetime* que proporciona la fecha y hora del servidor. Además del módulo también es necesario utilizar ciclos como *If* y *While*, que permitirán la creación de un intervalo de tiempo en el cual la bomba se encenderá automáticamente durante el día y fuera de ese rango, principalmente en la noche, la bomba permanezca apagada.

A continuación, se presenta un diagrama del actuador para la activación automática de una bomba de agua:

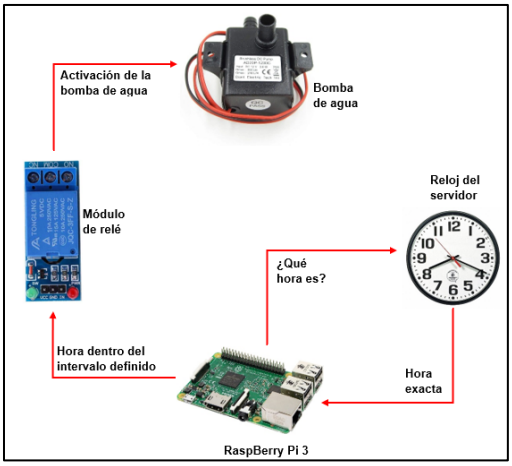


Figura 7.4.1: Diagrama de un actuador de activación de una bomba de agua.

En el diagrama se puede observar que el programa preguntará la hora exacta al servidor, si la hora se encuentra dentro de un intervalo de tiempo definido previamente, la Raspberry enviará una señal de activación al relé, y el relé cambiará su estado permitiendo el paso de la corriente que activa la bomba de agua. El programa estará diseñado para repetirse cada hora.

7.5 Sensores ultrasónico HC-SR04

El sensor ultrasónico HC-SR04 es un sensor comúnmente utilizado para calcular la distancia. Su funcionamiento es sencillo, consta de dos transductores, un emisor y un receptor llamados *Trigger* y *Echo*, respectivamente.

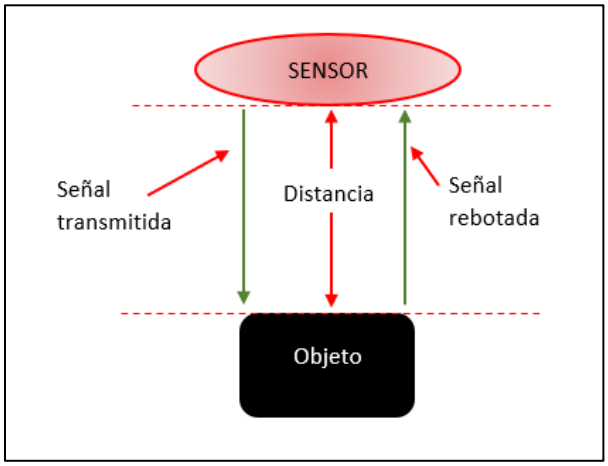
7.5.1: Sensor HC-SR04

El emisor envía una señal que viaja a la velocidad de la luz, esta señal choca con los objetos y es rebotada al receptor. El tiempo que tarda en viajar la onda desde el sensor al objeto, y luego nuevamente al sensor, puede ser calculado con ayuda de Python. Al conocer la velocidad de la señal y el tiempo que le tomó

regresar al sensor, es posible calcular la distancia de separación entre el objeto y el sensor **[XVI, XVII, XVIII]** con la siguiente formula:

$$distancia = velocidad * tiempo$$

Con ayuda de la siguiente imagen es posible analizar el proceso para obtener la distancia entre el sensor y el objeto:



7.5.2: Diagrama de distancia medida

Como se mencionó, el sensor emite una señal que viaja hasta ser rebotada por algún objeto, y lo que se obtiene a través de Python es el tiempo que tardó la señal en regresar, es decir, el tiempo en el que tardo la señal recorrer dos veces la distancia de separación entre el sensor y el objeto, con este análisis, y sabiendo que la velocidad del sonido es de $V_{sonido} = 34{,}300[\frac{cm}{s}]$, la formula de la distancia entre el sensor y el objeto quedaría de la siguiente manera:

$$distancia[cm] = (34{,}300\left[\frac{cm}{s}\right]) * (\frac{tiempo}{2}[s])$$

Donde el tiempo es el calculado por Python.

El sensor ultrasónico es de bastante ayuda para conocer la separación con algún objeto, y puede ser aplicado para conocer alturas en contenedores como será explicado más adelante.

Capítulo 8: Software

8.1 Rapsberry

Debido a que la tarjeta Raspberry Pi es un microordenador, necesita de un Sistema Operativo para funcionar. En un principio solo GNU/Linux estuvo interesado en el proyecto, por lo que la tarjeta fue diseñada para soportar Raspbian, un Sistema Operativo (SO) inspirado en Debian de GNU/Linux. Con el paso del tiempo más desarrolladores se interesaron debido a la cantidad de aplicaciones que los usuarios fueron descubriendo, por lo que en la actualidad es posible encontrar más sistemas operativos como Windows 10 IoT Core, una versión simplificada de Windows 10 adaptada para el desarrollo del Internet de las Cosas. Aunque cada vez hay más opciones de Sistemas Operativos la fundación Raspberry sigue recomendando el uso de Raspbian debido a que las tarjetas fueron diseñadas para ese SO, y los demás Sistemas aún se encuentran en sus inicios, por lo que son más susceptibles a fallas y la información aun es limitada **[10, 16]**.

Algo que siempre ha caracterizado a GNU/Linux es su flexibilidad a los cambios realizados por sus usuarios debido al código abierto que maneja. Esto permite que cualquier persona pueda modificar el código del SO para que funcione de acuerdo a sus necesidades, además de que su uso es gratuito. Estas características de GNU/Linux fueron integradas a Raspbian y es lo que hace a este SO adaptable a todas las aplicaciones de automatización e Internet de las Cosas.

El SO Raspbian es una versión simplificada de Debian que tuvo que ser modificado y adaptado debido a que la arquitectura de RaspBerry es una arquitectura ARM, similar a la utilizada por los teléfonos inteligentes, diseñada para realizar tareas con un mínimo de instrucciones, lo que permite prolongar la duración de las baterías. La arquitectura ARM tiene la desventaja de no poder ser utilizada en aplicaciones de alta potencia por lo que el uso de la Raspberry como un ordenador está limitado. Pero la baja potencia que maneja Raspberry es suficiente para hacer funcionar una red de sensores **[10, 16]**.

Existen varias versiones de Raspbian, cada una depende de la aplicación que se le dará a la Raspberry 3, para el proyecto será instalada la versión Raspberry PI OS (32-bit).

8.2 Configuración de la Rapsberry Pi

Las tarjeras Raspberry no contienen un Sistema Operativo instalado por los fabricantes por lo que es necesaria una instalación antes de su uso. Existen dos formas de llevar a cabo la instalación del SO Raspbian, manualmente y mediante imagen, puesto que la opción de imagen es la más sencilla, a continuación, se explican los pasos a seguir. Cabe recordar que el SO de una Raspberry debe ser instalado en una memoria MicroSD con una capacidad mínima de 8 Gb **[XIX]**.

1° El primer paso es acceder desde una computadora a la página oficial de descargas de Raspberry

$https://www.raspberrypi.com/software/$

Ya en la página, se selecciona la opción *Install Raspberry Pi OS using Raspberry Pi Imager*

2° Se selecciona la opción de descarga que corresponda con el Sistema Operativo de la computadora que se está utilizando. Puede ser Windows, macOS o Ubuntu.

3° Después de que se termina de descargar el archivo, se instala en la computadora siguiendo las instrucciones de las ventanas. Al finalizar la instalación se abre la aplicación de nombre *Raspberry Pi Imager*.

4° Antes de continuar en necesario conectar a la computadora la memoria MicroSD donde se instalará el SO de la Raspberry.

5° El siguiente paso es regresar a la ventana de la aplicación *Raspberry Pi Imager*, en ella primero se selecciona el SO que se desea instalar en la Raspberry, en este caso *Raspberry Pi OS (32-bit).* Después, en almacenamiento se selecciona la memoria MicroSD del paso cuatro.

6° Una vez seleccionado el SO y la memoria MicroSD se da clic en *Write* y se espera a que las configuraciones se guarden en la memoria. Finalizada la escritura en la microSD aparecerá un mensaje indicando que ya se puede retirar la memoria microSD.

7° Para continuar, se inserta la memoria MicroSD en la ranura de la Raspberry Pi y se enciende la tarjeta. Al encender el monitor desplegará un mensaje notificando que se instalará un SO. Lo que sigue es sencillo, simplemente se deben seguir las instrucciones entre las cuales se encuentra conectar la Raspberry a una red Wifi.

8° Por último, una vez establecida la conexión a internet comenzará la descarga de paquetes y demás archivos necesarios para el funcionamiento del SO. Una vez finalizada la descarga e instalación de los mismos se ha completado la instalación de Raspbian y está listo para usarse.

9° En algunos casos es posible que finalizada la instalación se despliegue una ayuda automática con la finalidad de introducir y familiarizar al usuario, además de un mensaje en el que se recomienda buscar actualizaciones disponibles del Sistema Operativo.

Además de la instalación del SO, es posible realizar más configuraciones, como el uso de dispositivos remotos para evitar la conexión del monitor, el teclado y el mouse cada vez que se utilice la Raspberry. A continuación, se describen los pasos a seguir para realizar esta configuración **[XX]**:

1° El software a utilizar es VNC Connect, que permite crear un escritorio remoto de la Raspberry Pi. La diferencia entre una conexión remota VNC y una SSH es que SSH solo permite conexiones remotas de consola, mientras VNC cuenta con las interfaces gráficas, por lo que es posible observar e interactuar con el escritorio de Raspbian. La instalación se realiza en dos partes, la

versión servidor que se instala en la Raspberry Pi, y la versión cliente, que se debe instalar en el equipo desde el que se manipulará la tarjeta.

2° Se debe instalar la versión servidor del software, llamada RealVNC, en Raspbian a través de los siguientes comandos:

$$sudo\ apt - get\ update$$
$$sudo\ apt - get\ install\ realvnc - vnc - server$$
$$sudo\ apt - get\ install\ realvnc - vnc - viewer$$

3° Después de la instalación se sigue la ruta Inicio → Preferencias → Configuración de Raspberry Pi → Interfaces, en la ventana de interfaces se habilita la opción VNC.

4° La instalación de la versión cliente depende del SO del equipo desde el que se manipulará remotamente la Raspberry. Para el caso de Windows 10 se accede a la página oficial de RealVNC y se descarga el software VNC Viewer compatible con el equipo, una vez descargado se instala siguiendo las instrucciones como cualquier otra aplicación.

5° Ya instaladas la versión servidor y la versión cliente, el paso final consiste en verificar que el logo de VNC se encuentra en la barra de tareas de la Raspberry, indicando que la interfaz VNC está habilitada. Por último, se abre VNC Viewer en el equipo que servirá como controlador remoto, se ingresa la IP de la Raspberry, y si es la primera vez que se accede a ella, por seguridad se solicitará el usuario y contraseña de la Raspberry. Ingresados los valores se da clic en continuar y si todo es correcto se desplegará la ventana remota de la Raspberry.

Capítulo 9: Conectividad

9.1 Conectividad Wifi

A partir de la versión 3 de Raspberry fue integrado un módulo Wifi a la tarjeta, lo que permite una conexión inalámbrica simple a internet. La conexión se realiza desde el momento en que se está instalando el SO en la tarjeta debido a que se necesita una conexión a internet para descargar todos los elementos necesarios para la instalación **[10, 16]**.

Raspbian cuenta con el logo de Wifi en el escritorio, por lo que solo se tiene que dar clic en el logo, seleccionar la red e introducir la contraseña para poder tener acceso a internet a través de Wifi.

9.2 Servidor y aplicación Web

Para convertir una Raspberry en un servidor Web es necesaria la instalación de una serie de paquetes conocido como "LAMP" y un software extra llamado phpMyAdmin **[XXI, XXII]**, cuyas funciones se explicarán más adelante.

El paquete LAMP debe su nombre a los softwares y el SO utilizado para la creación del servidor:

$$\boldsymbol{L} \rightarrow Linux$$

$$\boldsymbol{A} \rightarrow Apache$$

$$\boldsymbol{M} \rightarrow MySQL$$

$$\boldsymbol{P} \rightarrow PHP$$

A continuación, se hablará de cada uno de los softwares utilizados, su instalación en Raspbian y el proceso de la creación del servidor:

1° Para crear un servidor Web es necesario tener actualizado el Sistema Operativo, por lo que el primer paso consiste en actualizar el Sistema con los comandos:

$$sudo\ apt - get\ update$$
$$sudo\ apt - get\ upgrade$$

2° Instalación de Apache [XXI, XXII]:

Apache es el servidor Web que hace posible la comunicación y el intercambio de información entre un servidor y un cliente. Su instalación se realiza con un solo comando:

$$sudo\ apt - get\ install\ apache2$$

Después de la instalación es necesaria la comprobación, por lo que se debe abrir el navegador y en la barra de direcciones escribir la dirección IP de la Raspberry y presionar ENTER. Si todo funciona correctamente aparecerá la leyenda *It Works*.

3° Instalación de php [XXI, XXII]:

El lenguaje php es un lenguaje de programación Web que entre sus múltiples características se encuentra la creación de páginas Web dinámicas, ideales para la monitorización de información. El comando para su instalación es:

$$sudo\ apt-get\ install\ php\ libapache2-mod-php$$

4° Instalación de MySQL [XXI, XXII]:

MySQL es un sistema de gestión de base de datos en el cual se pueden crear, administrar y modificar grandes cantidades de información generadas. Para instalar MySQL se ejecuta:

$$sudo\ apt-get\ install\ php-mysql\ mariadb-server-10.0\ mariadb-client-10.0$$

Después de instalarse, MySQL solicitará la creación de una contraseña de seguridad. Se introduce la contraseña y se procede a reiniciar MySQL con el siguiente comando para terminar con la instalación:

$$sudo\ /etc/init.d/mysql\ restart$$

5° Instalación de phpMyAdmin [XXI, XXII]:

Este software libre permite la administración de las bases de datos de MySQL a través de un entorno gráfico mediante el navegador, lo que facilita la tarea. El siguiente comando permite la instalación de phpMyAdmin:

$$sudo\ apt-get\ install\ phpmyadmin$$

Dentro de la ventana generada se selecciona *Apache2* como servidor Web y posteriormente se selecciona la opción *Instalar la base de datos administrativa phpMyAdmin* y por último se genera una contraseña, que por facilidad se recomienda sea la misma que en MySQL **[XXI, XXII]**.

6° Vinculación de phpMyAdmin y Apache [XXI, XXII]:

La vinculación es necesaria para permitir la interacción y comunicación entre estos dos elementos. El primer paso es ejecutar el siguiente comando que abrirá un archivo llamado *apache2.conf*:

$$sudo\ nano\ /etc/apache2/apache2.conf$$

Al final del archivo se debe agregar la siguiente línea:

$$Include\ /etc/phpmyadmin/apache.conf$$

Se guardan los cambios con *Ctrl+O* y se cierra el archivo con *Ctrl+X*. Por último, se reinicia apache con:

$$sudo\ /etc/init.d/apache2\ restart$$

7° El paso final consiste en comprobar que phpMyAdmin funciona correctamente. Lo primero es ingresar al navegador y en la barra de direcciones escribir la dirección IP de la Raspberry como se muestra a continuación:

$$http://192.168.X.X/phpAdmin$$

Después de presionar ENTER aparecerá una página en la que se debe realizar un registro introduciendo *root* en usuario y en contraseña se introduce la establecida en MySQL.

Al ingresar se observará una página en la que es posible comenzar a generar de manera grafica bases de datos, y esto comprueba que la instalación fue correcta.

Con el paquete LAMP instalado ya es posible utilizar la Raspberry como servidor Web local. Se pueden crear páginas con HTML y php que solo pueden ser vistas localmente. Para poder acceder al servidor desde cualquier red externa es necesario realizar algunas configuraciones adicionales como definir una dirección IP estática en la Raspberry y definir un dominio de acceso, el cual es muy probable que tenga un costo.

9.3 Envío de alertas por email

El envió de correos electrónicos desde Python es bastante simple debido a que se cuenta con la biblioteca SMTP, *Simple Mail Transfer Protocol*, diseñada para el envío automático de correos **[XXIII]**. A continuación, se presenta la estructura básica de un programa que envía correos:

```
#Importamos la biblioteca:
import smtplib

#Definimos el texto del Asunto y el Mensaje:
asunto = "Aquí se escribe el asunto del correo"
mensaje = "Aquí escribimos el mensaje del correo"

#La siguiente línea define el formato del correo:
mensaje = "Subject: {}\n\n{}".format(asunto, mensaje)

#Indicamos la extensión del correo y el puerto utilizado:
server = smtplib.SMTP("smtp.office365.com", 587)

#Iniciamos el protocolo TLS:
server.starttls()

#Iniciamos sesión del correo destinatario:
server.login("Aquí se escribe el correo OUTLOOK completo", "Aquí se escribe la contraseña")

#Enviamos el correo:
server.sendmail("Aquí se escribe el correo destinatario", "Aquí se escribe el correo destino", mensaje)
```

Este programa puede ser adaptado, como más adelante se mostrará, para enviar correos de alerta cuando se presenten fallos dentro del sistema.

Capítulo 10: Implementación práctica

Para comenzar con la parte práctica se decidió dividir en cuatro subtemas la implementación práctica. La primera parte consistió en la preparación de la Raspberry Pi 3, la segunda en la construcción del invernadero, y las dos técnicas de cultivo hidropónico implementadas. La tercera parte se enfocó en el control individual de cada uno de los parámetros y, por último, en la cuarta parte se fusionaron los tres primeros subtemas para tener el sistema IoT completo.

10.1 Preparación de la Raspberry pi 3

El primer paso consistió en la instalación del Sistema Operativo. Para la instalación y la manipulación de la Raspberry 3 fueron necesarios algunos periféricos como un teclado, un mouse y un monitor. Además de lo anterior, para la tarjeta también fue necesario un cargador y una memoria micro SD con el SO ya grabado en ella.

Basándose en los pasos mencionados en el subtema 8.2, a continuación, se explica el grabado del Sistema Operativo en una microSD y la instalación del Sistema Operativo en la Raspberry.

1° A través de un navegador web se accedió a la página de descargas oficial de Raspberry Pi

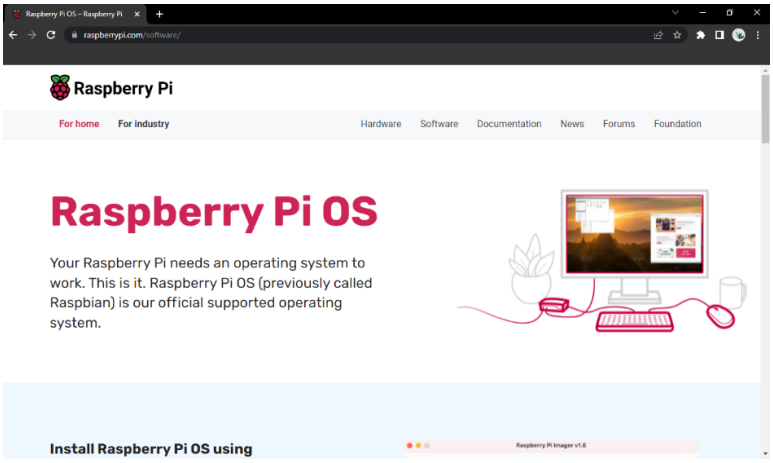


Figura 10.1.1: Página de descargas oficial de Raspberry Pi.

2° Bajando en la página se encuentra la sección *Install Raspberry Pi OS using Raspberry Pi Imager* donde se dio clic en la opción *Download for Windows,* debido a que la computadora utilizada para la escritura de la memoria microSD tiene un SO Windows 10.

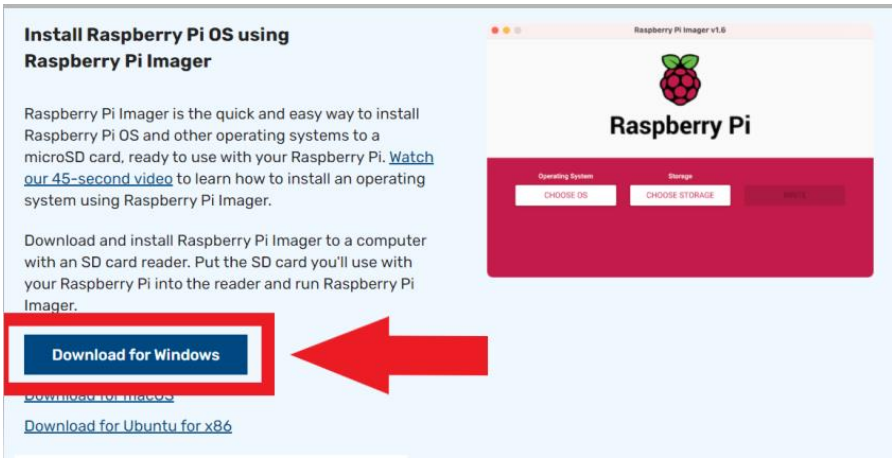


Figura 10.1.2: Selección del SO Windows.

3° Al terminar la descarga se dio clic en el archivo descargado para comenzar con la instalación del programa.

Figura 10.1.3: Ventana de instalación de "Raspberry Pi Imager".

Se siguieron las instrucciones de instalación y al final, cuando terminó la instalación, automáticamente se inició el programa llamado *Raspberry Pi Imager*.

Figura 10.1.4: Ventana de inicio de "Raspberry Pi Imager".

4° Para utilizar el programa recién instalado se insertó en la computadora la microSD a utilizar, de esta manera seria más fácil encontrarla al momento de seleccionar el tipo de almacenamiento. La memoria microSD utilizada consta de las siguientes características:

Figura 10.1.5: Memoria microSD utilizada.

Al estar conectada la computadora la reconoció como unidad de almacenamiento y le asignó el nombre *Unidad D:/.*

5° El siguiente paso fue la selección del SO que se quería instalar en la Raspberry, además de la memoria en la cual se escribiría. Las siguientes imágenes muestran cada una de las opciones y las selecciones finales.

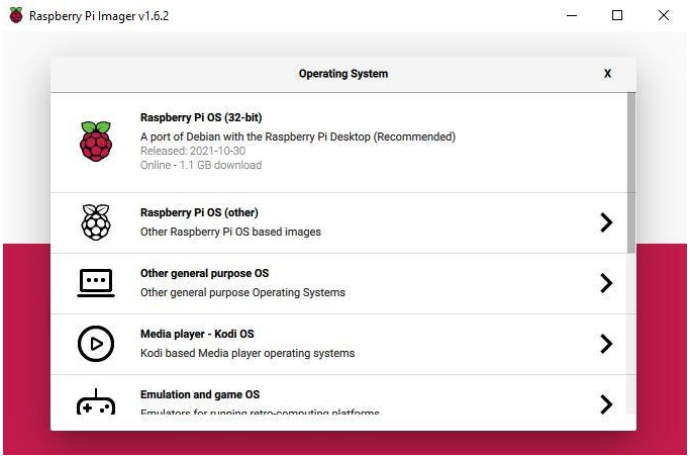


Figura 10.1.6: Sistemas Operativos disponibles para RB3.

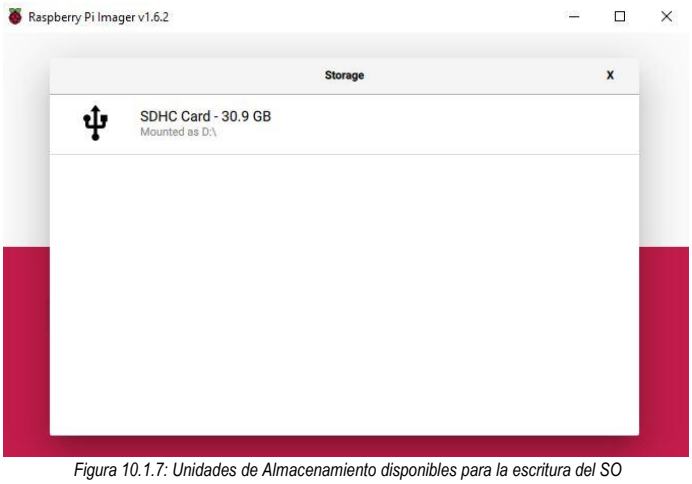


Figura 10.1.7: Unidades de Almacenamiento disponibles para la escritura del SO

Figura 10.1.8: Opciones seleccionadas.

6° Después de seleccionar el SO y la Unidad de almacenamiento se dio clic en el botón *Write* y comenzó el proceso de escritura. Al finalizar el proceso el programa arrojó un mensaje indicando que la memoria microSD estaba lista para ser removida.

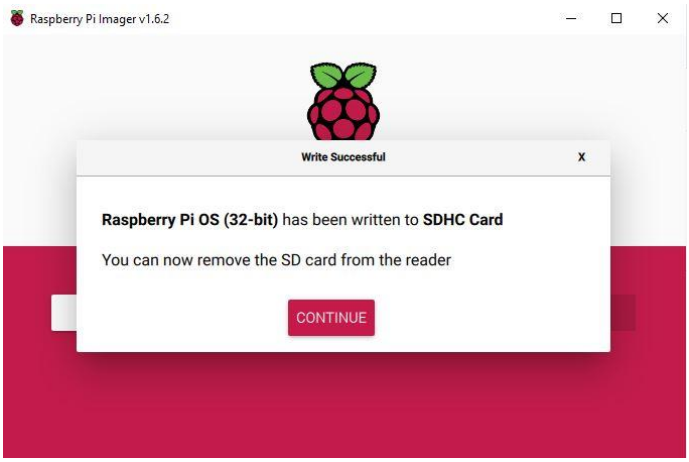


Figura 10.1.9: Mensaje que indica que la escritura finalizó.

7° La memoria microSD fue extraída de la computadora e insertada en la ranura correspondiente en la RaspBerry Pi 3. Para la instalación ya se encontraban conectados el monitor, el teclado y el mouse.

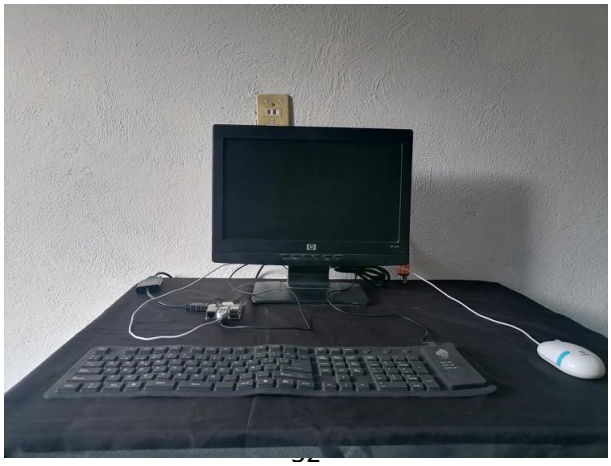


Figura 10.1.10: Raspberry con periféricos conectados antes de la instalación del SO.

Después de tener todo conectado, la memoria microSD y los periféricos, se procedió a alimentar la Raspberry para encenderla. Cuando el cargador se conectó la tarjeta se inició automáticamente.

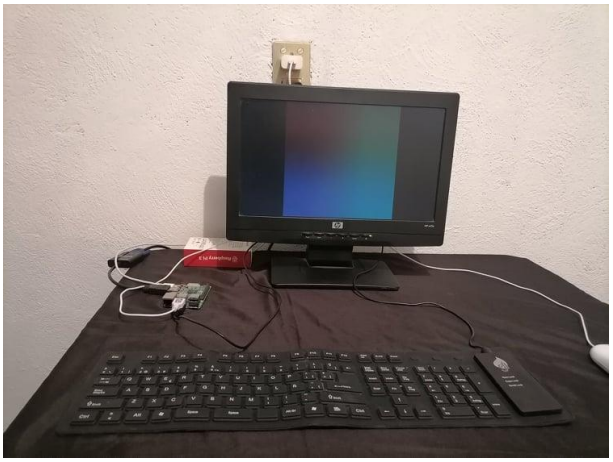


Figura 10.1.11: Inicialización de la RB3 al conectarle el cargador.

8° El paso final consistió en seguir las instrucciones de instalación entre las cuales se encuentra conectarse a una red wifi, desde donde se descargan todos los paquetes necesarios para el funcionamiento del Sistema Operativo. Al finalizar la instalación se abrió la pantalla de escritorio y apareció un mensaje de bienvenida.

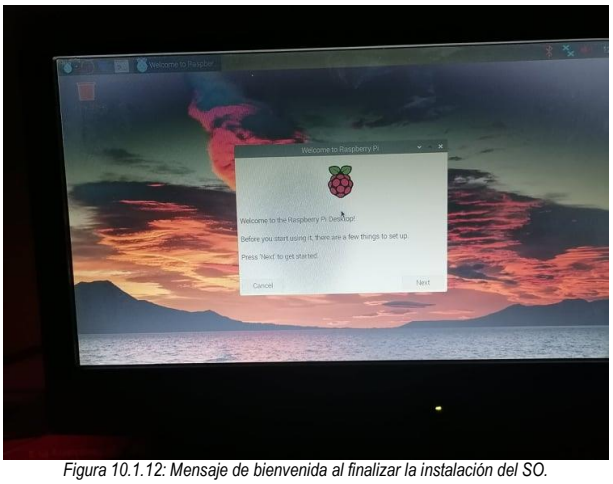


Figura 10.1.12: Mensaje de bienvenida al finalizar la instalación del SO.

Con la aparición del mensaje de bienvenida se finalizó la instalación del Sistema Operativo Raspbian.

Después de la instalación para su manipulación era necesario contar siempre con un mouse, una pantalla y un teclado, lo que se convertía en algo impráctico. Una solución a este inconveniente fue la configuración de la interfaz VNC que permite la manipulación remota de la Rasberry a través de cualquier dispositivo compatible con VNC conectado a la misma red.

Para la utilización de VNC es necesario conocer la dirección IP de la Raspberry para poder acceder a ella. Es bien sabido que en una red de área local la dirección IP es dinámica, es decir, que cambia en los dispositivos cada vez que se establece una conexión a la red. Lo anterior obligaría a conocer primero la dirección IP antes de realizar la conexión remota. Afortunadamente una Raspberry puede ser configurada

para mantener una dirección estática y evitar que cambie cada vez que establece una conexión a una red facilitando las conexiones remotas.

A continuación, se explican los pasos que se siguieron para establecer una dirección IP estática en la Raspberry:

1° La manera más sencilla de configurar una dirección IP estática a una Rasberry Pi, y cualquier otro dispositivo, es mediante la configuración del protocolo DHCP accediendo a las configuraciones del modem que proveerá de Internet a la Raspberry. En este caso un modem de Total Play. Por lo que se accedió a la dirección 192.168.100.1 a través del navegador Chrome.

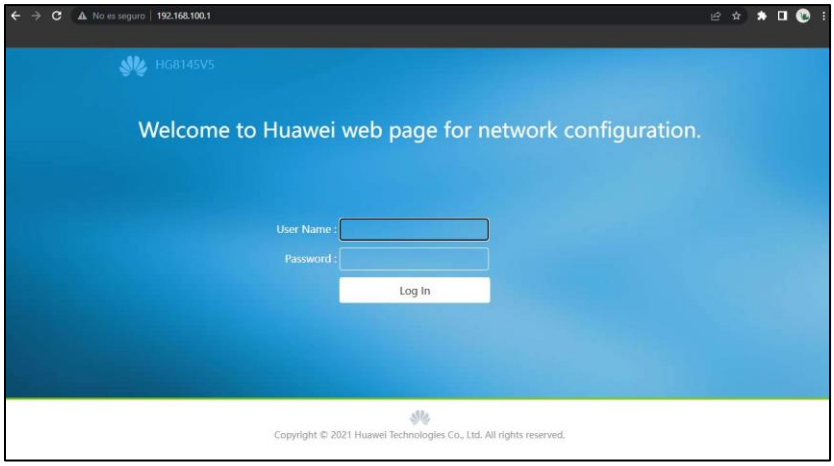


Figura 10.1.13: Dirección de acceso a modem Total Play.

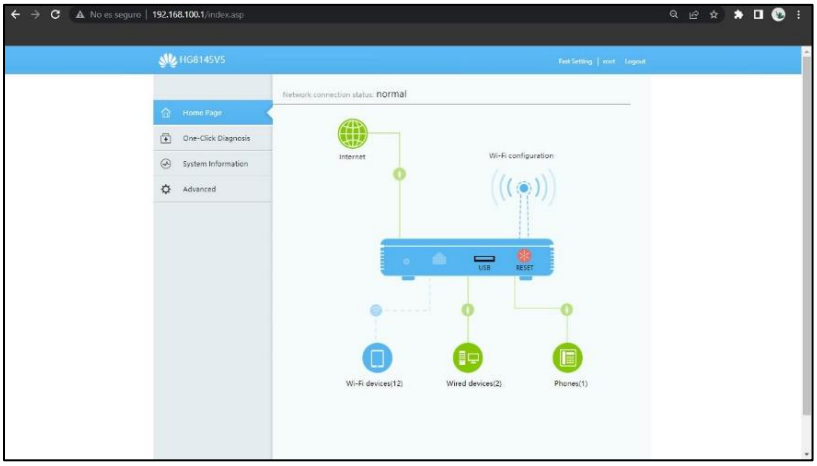


Figura 10.1.14: Página principal de configuración del modem.

2° En la página principal se siguió la ruta Configuraciones → LAN → DHCP Static IP:

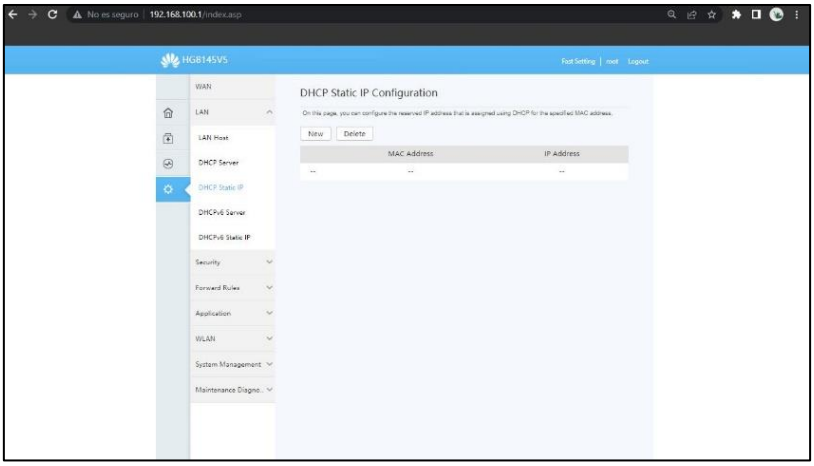


Figura 10.1.15: Configuración de DHCP.

3° Después de seguir la ruta se observó la opción para vincular un nuevo dispositivo a una IP estática. Se dio clic a la opción *New* y apareció la siguiente imagen:

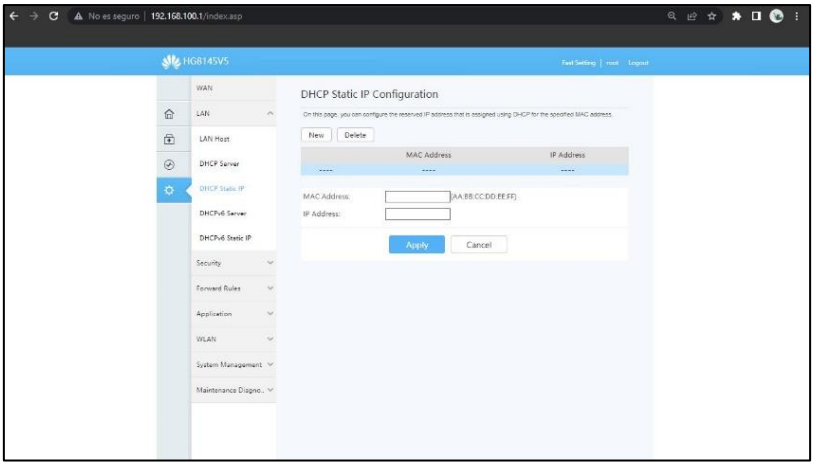


Figura 10.1.16: Ventana para agregar un nuevo dispositivo.

4° En la nueva ventana se observó que para asignar una IP estática a un dispositivo era necesaria la dirección MAC del mismo. También, fue necesario establecer que IP se asignaría al dispositivo. En este caso, y para evitar problemas de conexión, se decidió asignar la IP que en ese momento el modem le había asignado dinámicamente a la Raspberry, de esta manera nos aseguraríamos que ningún otro dispositivo conectado a la red tuviera la misma dirección.

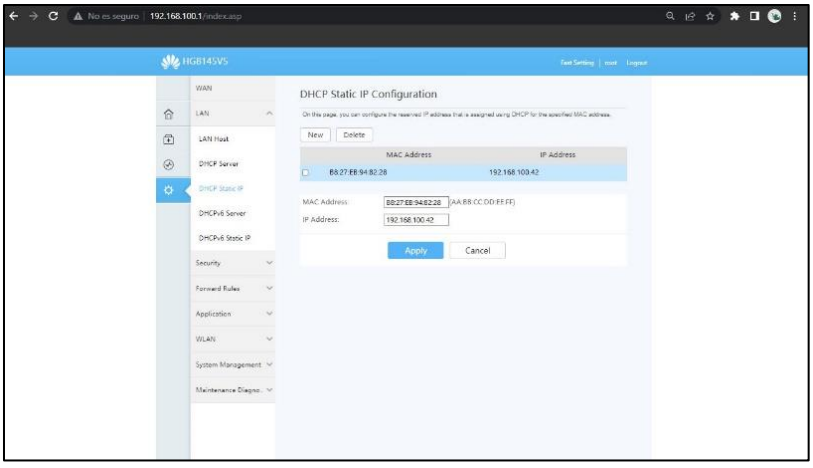


Figura 10.1.17: Asignación de una IP estática a la RB3

5° Después de agregar las direcciones MAC e IP se dio clic en *Apply* y el modem reservó esa dirección exclusivamente a la Raspberry 3. Para comprobar que el proceso para la asignación fue correcto, se apagó la Raspberry y se encendió nuevamente, y se comprobó que su dirección IP era 192.168.100.42.

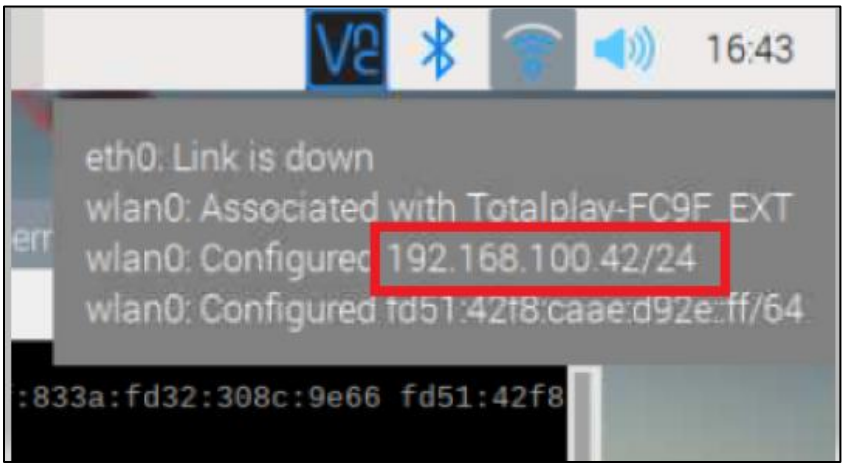


Figura 10.1.18: Comprobación de IP estática.

Para continuar con la preparación de la Rasberry el siguiente paso consistió en la implementación de la interfaz VNC para poder manipular remotamente la Raspberry. Igualmente, el proceso se realizó siguiendo los pasos establecidos anteriormente.

1° Primero se estableció utilizar una laptop con sistema operativo Windows 10 para el acceso remoto a la Raspberry Pi 3.

2° Con la ayuda del teclado, el ratón y un monitor se accedió a la ventana de comandos de la Raspberry, donde se ejecutaron los comandos

$$sudo\ apt - get\ update$$
$$sudo\ apt - get\ install\ realvnc - vnc - server$$
$$sudo\ apt - get\ install\ realvnc - vnc - viewer$$

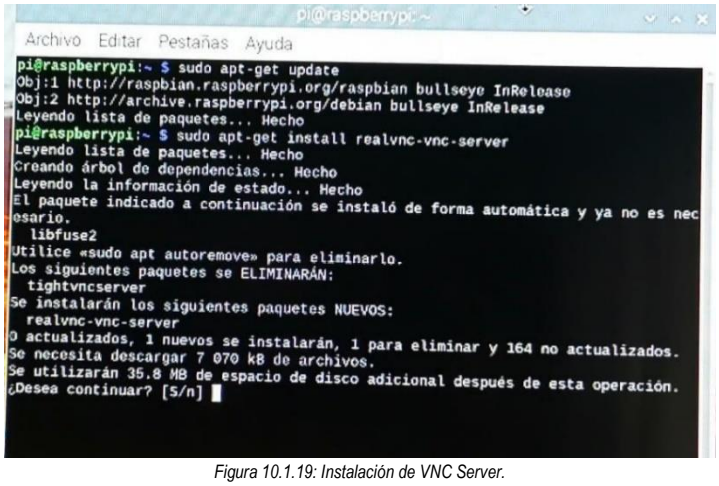


Figura 10.1.19: Instalación de VNC Server.

Los comandos se ejecutaron uno a uno aceptando los permisos que se solicitaron.

3° Después de ejecutar los tres comandos del paso 2, lo único que restaba en la Raspberry era la activación de la interfaz VNC. La activación se realizó siguiendo la ruta Inicio → Preferencias → Configuración de Raspberry Pi → Interfaces. Donde se desplegó la siguiente ventana:

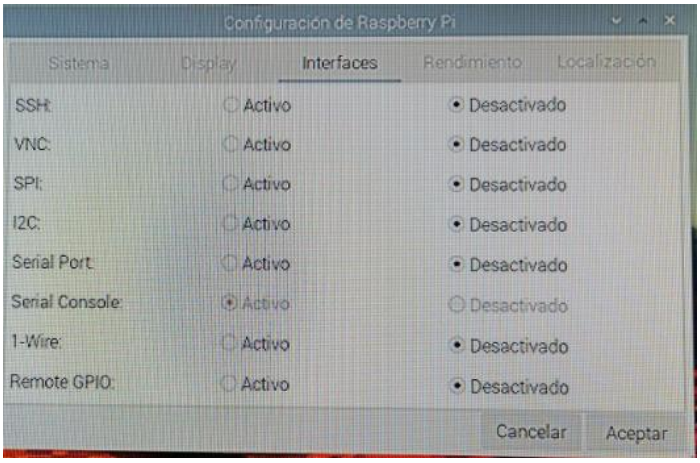


Figura 10.1.20: Ventana de Interfaces.

En la ventana de interfaces se activó VNC y se dio clic en aceptar para aplicar los cambios realizados.

4° Una vez lista la Raspberry, el siguiente paso fue la instalación de VNC Viewer en la laptop. Para iniciar la descarga se accedió a la página oficial https://www.realvnc.com/es/connect/download/viewer/, donde se dio clic en la opción *Descargar VNC Viewer* para el Sistema Operativo a utilizar, en este caso Windows 10. Una vez descargado se siguieron las instrucciones de instalación y se desplegó la ventana de inicio. En la ventana de inició se desplegó una nueva ventana para agregar un escritorio remoto, donde además de la dirección IP, se solicitaba el nombre de usuario y la contraseña del dispositivo al que nos queríamos conectar remotamente.

5° El último paso consistió en ingresar los datos solicitados, y tras comprobar que eran correctos se logró la conexión remota de la Raspberry, por lo que desde ese momento ya no fue necesaria la utilización de periféricos para su control.

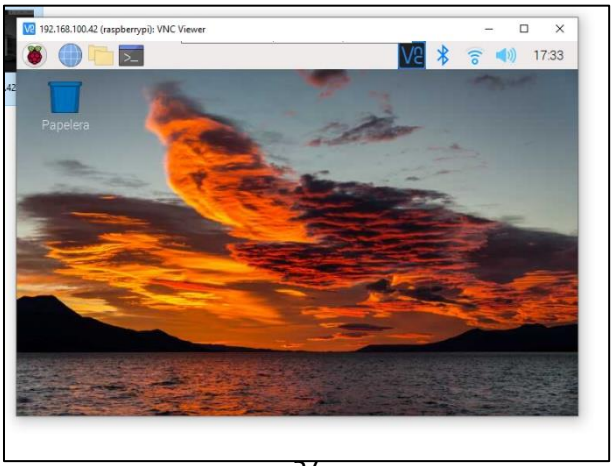


Figura 10.1.21: Conexión a escritorio vía remota.

Después de haber configurado la Raspberry para ser manipulada remotamente se comenzó con el diseño del sistema. El sistema se basa en un servidor el cual concentra a todos los dispositivos como sensores, actuadores, placas, etc. Estos dispositivos generan información que es almacenada en bases de datos que se encuentran disponibles para que los usuarios puedan acceder a ellas a través de la red.

Para comenzar el diseño del sistema fue necesaria la instalación de la pila LAMP en la Raspberry, que agrupa las tecnologías necesarias para utilizar a la Raspberry como un servidor Apache que contenga bases de datos MySQL y que permita la creación de páginas Web dinámicas a través de PHP. Para la instalación de LAMP se siguieron las instrucciones explicadas el subtema 9.2 y son explicadas a continuación:

1° Siempre que se realizan modificaciones e instalan paquetes a la Raspberry es necesario buscar e instalar las ultimas actualizaciones. Por lo que primer paso fue la ejecución de los comandos

$$sudo\ apt - get\ update$$

$$sudp\ apt - get\ upgrade$$

Aceptando los permisos que se solicitan mientras se ejecutan los comandos. Terminada la actualización se procedió a la instalación de la pila LAMP.

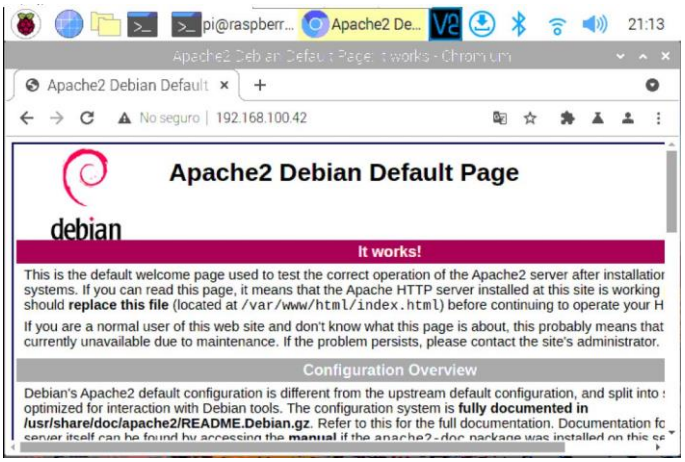
2° El primer elemento de la pila LAMP es Apache, se instaló mediante el comando

$$sudo\ apt - get\ install\ apache2$$

Del que igualmente fue necesario aceptar todos los permisos que solicitaba para la instalación de Apache. Después de aceptarlos la instalación fue casi inmediata.

10.1.22: Instalación de Apache.

Terminada la ejecución del comando se ingresó al navegador Web para comprobar que Apache se había instalado correctamente. En el navegador se accedió a la Rasberry a través de su dirección IP 192.168.100.42. Al ingresar se desplegó una página Web de apache con la leyenda *It Works* lo que indicaba que Apache ya se encontraba instalando y funcionando correctamente en la tarjeta.

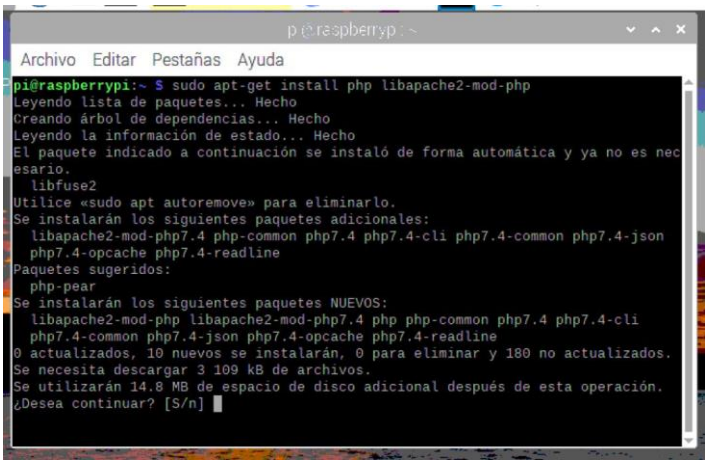


10.1.23: Comprobación de la instalación de Apache.

3° Para la instalación de PHP se utilizó el comando

$$sudo\ apt - get\ install\ php\ libapache2 - mod - php$$

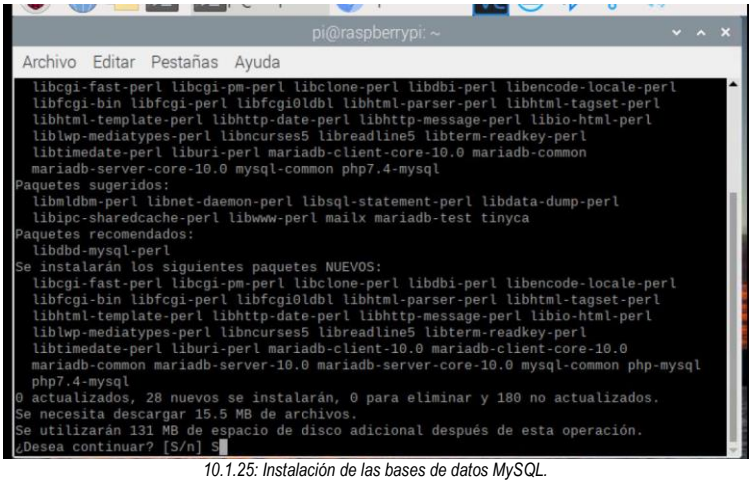
Nuevamente fue necesario aceptar los permisos solicitados. Después de aceptar comenzó la instalación y duró solo unos segundos.



10.1.24: Instalación de PHP.

4° El siguiente paso consistió en la instalación de las bases de datos MySQL para PHP y Mariadb, que es una versión de MySQL adaptada a Debian. La instalación de ambas bases de datos se realizó con el comando:

$$sudo\ apt - get\ install\ php - mysql\ mariadb - server\ mariadb - client$$



10.1.25: Instalación de las bases de datos MySQL.

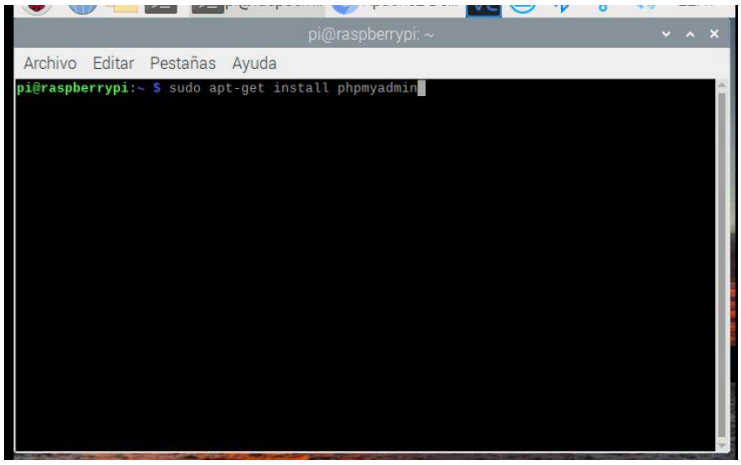
En este caso también fue necesario aceptar los permisos.

Una vez instaladas MySQL y Mariadb, se reinició MySQL para guardar todos los cambios en la tarjeta. El reinicio se hizo con el comando

$$sudo\ /etc/init.d/\ mysql\ restart$$

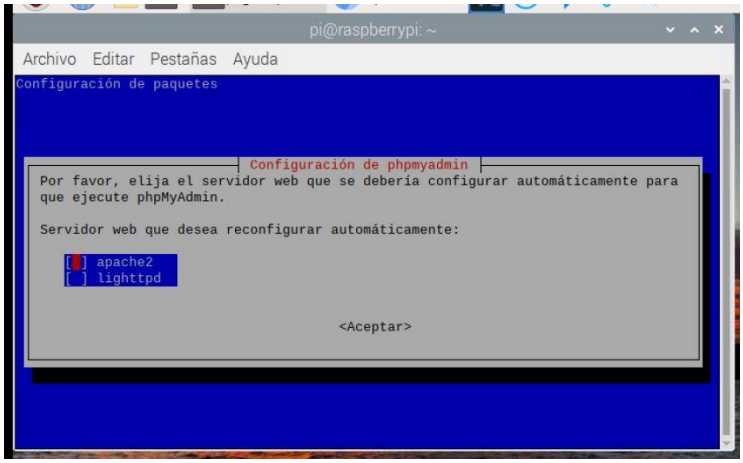
5° Para manipular las bases de datos de manera gráfica y sencilla se instaló PHPMyAdmin ejecutando el comando

$$sudo\ apt-get\ install\ phpmyadmin$$



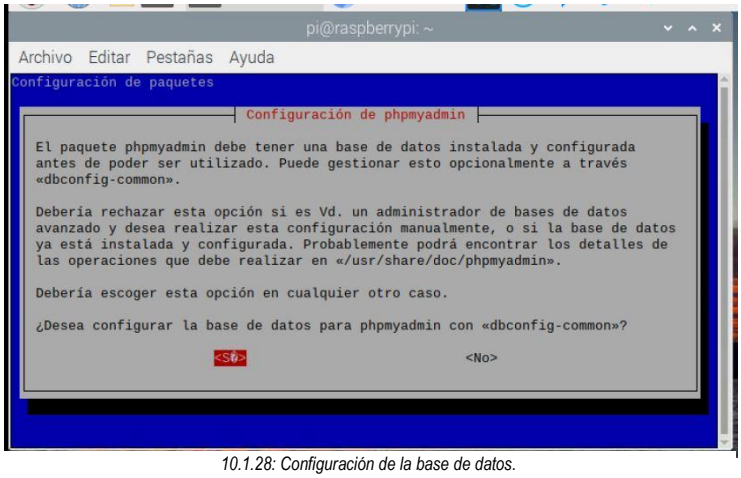
10.1.26: Instalación de las bases de datos PHPMyAdmin.

Después de aceptar los permisos se abrió una ventana preguntando el servidor Web a utilizar, en este caso se seleccionó Apache2.



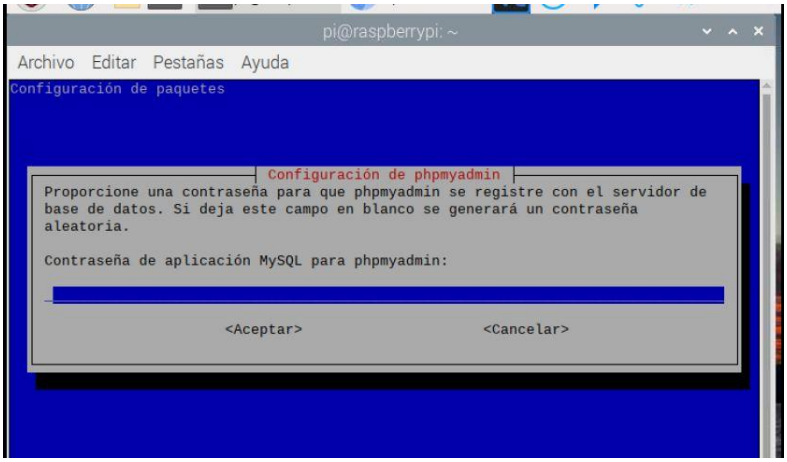
10.1.27: Selección del Servidor Web.

Posteriormente, apareció una segunda ventana la cual preguntaba si se deseaba configurar la base de datos para PHPMyAdmin. En esta ventana se seleccionó la opción *Sí*.



10.1.28: Configuración de la base de datos.

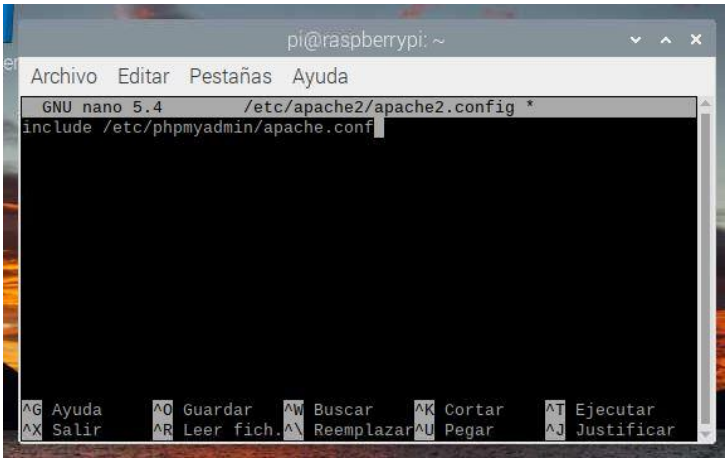
Para terminar con la instalación de PHPMyAdmin nuevamente apareció una ventana solicitando una contraseña para manejar PHPMyAdmin. En este caso se designó la contraseña *TesisIoT*, que será la que se utilice en este proyecto siempre que solicite una contraseña.



10.1.29: Generación de contraseña.

6° Debido a que PHP y Apache son dos herramientas completamente diferentes fue necesario realizar una vinculación entre ambas para poder trabajar en conjunto. Esto se realizó a través de la creación de un archivo PHP llamado *Apache2.config* al que se le agrega la línea:

$$include\ /etc/phpmyadmin/apache.conf$$



10.1.30: Vinculación de PHP y MySQL.

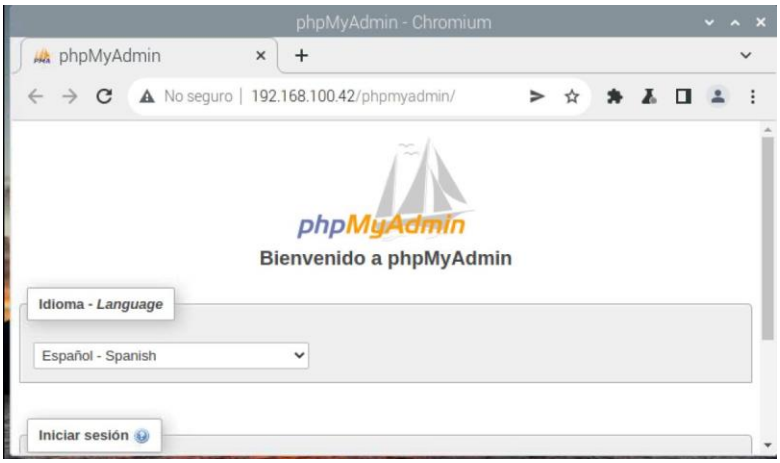
El archivo fue guardado y cerrado. Y, por último, se reinició nuevamente apache con

$$sudo\ /etc/init.d/apache2\ restart$$

7° Para terminar con la instalación de la pila LAMP, el paso final fue comprobar que PHPMyAdmin se había instalado correctamente. Esto se hizo accediendo a través de un navegador Web a

$$192.168.100.42/myphpadmin$$

Debido a que la pagina cargo correctamente se comprobó que la pila funcionaba correctamente.



10.1.31: Página de PHPMyAdmin

Con la instalación de la pila LAMP finalizó la primera parte de la implementación práctica, donde fueron instalados los paquetes básicos y realizadas las configuraciones necesarias para comenzar con el diseño del Invernadero Hidropónico Inteligente.

10.2 Construcción del invernadero

Debido a que el objetivo de la tesis va enfocado al diseño de un sistema inteligente, la construcción del invernadero se explicará de manera muy superficial. El diseño consta de una estructura de madera cubierta con plástico transparente para permitir el paso de la luz del sol. El siguiente boceto muestra la idea inicial del invernadero.

10.2.1: Boceto del invernadero

Como se mencionó al principio, se buscaba que el invernadero se adaptara a cualquier espacio de la casa, por lo que se decidió instalarlo en un pequeño espacio de la azotea.

10.2.2: Espacio de instalación.

El resultado final fue muy parecido al diseño del boceto. La estructura se realizó con tiras de madera de 5 [cm] de ancho. La madera fue lijada e impermeabilizada para evitar afectaciones con la lluvia. Para armar la estructura se utilizaron escuadras de metal que fueron fijadas con pijas para madera.

10.2.3: Unión de dos tiras de madera.

Una vez terminada la estructura, se colocó una pared de madera de 1 [m] de alto sobre la estructura del invernadero. El objetivo de la barda fue proteger el invernadero de los animales, además de darle más solides a la estructura y evitar que se moviera con el viento. La madera utilizada para la pared fue madera reciclada de muebles, una puerta y dos hojas de triplay. La pared también fue impermeabilizada.

10.2.4: Estructura terminada.

Después de comprobar que la estructura ya no se movía fue fijada a una pared de concreto y al piso con clavos y tornillos. De esta manera se evitarían accidentes con el viento. El paso final consistió en la colocación del plástico en el resto de la estructura. Se utilizó un plástico comercial llamado *Frost Kings* de 3 [m] por 7.6 [m] aprox. El plástico fue fijado a la estructura con pijas, clavos y rondanas.

10.2.5: Fijación del plástico a la estructura de madera.

Durante la construcción del invernadero y la fijación del plástico se tuvo en cuenta la ventilación, por lo que se dejaron los dos orificios donde se colocarían los ventiladores para la circulación del aire al interior.

10.2.6: Orificio inferior para el invernadero.

10.2.7: Orificio superior para el invernadero.

Las dimensiones finales del invernadero fueron 2 [m] de ancho por 2.5 [m] de largo, con una altura mínima de 2 [m] y una máxima de 2.2 [m]. El propósito de la diferencia de alturas fue tener una inclinación en el techo del invernadero y evitar la acumulación de agua de lluvia.

10.2.8: Vista frontal del invernadero terminado.

La imagen muestra la vista frontal del invernadero terminado. Se observa la estructura realizada con tiras de madera, la pared hecha con madera reciclada y el plástico cubriendo el resto del invernadero.

Después de construir el invernadero se procedió a construir las estructuras de las dos técnicas de cultivo hidropónico que serían utilizadas. Al igual que con la construcción del invernadero, la implementación de las dos técnicas será explicada de manera muy superficial debido a que el objetivo del proyecto es la implementación de tecnología al invernadero.

Para la construcción de la cama de cultivo capilar se utilizaron cuatro tablas de 35 [cm] de ancho para simular una caja, de las cuales dos tablas tenían 1 [m] de largo, y las otras dos 1.5 [m].

Las cuatro tablas se unieron con clavos formando las paredes de una caja de 35 [cm] de alto, para evitar la filtración de agua se utilizaron varias capas de plástico negro que cubrieron el piso y las paredes interiores de la caja, se colocó el tubo de PVC que funcionaría como depósito de agua y se agregó la capa de grava para ayudar a la retención del agua.

10.2.9: Capa de graba en la cama de cultivo capilar

Después de colocar la grava se colocó malla mosquitera y encima de la malla se colocó la capa de tierra, el objetivo de la malla fue evitar que la tierra bajara a la capa de graba.

10.2.10: Cama de cultivo capilar

Al final la caja fue completamente cubierta de tierra, donde se cultivaron las primeras semillas. También, por el tubo de PVC se agregó agua a la capa de grava, que es de donde las raíces de las plantas absorben el agua.

10.2.11: Tubo de PVC que proporciona agua al cultivo

Para la hidroponía NFT se utilizaron cuatro segmentos de PVC de 3 pulgadas ancho por 2 metros de largo. Los segmentos fungirán como canales en los que se encontraran las plantas y la solución nutritiva fluirá a través de ellos.

Para realizarlos se colocaron dos soportes de madera de 1.7 [m] de alto.

10.2.12: Soportes de canales

En los soportes se colocaron argollas y alambre para sostener los canales que previamente fueron perforados para colocar las plantas.

10.2.13: Colocación de canales

Después de colocar los cuatro canales comenzó la colocación de la tubería de ½ pulgada de ancho para el descenso de la solución.

10.2.14: Instalación completa de los canales

En la imagen se observa que de lado izquierdo se tienen tanto el codo superior y el desagüe del último canal. Por el codo superior, que se encuentra en el canal más alto, entrará el agua con solución nutritiva, recorrerá todos los canales y regresará al contenedor, para de nuevo ser llevada por la bomba al canal superior completando el ciclo de recirculación.

Con el contenedor de solución nutritiva, la bomba y la manguera que llevan la solución al canal más alto el sistema, se finalizó la instalación del cultivo NFT. A continuación, se presenta una imagen del sistema completo.

10.2.15: Canales para cultivo NFT terminados

El subtema 10.2 de construcción del invernadero e instalación de las técnicas de cultivo utilizada fue explicado superficialmente debido a que el propósito del proyecto consiste en la implementación de los sistemas tecnológicos que le darán inteligencia al invernadero. Por lo que a continuación se explica detalladamente el diseño de los sistemas, así como la implementación práctica.

10.3 Implementación del sistema de control de los parámetros

En esta etapa se explicará cómo se diseñó el sistema de cada uno de los parámetros, los sensores utilizados, las conexiones, el código Python implementado, el ingreso de los valores a MySQL y su despliegue en una página Web mediante php. También se explica la manera en que los sistemas envían notificaciones al cliente mediante correo electrónico.

Para facilitar la explicación este subtema se divide en cada uno de los sistemas que se diseñó.

10.3.1 Sistema de Control de Humedad y Temperatura

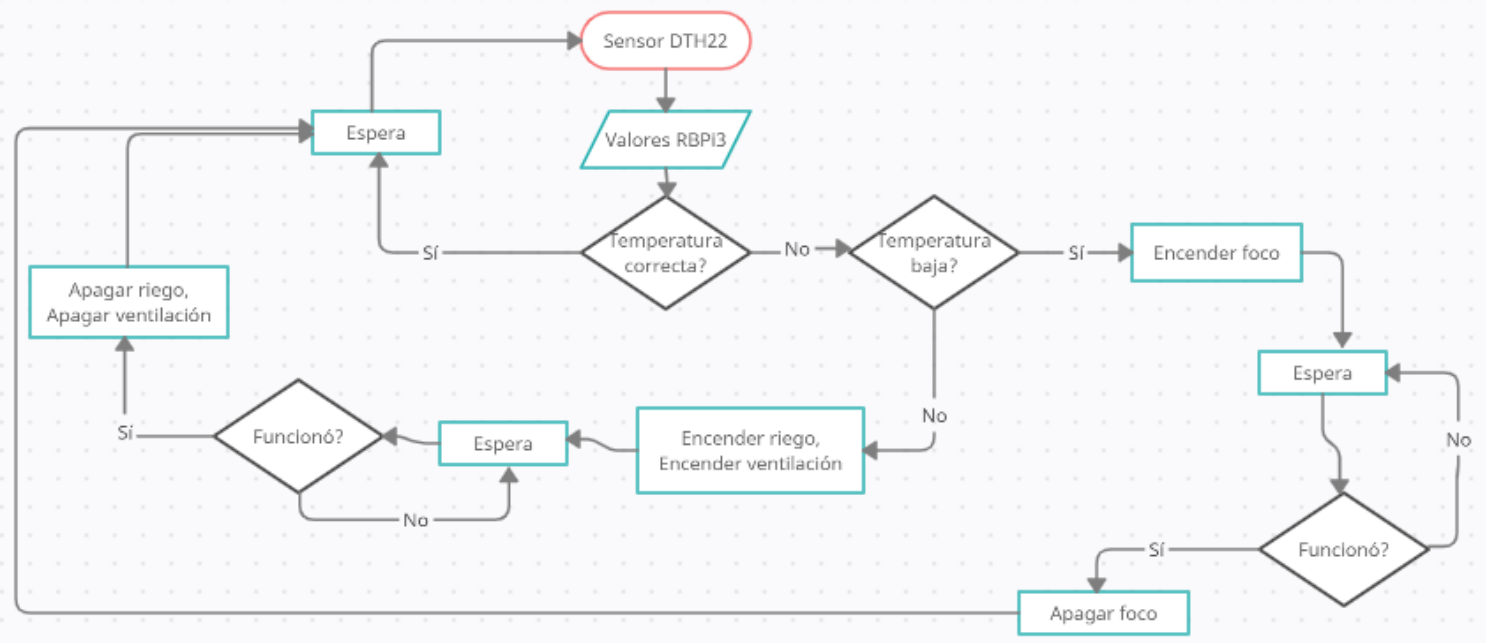
El primer pasó consistió en tener una idea clara de los parámetros, de esta manera seria fácil diseñar un diagrama de flujo en el cual basarse para el desarrollo del sistema. A continuación, se explica el proceso.

Para el parámetro de temperatura se esperaba que el invernadero tuviera una temperatura entre los 10 – 25 [°C]. Si la temperatura salía de este rango el sistema tendría que saber si la temperatura medida se encontraba por arriba o por abajo del rango establecido para tomar una decisión. Si la temperatura fuera menor se tendría que encender un foco especial para invernadero, que ofrece una radiación similar al sol que calienta al invernadero, de esta manera la temperatura comenzaría a subir.

Si la temperatura fuera mayor se tendría un sobrecalentamiento en el invernadero, en este caso se tendrían que activar el riego y la ventilación para enfriar rápidamente el interior a través de la circulación del aire húmedo.

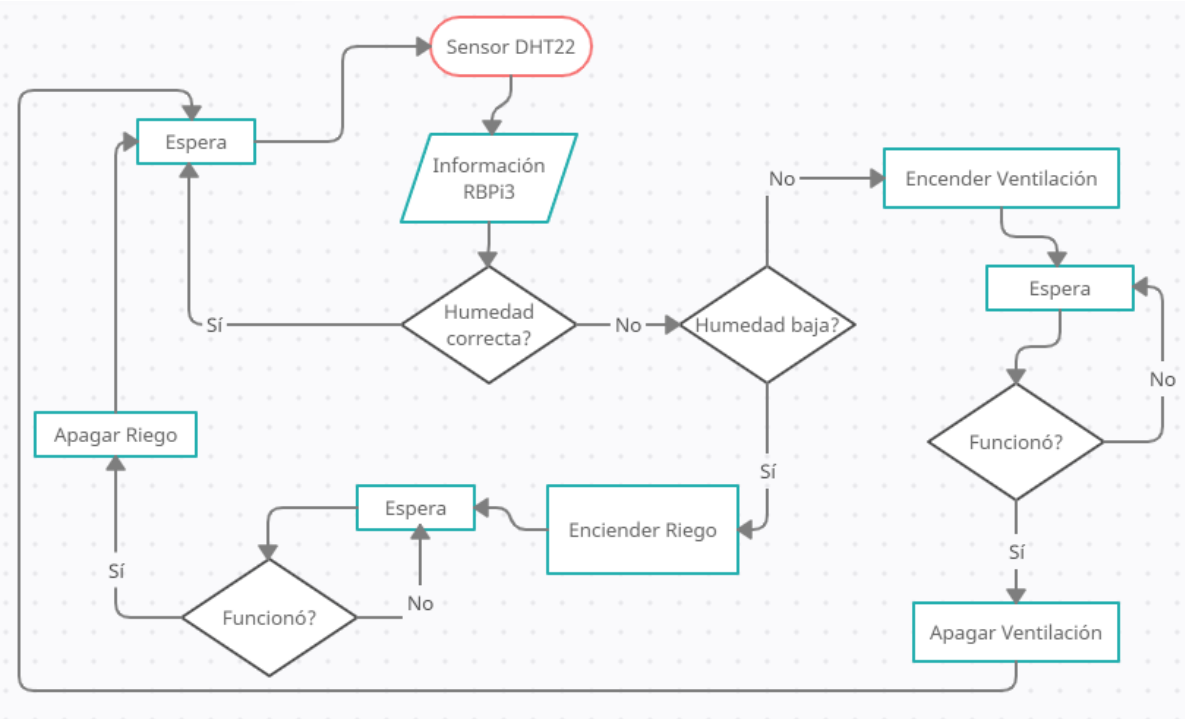
Teniendo en cuenta los procesos explicados anteriormente, y los tiempos de espera, el diagrama de flujo de temperatura fue el siguiente:

Algo similar ocurriría con la humedad, después de obtener su valor el sistema sería capaz de determinar si se encontraba en un rango de 50 – 75 %.



En caso negativo el sistema tendría que activar el riego para valores inferiores a 50% o activar la ventilación para valores mayores al 75%. Después de analizar la idea y las posibles circunstancias el diagrama de flujo quedó de la siguiente manera.

En esta sección se está realizando el análisis de los parámetros temperatura y humedad juntos debido a que ambos se realizan con el mismo sensor, el DTH22. Y también, utilizan los mismos elementos para



tener controlados sus valores que son riego, ventilación y una tira LED especial para plantas que produce calor.

Los diagramas de flujo de Temperatura y Humedad sirvieron como guía para realizar el sistema. El primer paso, fue la creación de una base de datos en MySQL donde se guardarían los datos obtenidos por el sensor DTH22, además de la fecha y hora para tener un mejor control de los parámetros.

A continuación, se menciona el paso a paso de la creación de la base de datos donde se almacenan los valores:

1° En la ventana de comandos de la Raspberry Pi se tecleó el comando:

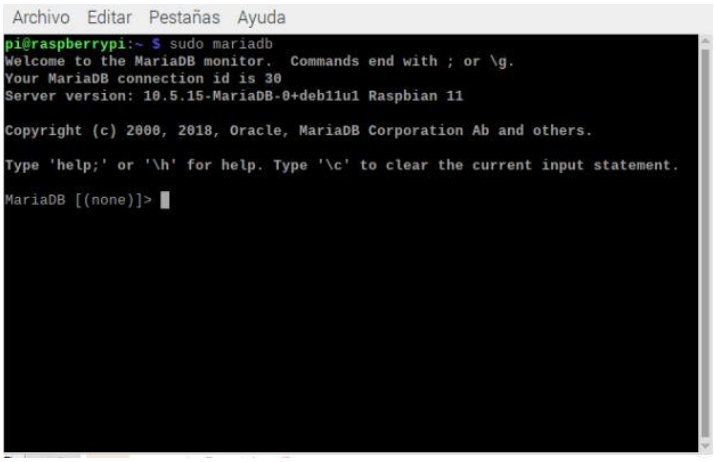
$$sudo\ apt - get - y\ install\ python - mysql.connector$$

El comando fue necesario para la instalación de los paquetes que permiten una conexión entre un programa desarrollado en Python y MySQL.

2° También, se ejecutó el comando:

$$sudo\ mariadb$$

Que desplegó la ventana de comandos de MySQL donde se pueden crear y manipular las bases de datos del servidor. En este caso se tuvo que ingresar para crear la base de datos para el sensor.



10.3.1.1: Ventana de Comandos MySQL

3° Dentro de Mariadb para la creación de la base de datos primero fue necesario crear un usuario y su contraseña, además de especificar el nombre del host. En nuestro caso, el usuario fue *pi*, la contraseña *TesisIoT2021*, y el host *localhost*. La creación del usuario se realizó mediante el comando:

$$CREATE\ \ USER\ \ \text{pi"}@localhost\ \ \text{IDENTIFIED BY}\ TesisIoT2021";$$

Después del *;* se presionó *Enter* para ejecutar el comando.

4° Para poder manipular las bases de datos el usuario debía tener todos los permisos. Para otorgárselos se ocuparon los comandos:

$$GRANT\ ALL\ PRIVILEGES\ ON\ *.*\ TO\ \text{"pi "@"localhost";}$$

$$FLUSH\ PRIVILEGES$$

De nuevo, al final de cada comando se presionó *Enter* para ejecutarlo.

5° Por último, para salir de Mariadb se ejecutó:

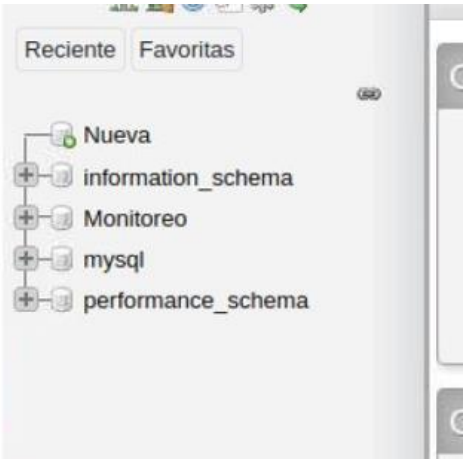
$$quit$$

6° Para comprobar que el usuario se había creado correctamente, se accedió a

$$192.168.100.42/phpmyadmin$$

Donde el usuario y la contraseña fueron los definidos en el paso tres.

7° Ya dentro de phpMyAdmin, en la parte izquierda se mostraban todas las bases de datos creadas por defecto. Se dio clic en la opción *Nueva* y se creó una base de datos llamada *Invernadero*.

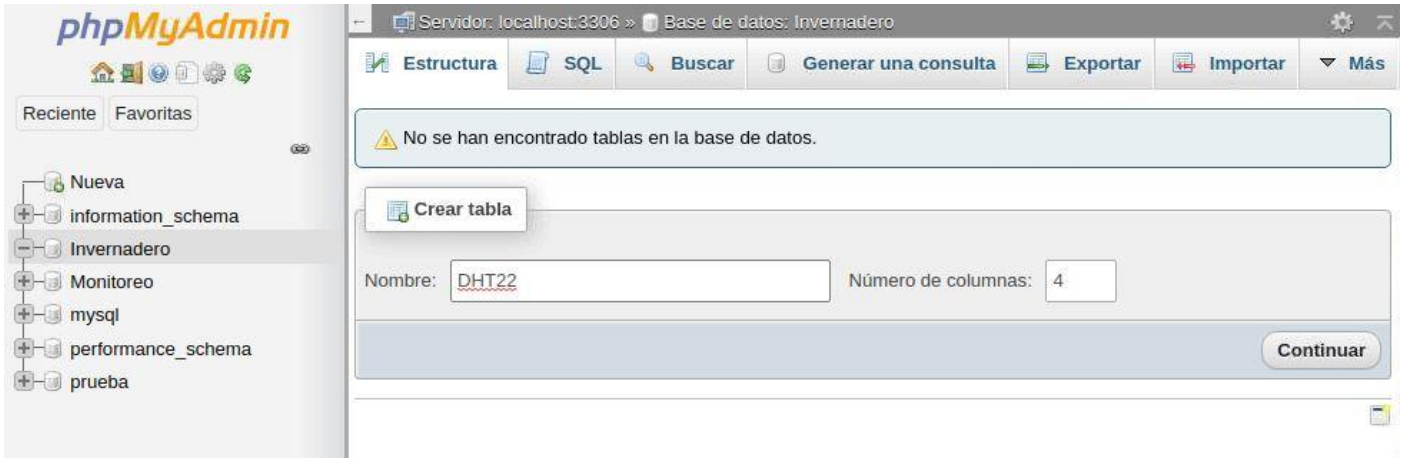


10.3.1.2: Bases de datos por defecto

10.3.1.3: Creación de una base de datos llamada "Invernadero"

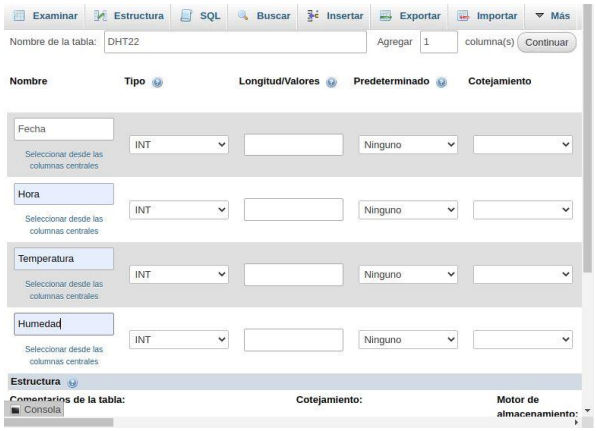
8° Después de dar clic en *Crear*, la base de datos se visualizó en la lista junto a las tablas creadas por defecto. Se dio clic en la base de *Invernadero,* se abrieron nuevas opciones y se hizo clic en la opción *Crear tabla*, donde se guardarían todos los valores, en este caso la tabla fue nombrada *DHT22* y se configuró con cuatro columnas.

Para terminar el paso ocho, se dio clic en la opción *Continuar*.



10.3.1.4: Creación de la tabla "DTH22" dentro de la base de datos "Invernadero"

9° Se cargó una nueva ventana en donde se solicitaba ingresar el nombre de cada columna, así como el tipo de variable que almacenarían. En este caso los nombres de las columnas fueron *Fecha*, *Hora*, *Temperatura* y *Humedad*. Con variables *Date*, *Time*, *Float* y *Float* respectivamente.



10.3.1.5: Nombres de las columnas de la tabla "DHT22"

Después de crear la base de datos que almacenaría los valores del sensor *DTH22* se procedió a la creación del código Python.

Para iniciar el código fue necesaria la instalación de los paquetes *ADAFRUIT*, que son los que permiten la interacción entre Python y el sensor *DHT22*. La instalación se realizó con el comando:

$$pip3\ install\ adafruit-circuitpython-dht$$

El primer código que se realizó fue una prueba para comprobar que el sensor funcionaba correctamente. Se abrió un archivo mediante el comando:

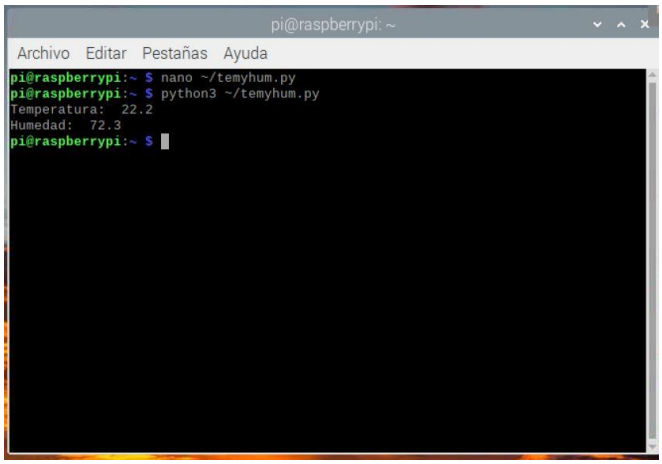
$$nano\ \sim/temyhum.py$$

Donde *tempyhum.py* fue el nombre que se le dio al programa de prueba. Al abrirse el archivo se escribió el código:

```
import adafruit_dht
import board
import time

# Se especifica el tipo de sensor
SENSOR = adafruit_dht.DHT22(board.D2, use_pulseio = False)

# En el bloque "try" se especifica el código del programa
try:

    TEMPERATURA = SENSOR.temperature
    HUMEDAD = SENSOR.humidity

    print ("Temperatura: ", TEMPERATURA)
    print ("Humedad: ", HUMEDAD)

    time.sleep(5)

# En el bloque "except" se especifica que hacer en caso de algún tipo de error, de esta manera se evita que el programa se detenga
except RuntimeError as error:

    print ("Recalculando valores, por favor espere...")
```

Los segmentos *try* y *except* fueron necesarios debido a que el sensor es propenso a errores. Si no existieran *try* y *except* el código se detendría debido a fallas en las mediciones. En caso de presentarse un error se le indicó al programa que no se detuviera, si no que desplegara el mensaje "*Recalculando valores, por favor espere…*". En este código parece no ser tan importante esta parte del código, pero más adelante se volvió fundamental para que el sistema funcionara descartando dichos errores.

Al ejecutarse dio como resultado una temperatura de 22.2 [°C] y una humedad de 72.3%.



10.3.1.6: Prueba de funcionamiento del sensor DHT22

El segundo código realizado consistió en realizar un programa que de nuevo midiera humedad y

temperatura, pero ahora de manera cíclica. Para ello se llevó el siguiente análisis: Debido a que el sensor *DHT22* es propenso a errores nuevamente se utilizarían los bloques *try* y *except*, en los cuales se especifica que hacer en un funcionamiento normal del programa, y que hacer en caso de presentarse un error. Para reducir las probabilidades de error y evitar que el programa se detuviera completamente en caso de presentarse un error se decidió crear una función por cada parámetro. Es decir, una función para temperatura y otra función para humedad, de esta manera al presentarse un error en la medición, por ejemplo, la humedad, solo se tendría que ejecutar el segmento de código de la humedad, ahorrando tiempo y procesamiento redundante en la medición de la temperatura.

En este código, también se especificaron los límites de variación de los parámetros, así como los mensajes que el programa tenía que desplegar según los valores de humedad y temperatura.

A continuación, se presenta el código con explicaciones en los comentarios:

```
#i Importación de bibliotecas
import adafruit_dht
import board
import time
import RPi.GPIO as GPIO

#Se define el tipo de sensor a utilizar y los pines de conexión para el sensor y los actuadores
SENSOR = adafruit_dht.DHT22(board.D2, use_pulseio = False)
ACTVEN = 3
ACTFOCO = 4
ACTRIEGO = 5

# Configuración de los pines utilizados
GPIO.setmode(GPIO.BCM)
GPIO.setup(ACTVEN, GPIO.OUT)
GPIO.setup(ACTFOCO, GPIO.OUT)
GPIO.setup(ACTRIEGO, GPIO.OUT)

# Definición de la función que medirá temperatura
def temp():
        try:
                # Se pide al sensor medir la temperatura y guardarla en la variable TEMPERATURA
                TEMPERATURA = SENSOR.temperature
                # Impresión de temperatura
                print ("\nTemperatura: ", TEMPERATURA, "°")
                # Utilización de if, elif y else para definir los límites de variación de la temperatura
                # En este caso se define el límite inferior
                if TEMPERATURA < 10:
                        # En caso que la temperatura se encuentre por debajo del límite se despliega el siguiente mensaje
                        print ("\n\tTemperatura baja, iniciando proceso de calentamiento, por favor espere...")
                        # Utilización de un ciclo while para elevar la temperatura, se le pide al programa que ejecute el
                        # código dentro del bloque while mientras la temperatura sea menor a 15 °C
                        while TEMPERATURA < 10:
                                # Debido a que se utilizará el sensor para saber si la temperatura ya se elevó, es
                                # necesario utilizar los bloques try y except por si el sensor presenta alguna falla a la
                                # hora de medir
                                try:
                                        # Se activa foco para calentar el invernadero y se mide nuevamente la
                                        # temperatura
                                        GPIO.output(ACTFOCO, GPIO.LOW)
                                        TEMPERATURA = SENSOR.temperature
                                # En este caso, ante un error se le pide al programa que lo ignore y pase a la siguiente
                                # línea
                                except RuntimeError as error:
                                        pass
                                # Se le pide al programa se detenga 10 segundos antes de volver al inicio del while para
                                # comparar la temperatura
                                time.sleep(10)
                                print ("\n\tCalentando, por favor espere...")
```

```
                        # Cuando la temperatura ya es mayor a 15 °C y el código salió del bloque while, se le pide al código
                        # que apague el foco de calentamiento e imprima nuevamente la temperatura junto a un aviso de que # la
                        temperatura se restableció
                        GPIO.output(ACTFOCO, GPIO.HIGH)
                        print ("\nTemperatura: ", TEMPERATURA, "°")
                        print ("\n\n\tLa temperatura se ha restablecido\n")
                # Definición del límite superior, se especifica que hacer en caso de que se sobrepase ese limite
                elif TEMPERATURA > 25:
                        # Se pide imprimir un aviso de advertencia sobre un calentamiento dentro del invernadero
                        print ("\n\tTemperatura muy alta, iniciando proceso de enfriamiento, por favor espere...")
                        # Utilización de un bloque while para definir qué hacer cuando se sobrepasan los 25 °C
                        while TEMPERATURA > 25:
                                try:
                                        # Activación de la ventilación y el riego
                                        GPIO.output(ACTVEN, GPIO.LOW)
                                        GPIO.output(ACTRIEGO, GPIO.LOW)
                                        # Nuevamente se pide la temperatura al sensor
                                        TEMPERATURA = SENSOR.temperature
                                except RuntimeError as error:
                                        pass
                                time.sleep(10)
                                print ("\n\tEnfriando, por favor espere...")
                         # Cuando la temperatura deja de ser mayor a 25 °C, se desactivan el riego y la ventilación
                        GPIO.output(ACTVEN, GPIO.HIGH)
                        GPIO.output(ACTRIEGO, GPIO.HIGH)
                        # Se imprime nuevamente la temperatura junto a un aviso de que se ha restablecido
                        print ("Temperatura: ", TEMPERATURA, "°")
                        print ("\n\n\tLa temperatura se ha restablecido\n")
                #Si la temperatura no es mayor a 25 °C, ni menor a 15 °C, entonces la temperatura se encuentra dentro del rango
                # deseado, en este caso se utilizó el bloque "else" para imprimir un aviso informando que la temperatura es
                # correcta
                else:
                        print ("\n\tLa temperatura es correcta")
        # Bloque except que especifica que hacer en caso de error la primera vez que se pide la temperatura en la función temp()
        except RuntimeError as error:
                print ("\n\n\tRecalculando valores de temperatura, por favor espere...\n")

# La función hum()  es bastante similar en cuanto a su funcionamiento, consta de bloques de comparación, en este caso de valores de
# humedad, muy parecidos a los de temperatura, por lo que los comentarios del código de esta función se omiten.
def hum():
        try:
                HUMEDAD = SENSOR.humidity
                print ("Humedad: ", HUMEDAD, "%")
                if HUMEDAD < 50:
                        print ("\n\tHumedad muy baja, iniciando riego, por favor espere...")
                        while HUMEDAD < 50:
                                try:
                                        GPIO.output(ACTRIEGO, GPIO.LOW)
                                        HUMEDAD = SENSOR.humidity
                                except RuntimeError as error:
                                        pass
                                time.sleep(10)
                                print ("\n\tReajustando humedad, por favor espere...")
                        GPIO.output(ACTRIEGO, GPIO.HIGH)
                        print ("Humedad: ", HUMEDAD, "%")
                        print ("\n\n\tLa humedad se ha reestablecido\n")
                elif HUMEDAD > 75:
                        print ("\n\tHumedad muy alta, activando ventilación, por favor espere...")
                        while HUMEDAD > 75:
                                try:
                                        GPIO.output(ACTVEN, GPIO.LOW)
                                        HUMEDAD = SENSOR.humidity
                                except RuntimeError as error:
                                        pass
                                time.sleep(10)
                                print ("\n\tReajustando humedad, por favor espere...")
                        GPIO.output(ACTVEN, GPIO.HIGH)
                        print ("Humedad: ", HUMEDAD, "%")
```

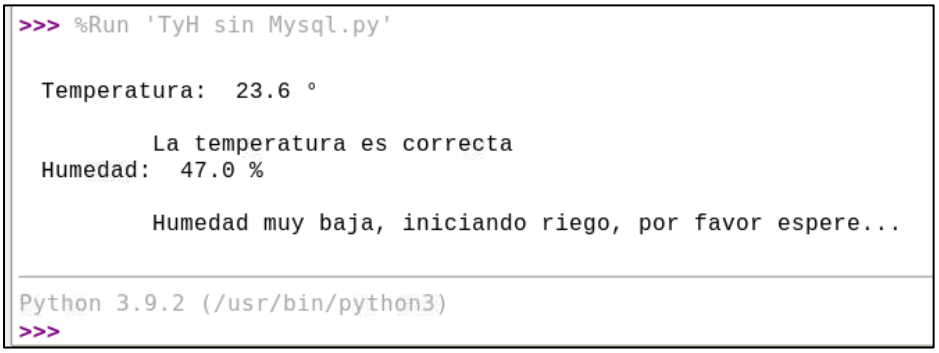
```
                print ("\n\n\tLa humedad se ha reestablecido\n")
        else:
                print ("\n\tEl porcentaje de humedad es correcto")
    except RuntimeError as error:
        print ("\n\n\tRecalculando valores de humedad, por favor espere...\n")


# En este bucle while es donde realmente inicia el programa, primero, se define un bucle infinito, para que funcione todo el tiempo, a
# menos que sea interrumpido por el usuario, dentro del bloque se llama primero apaga los actuadores, después hace un llamado a las
# funciones, espera un momento y regresa nuevamente al inicio, creando un el bucle infinito
while True:
        GPIO.output(ACTVEN, GPIO.HIGH)
        GPIO.output(ACTFOCO, GPIO.HIGH)
        GPIO.output(ACTRIEGO, GPIO.HIGH)
        temp()
        hum()
        time.sleep(10)
```

El código que acaba de ser presentado fue el código base para el sistema de medición de temperatura y humedad. Como lo especifican los comentarios, además de medir la temperatura y la humedad, también imprime avisos para saber si los valores se encuentran dentro de un rango definido por bucles *if*, *elif* y *else*, que fueron utilizados como bucles de comparación. El bucle *while* también fue utilizado como bloque de comparación, pero en este caso para indicarle al programa que hacer cuando los valores de los parámetros se salían de los valores establecidos previamente con *if*, *elif* y *else*.

El bucle *while* le indica al programa que active los actuadores de la ventilación, la calefacción o el riego mientras el valor del parámetro se encuentre fuera del rango, después que espere unos segundos, compare nuevamente los valores con los deseados, y si ya se encuentran dentro del rango apague los actuadores.

En el código se omitieron los comentarios de la función *hum()*, que mide la humedad, debido a que los comentarios hubiesen sido bastante similares a los de la función que mide la temperatura.

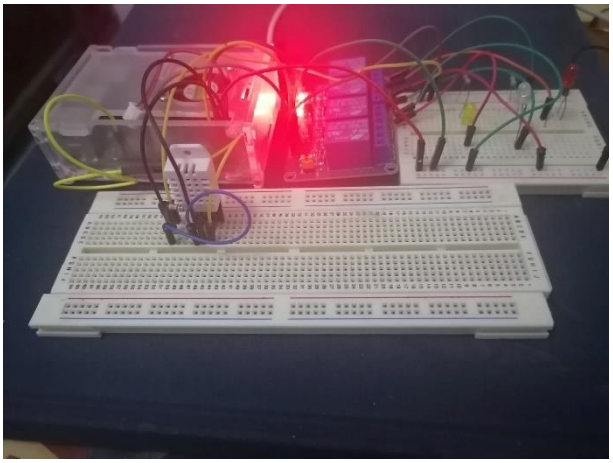
En la parte final del código se encuentra nuevamente un bucle *while*, en este caso un bucle infinito, ya que el propósito del proyecto es un sistema que se encuentre funcionando las 24 horas del día. En el bucle se observa que primero se solicita apagar los actuadores, después se llama a la función que mide, controla e imprime en pantalla el valor de la temperatura. Después, se hace el llamado a la función de la humedad que igualmente mide, controla e imprime los valores, en este caso de humedad. Al final se le pide al programa pausarse unos segundos y regresar nuevamente al inicio. Y todo este ciclo se repite infinitamente a menos que el usuario lo detenga.



10.3.1.7: Programa ejecutándose

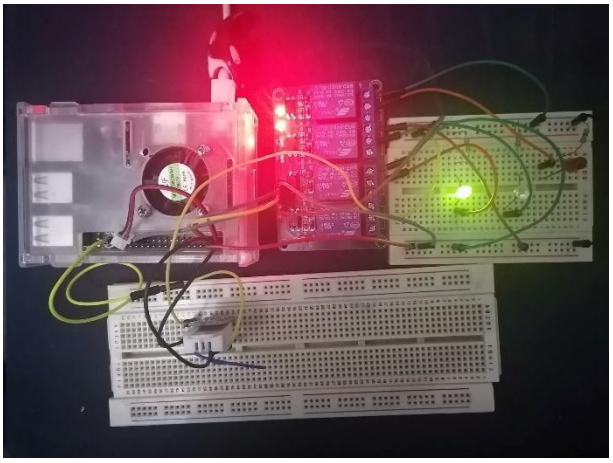
La imagen anterior muestra el código en ejecución, en este caso el ciclo *while* infinito se ejecutó una sola vez antes de ser detenido. La imagen muestra que la temperatura medida por el sensor se encontraba dentro del rango, mientras que el valor de la humedad se encontraba por debajo del valor deseado, por tal razón se desplegó el mensaje informativo.

En esta parte del diseño no se utilizaron todos los elementos necesarios para controlar la temperatura y la humedad debido a que el diseño aun no estaba completo y aun se buscaban errores o mejoras en el código. Los ventiladores, el foco de calentamiento y el sistema de riego fueron representados con LEDs de diferentes colores que se encendían o apagaban según el valor de los parámetros. Para cambiar el estado de los LEDs se utilizaron módulos de relés.



10.3.1.8: Pruebas con LEDs

En la imagen 10.3.1.8 se observa que el valor de la humedad es bajo por lo que, según el diseño del código, el LED que representaba al sistema de riego debía encenderse.



10.3.1.9: Activación del LED de riego.

La imagen anterior muestra que efectivamente solo se encendió el LED que representaba al sistema de riego. Con esta simulación y muchas otras más, se comprobó que el programa diseñado para medir humedad y temperatura trabajaba correctamente.

Partiendo del código anterior, y después de comprobar su funcionamiento, se agregaron las líneas faltantes para que el sistema de temperatura y humedad quedara completo. Se agregaron los segmentos necesarios para que los valores de humedad y temperatura se guardaran en la base de datos anteriormente configurada y se enviara un correo electrónico de alerta al usuario informando sobre cambios importantes en los valores de los parámetros. También, se hicieron las correcciones necesarias para tener los parámetros dentro de los rangos establecidos en capítulos anteriores. A continuación, se presenta una tabla informativa, en la que se vuelven a mencionar los rangos ideales para el invernadero, el correo y la contraseña utilizados para el envío de correos electrónicos y demás información que se tuvo en consideración al momento de terminar el código del programa:

Rango de temperatura ideal:	10 – 25 [°C]
Rango de humedad ideal:	50 – 75 [%]
Correo emisor:	invernaderoiotsagme@outlook.com
Contraseña del correo:	TesisIoT2021
Servidor SMTP para Outlook:	@smtp.office365.com
Puerto SMTP:	587

Agregando las alertas por correo electrónico, los rangos de humedad y temperatura, e indicando al programa guardar dichos valores en la tabla *DHT22* de la base de datos *Invernadero*, previamente creada, el código final del sistema de humedad y temperatura fue:

```
import adafruit_dht
import board
import time
import RPi.GPIO as GPIO
# Se agregaron dos nuevas bibliotecas, mysql.connector, que conecta Python con MySQL y smtplib que envía correos electrónicos a
# través del protocolo SMTP
import mysql.connector
import smtplib

SENSOR = adafruit_dht.DHT22(board.D2, use_pulseio = False)
# Para poder manipular la base de datos es necesaria la siguiente línea, en la que se especifica la información para comprobar la identidad
mibd = mysql.connector.connect(host = "localhost", user = "pi", passwd = "TesisIoT2021", database = "Invernadero")
ACTVEN = 3
ACTFOCO = 4
ACTRIEGO = 5

GPIO.setmode(GPIO.BCM)
GPIO.setup(ACTVEN, GPIO.OUT)
GPIO.setup(ACTFOCO, GPIO.OUT)
GPIO.setup(ACTRIEGO, GPIO.OUT)

# Definición de las funciones que serán utilizadas
def temp():
        try:
                TEMPERATURA = SENSOR.temperature
                print ("\nTemperatura: ", TEMPERATURA, "°")
                if TEMPERATURA < 10:
                        print ("\n\tTemperatura baja, iniciando proceso de calentamiento, por favor espere...")
                        # En la variable mensaje se escribe la cadena de texto que se enviará en el cuerpo del correo
                        mensaje = """
                                ESTE ES UN MENSAJE DE ALERTA:

                                LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION, SE
                                DESCRIBE LA FALLA:

                                PARAMETRO AFECTADO: Temperatura
                                ESTADO: Muy baja
                                ACCIONES TOMADAS: Sistema de calefaccion activado
```

```
                    POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

                    GRACIAS!

                    """
            # Especificación de la estructura del correo, se observa que se pide agregar asunto y mensaje
            mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
            # Es necesario agregar la información del servidor SMTP del correo utilizado, en este caso para
            # correos Outlook los datos son:
            server = smtplib.SMTP("smtp.office365.com", 587)
            # Se inicializa el servidor
            server.starttls()
            # Se inicia sesión de correo Outlook
            server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
            # Envío del correo con la alerta de baja temperatura
            server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
            # Por seguridad, es recomendable cerrar la sesión después de enviar el correo
            server.quit()
            # Después de enviar el correo se inicia el proceso de enfriamiento explicado anteriormente
            while TEMPERATURA < 10:
                    try:
                            GPIO.output(ACTFOCO, GPIO.LOW)
                            TEMPERATURA = SENSOR.temperature
                    except RuntimeError as error:
                            pass
                    time.sleep(120)
                    print ("\n\tCalentando, por favor espere...")
            GPIO.output(ACTFOCO, GPIO.HIGH)
            print ("\nTemperatura: ", TEMPERATURA, "°")
            print ("\n\n\tLa temperatura se ha reestablecido\n")
            # Una vez restablecida la temperatura, nuevamente se envía correo, en este caso para informar que la
            # temperatura regresó a la normalidad
            mensaje = """
                    ESTE ES UN  MENSAJE DE ACTUALIZACION:

                    LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

                    GRACIAS!!

                    """
            mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
            server = smtplib.SMTP("smtp.office365.com", 587)
            server.starttls()
            server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
            server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
            server.quit()
    elif TEMPERATURA > 25:
            print ("\n\tTemperatura muy alta, iniciando proceso de enfriamiento, por favor espere...")
            # Preparación del correo que será enviado para notificar alta temperatura
            mensaje = """
                    ESTE ES UN MENSAJE DE ALERTA:

                    LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION, SE
                    DESCRIBE LA FALLA:

                    PARAMETRO AFECTADO: Temperatura
                    ESTADO: Muy alta
                    ACCIONES TOMADAS: Sistema de enfriamiento activado

                    POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

                    GRACIAS!

                    """
            mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
            server = smtplib.SMTP("smtp.office365.com", 587)
            server.starttls()
```

```
                server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
                server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
                server.quit()
                while TEMPERATURA > 25:
                        try:
                                GPIO.output(ACTVEN, GPIO.LOW)
                                GPIO.output(ACTRIEGO, GPIO.LOW)
                                TEMPERATURA = SENSOR.temperature
                        except RuntimeError as error:
                                pass
                        time.sleep(60)
                        print ("\n\tEnfriando, por favor espere...")
                GPIO.output(ACTVEN, GPIO.HIGH)
                GPIO.output(ACTRIEGO, GPIO.HIGH)
                print ("Temperatura: ", TEMPERATURA, "°")
                print ("\n\n\tLa temperatura se ha reestablecido\n")
                # Envío de correo para informar que la temperatura se ha normalizado
                mensaje = """
                        ESTE ES UN  MENSAJE DE ACTUALIZACION:

                        LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

                        GRACIAS!!

                        """
                mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
                server = smtplib.SMTP("smtp.office365.com", 587)
                server.starttls()
                server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
                server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
                server.quit()
        else:
                print ("\n\tLa temperatura es correcta")
        # Después de comprobar que la temperatura se encuentra dentro del rango, su valor se devuelve para utilizarlo en
        # el código principal
        return TEMPERATURA
except RuntimeError as error:
        pass

def hum():
        try:
                HUMEDAD = SENSOR.humidity
                print ("\nHumedad: ", HUMEDAD, "%")
                if HUMEDAD < 50:
                        print ("\n\tHumedad muy baja, iniciando riego, por favor espere...")
                        mensaje = """
                                ESTE ES UN MENSAJE DE ALERTA:

                                LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION SE
                                DESCRIBE LA FALLA:

                                PARAMETRO AFECTADO:Humedad
                                ESTADO: Muy baja
                                ACCIONES TOMADAS: Sistema de riego activado

                                POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

                                GRACIAS!

                                """
                        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
                        server = smtplib.SMTP("smtp.office365.com", 587)
                        server.starttls()
                        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
                        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
                        server.quit()
                        while HUMEDAD < 50:
                                try:
```

```
                                        GPIO.output(ACTRIEGO, GPIO.LOW)
                                        HUMEDAD = SENSOR.humidity
                                except RuntimeError as error:
                                        pass
                                time.sleep(10)
                                print ("\n\tReajustando humedad, por favor espere...")
                        GPIO.output(ACTRIEGO, GPIO.HIGH)
                        print ("Humedad: ", HUMEDAD, "%")
                        print ("\n\n\tLa humedad se ha reestablecido\n")
                        mensaje = """
                                ESTE ES UN  MENSAJE DE ACTUALIZACION:

                                LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

                                GRACIAS!!

                                """
                        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
                        server = smtplib.SMTP("smtp.office365.com", 587)
                        server.starttls()
                        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
                        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
                        server.quit()
        elif HUMEDAD > 75:
                        print ("\n\tHumedad muy alta, activando ventilación, por favor espere...")
                        mensaje = """
                                ESTE ES UN MENSAJE DE ALERTA:

                                LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION SE
                                DESCRIBE LA FALLA:

                                PARAMETRO AFECTADO: Humedad
                                ESTADO: Muy alta
                                ACCIONES TOMADAS: Sistema de ventilacion activado

                                POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

                                GRACIAS!

                                """
                        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
                        server = smtplib.SMTP("smtp.office365.com", 587)
                        server.starttls()
                        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
                        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
                        server.quit()
                        while HUMEDAD > 75:
                                try:
                                        GPIO.output(ACTVEN, GPIO.LOW)
                                        HUMEDAD = SENSOR.humidity
                                except RuntimeError as error:
                                        pass
                                time.sleep(180)
                                print ("\n\tReajustando humedad, por favor espere...")
                        GPIO.output(ACTVEN, GPIO.HIGH)
                        print ("Humedad: ", HUMEDAD, "%")
                        print ("\n\n\tLa humedad se ha reestablecido\n")
                        mensaje = """
                                ESTE ES UN  MENSAJE DE ACTUALIZACION:

                                LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

                                GRACIAS!!

                                """
                        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
                        server = smtplib.SMTP("smtp.office365.com", 587)
                        server.starttls()
```

```
                        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
                        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
                        server.quit()
                else:
                        print ("\n\tEl porcentaje de humedad es correcto")
                return HUMEDAD
        except RuntimeError as error:
                pass

# Código principal
while True:

        GPIO.output(ACTVEN, GPIO.HIGH)
        GPIO.output(ACTFOCO, GPIO.HIGH)
        GPIO.output(ACTRIEGO, GPIO.HIGH)
        # Se define el asunto de los correos, en este caso a través de una variable global debido a que será el mismo para todos los
        # correos
        asunto = "Notificacion Invernadero IoT"

        # Llamado de las funciones temp() y hum(), se pide que los valores retornados se guarden en las variables
        # TEMPERATURA y HUMEDAD, respectivamente
        TEMPERATURA = temp()
        HUMEDAD = hum()
        # También se pide la fecha y hora del servidor
        FECHA = time.strftime("%y/%m/%d")
        HORA = time.strftime("%H:%M:%S")

        # Se crea el cursor que manipulará la tabla en MySQL
        mycursor = mibd.cursor()
        # Comienza la configuración para el llenado de la tabla DTH22, se le pide al cursor que guarde datos en las variables de
        # la tabla
        sql = "INSERT INTO DHT22(Fecha, Hora, Temperatura, Humedad) VALUES (%s, %s, %s, %s)"
        # Se especifican los datos que serán guardados en la tabla, así como el orden en los que se guardarán
        val = (FECHA, HORA, TEMPERATURA, HUMEDAD)
        # La siguiente línea hace que comience el llenado de la tabla
        mycursor.execute(sql, val)
        # Después de ingresar los datos a la base de datos se hace un borrado de la memoria de la RBPi
        mibd.commit()

        # Cuando se tienen todos los parámetros en orden, se le pide al sistema que envié un correo informando el estado
        mensaje = """
                ESTE ES UN MENSAJE DE NOTIFICACION:

                TODOS LOS PARAMETROS SE ENCUENTRAN EN ORDEN!

                """
        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
        server = smtplib.SMTP("smtp.office365.com", 587)
        server.starttls()
        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
        server.quit()

        # El código dormirá 900 segundos (15 minutos) antes de volver a verificar el estado del invernadero
        time.sleep(900)
```

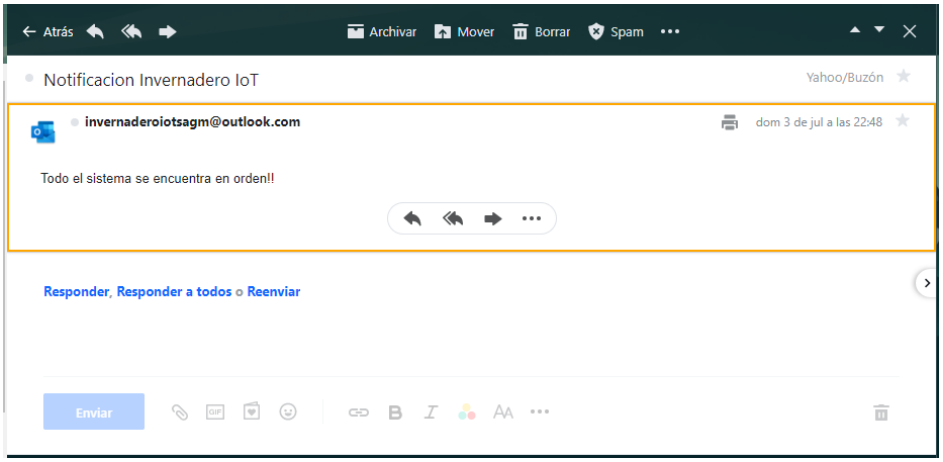
Para este caso nuevamente solo se comentó la parte del código de la función de temperatura debido a que la función de humedad es bastante similar, la única diferencia es que miden parámetros diferentes.

Se observa que el código presentado al inicio del subtema se complementó con las líneas necesarias para enviar correos electrónicos a través del protocolo SMTP. Se decidió utilizar un correo Outlook por facilidad debido a que Gmail y Yahoo! cuentan con altos protocolos de seguridad que bloqueaban el inicio de sesión desde el programa no verificados por sus sistemas de seguridad.

El programa diseñado envía notificaciones vía correo electrónico para informar cuando un parámetro se encuentra fuera del rango definido, en el correo también informa de las acciones a tomar. Una vez que el parámetro vuelve a la normalidad se envía un nuevo correo notificándolo. Otra de las notificaciones que se envían por correo es cuando no se tienen problemas, en este caso avisando que el invernadero opera correctamente.

El programa fue configurado para monitorear humedad y temperatura cada 900 segundos, lo que equivale a 15 minutos.

El funcionamiento de los actuadores fue demostrado anteriormente, por lo que a continuación, solo se mostrará que las notificaciones vía correo electrónico funcionan correctamente. También, se mostrará que la base de datos comenzó a guardar información en la tabla *DTH22*.



10.3.1.10: Notificación vía correo electrónico

En la imagen 10.3.1.11 se muestra que la tabla *DTH22* comenzó a guardar información. Se observa que las cuatro columnas creadas contienen datos con diferentes fechas y en diferentes horas.

Cada renglón de la tabla fue creado automáticamente en cada ciclo *while*, y seguirá agregándose un nuevo renglón cada vez que se ejecute el sistema de automatización de Temperatura y Humedad. Esta información será enviada vía correo electrónico y mostrada en una página web de la que más adelante se hablará.

+ Opciones

Fecha	Hora	Temperatura	Humedad
2022-07-03	16:31:04	24.6	42.2
2022-07-03	16:32:57	24.7	41.3
2022-07-03	16:33:08	24.7	42.1
2022-07-03	16:35:35	24.7	40.5
2022-07-03	16:35:46	24.8	42
2022-07-03	22:25:58	26.1	39.5
2022-07-03	22:27:24	26.1	40.7
2022-07-03	22:27:56	26	39
2022-07-03	22:31:18	26.1	41.1

10.3.1.11: Tabla DTH22 con valores

Después de comprobar el funcionamiento de la base de datos en MySQL el siguiente paso fue mostrar el contenido en una página Web. En este caso, para el diseño de la página se utilizó el lenguaje php junto a HTML debido a la necesidad de utilizar paginas dinámicas para cambiar la información automáticamente cada vez que se tuviera un valor nuevo.

El código se creó desde la ventana de comandos de la Raspberry con los permisos de administrador *sudo* para poder utilizar la tarjeta como el servidor de la página Web.

Primero se ejecutó el comando:

$$sudo\ nado\ /var/www/html/HyTFinal.php$$

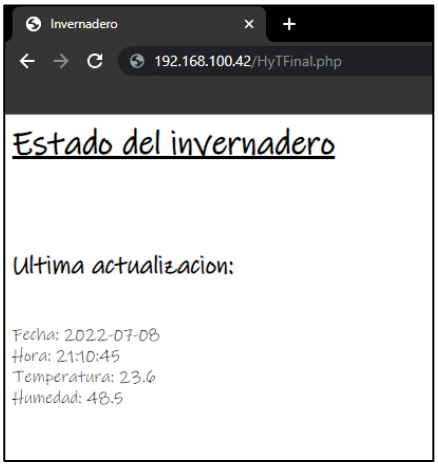
Que creo un archivo HTML llamado *HyTFinal.php* en la ruta */var/www/html/*, la cual solo puede ser modificada por el administrador. Después de ejecutar el comando, en automático se abrió el archivo con el nombre ya mencionado y se escribió el código php:

```
<html>
        <head>
                // Nombre que tendrá la pestaña de la página
                <title> Invernadero IoT </title>
                <meta charset="utf-8">
        </head>
    <body>
        <?php
            // Se crea la conexión entre php y MySQL, en la que se colocan el host, el usuario y la contraseña
            $link = mysqli_connect("localhost", "pi", "TesisIoT2021");
            // La siguiente línea especifica la base de datos en la que se va a trabajar
            mysqli_select_db($link, "Invernadero");
            // Después de especificar la base de datos es necesario crear un puntero para manipular la tabla
            $result = mysqli_query($link, "SELECT * FROM DHT22");
            // Se le pidió al código el número de renglones guardados en la tabla, más adelante se especifica el motivo
            $renglones = mysqli_num_rows($result);
            // Posición del último valor guardado
            $ultima = $renglones-1;
            // Se dirige el puntero al último renglón de la tabla
            mysqli_data_seek($result, $ultima);
            // El ultimo renglón se extrae en un arreglo
            $extraido = mysqli_fetch_array($result);
            // Inicia la impresión de texto en la página web
            echo "<h1><u>Estado del invernadero</u></h1><br>";
            echo "<br><h2>Ultima actualización:</h2><br>";
            // Se pide que del arreglo extraído se imprima el elemento llamado Fecha
            echo "Fecha: ".$extraido["Fecha"]."<br>";
            // Después se imprimen los elementos Hora, Temperatura y Humedad sucesivamente
            echo "Hora: ".$extraido["Hora"]."<br>";
            echo "Temperatura: ".$extraido["Temperatura"]."<br>";
            echo "Humedad: ".$extraido["Humedad"]."<br>";
            // Se borra memoria
            mysql_free_result($result);
            // Por seguridad se cierra sesión en MySQL
            mysql_close($link);
        ?>
    </body>
</html>
```

El código inicia indicando el nombre de la página Web, en este caso *Invernadero IoT*, posteriormente se pasó a la programación PHP donde se inició sesión a través de una variable llamada *$link*, que sirvió como enlace entre PHP y MySQL, en dicha variable se especificó la información necesaria para iniciar sesión en MySQL. Después de iniciar sesión se tuvo que especificar la base de datos y la tabla de las cuales obtendríamos la información para la página.

Para poder trabajar con la información de una tabla es necesario crear un puntero que navegará por toda la información, en este caso se creó el puntero de nombre *$result*, que al inicio apuntó sobre toda la tabla *DTH22*. La idea que se tuvo para la página fue que se mostrara solo el ultimo valor leído por el sensor, para este caso era necesario que el puntero solo seleccionara el último renglón. Debido a que con cada ciclo *while* se va agregando un nuevo renglón jamás será el mismo número de renglón el que se mostrará en la página, por lo que era necesario saber la posición del último renglón antes de imprimir la información. La manera de solucionar este problema fue que primero el código le preguntara a MySQL con cuantos renglones contaba la tabla *DTH22*, esto se hizo con la variable *$renglones,* y recordando que la posición de una tabla siempre inicia desde el 0, la posición del último renglón sería *$renglones – 1*, que se guardó en la variable *$ultima*. Después de calcular la última posición, el puntero fue enviado al último renglón y su información se extrajo en un arreglo llamado *$extraido*. Debido a que el renglón ya pertenecía a una tabla, la posición de cada elemento del arreglo conservó el nombre de su columna, por tal motivo se pudo imprimir la información guardada en las posiciones *Fecha*, *Hora*, *Temperatura* y *Humedad*.

La siguiente imagen muestra el resultado al ingresar a la página *192.168.100.42/HyTFinal.php* desde el navegador de cualquier dispositivo conectado a la misma red.



10.3.1.12: Página Web

Al comprobar el funcionamiento del programa que mide Temperatura y Humedad, de la base de datos, de la página web y de realizar simulaciones con LEDs, el último paso consistió en probar el sistema nuevamente, pero ahora con los elementos reales, es decir, los ventiladores, el riego y la tira LED de calentamiento. Primero se probó sin instalar en el invernadero, de esta manera se comprobaría su funcionamiento y sería más fácil arreglar cualquier desperfecto antes de montar todos los elementos.

10.3.1.13: Prueba del Sistema de Humedad y Temperatura

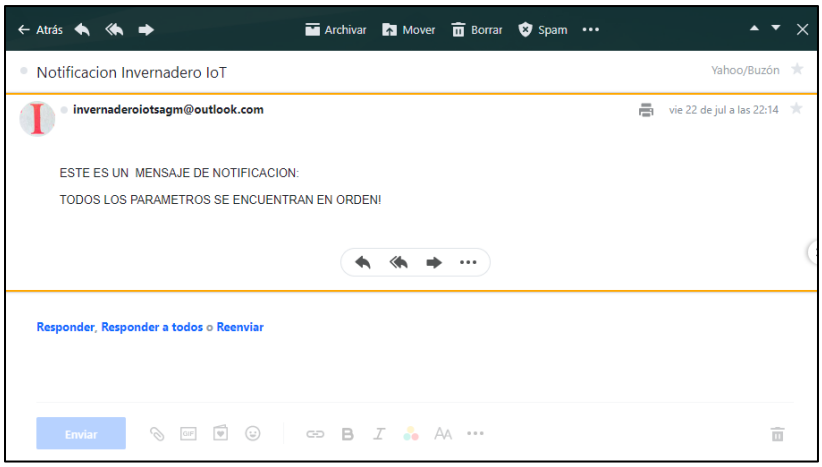
En la parte superior de la imagen se observan los dos ventiladores, en la parte inferior izquierda se observa la bomba que activará el riego, y en la parte inferior derecha se observa el carrete que contiene la tira LED de calentamiento e iluminación. Al centro se observa la Raspberry, y de su lado derecho, de color azul, el módulo de relés.

Para realizar pruebas, dentro de la habitación utilizada se tenía una humedad muy por debajo de la establecida para el invernadero, por lo que se modificó el rango de temperatura solo para la realización de las pruebas. En este caso el nuevo rango de humedad fue entre el 35% y el 70%. Cuando se puso en funcionamiento el programa del sistema se mostró el siguiente mensaje:

```
Temperatura:  24.2 °

        La temperatura es correcta

Humedad:  37.0 %

        El porcentaje de humedad es correcto
```

10.3.1.14: Mensaje que indica que el invernadero de encuentra trabajando en orden

Según el diseño del código, si la temperatura y la humedad se encontraban dentro de los rangos definidos se tenía que notificar a través de un correo. El correo recibido fue el siguiente:



10.3.1.15: Mensaje de notificación de parámetros correctos

Después, para comprobar el funcionamiento se dejó ejecutando el programa por un tiempo y se comprobó la existencia de errores que detenían la ejecución del programa. Los errores eran lanzados cuando el sensor DTH22 no era capaz de leer un valor de humedad o de temperatura. En este caso, la variable *HUMEDAD* o *TEMPERATURA* no almacenaba información y al momento de que Python le enviaba la información a MySQL para guardarla en la base de datos se arrojaba un error en el que se mencionaba que MySQL no era capaz de guardar variables *Null*, es decir, variables vacías. Otro problema que también arrojó el programa fue que en muy pocas ocasiones el sensor realizaba la medición, pero no retornaba el valor, por lo que el programa no era capaz de saber si el valor de las variables *TEMPERATURA* y *HUMEDAD* era un entero, una cadena, un booleano, etc., por lo que le asignaba automáticamente un valor *None*, y esto creaba conflictos debido a que en los bucles de comparación no se podía trabajar con un tipo *None* y un tipo *int*. Este error también detenía la ejecución del programa.

Para solucionar estos dos errores lo que se hizo fue asignarles valores a las variables en caso de que fueran variables *Null* o *None*, esto a través de ciclos *if*. Además, para prevenir futuros errores se agregó una excepción final para que el código no se detenga en caso de falla. Por lo que al final el código del sistema de Temperatura y Humedad quedo de la siguiente manera:

```
import adafruit_dht
import board
import time
import RPi.GPIO as GPIO
import mysql.connector
import smtplib

SENSOR = adafruit_dht.DHT22(board.D2, use_pulseio = False)
mibd = mysql.connector.connect(host = "localhost", user = "pi", passwd = "TesisIoT2021", database = "Invernadero")
ACTVEN = 3
ACTFOCO = 4
ACTRIEGO = 5

GPIO.setmode(GPIO.BCM)
GPIO.setup(ACTVEN, GPIO.OUT)
GPIO.setup(ACTFOCO, GPIO.OUT)
GPIO.setup(ACTRIEGO, GPIO.OUT)

def temp():
    try:
        TEMPERATURA = SENSOR.temperature
        # Se asigna un valor de 20 °C cuando el sensor no retorna valores y se tiene un None
        if TEMPERATURA is None:
            TEMPERATURA = 20
        print ("\nTemperatura: ", TEMPERATURA, "°")
        if TEMPERATURA < 10:
            print ("\n\tTemperatura baja, iniciando proceso de calentamiento, por favor espere...")
            mensaje = """
                ESTE ES UN  MENSAJE DE ALERTA:

                LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION, SE DESCRIBE LA FALLA:

                PARAMETRO AFECTADO: Temperatura
                ESTADO: Muy baja
                ACCIONES TOMADAS:Sistema de calefaccion activado

                POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

                GRACIAS!

                """
            mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
            server = smtplib.SMTP("smtp.office365.com", 587)
            server.starttls()
            server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
            server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
            server.quit()
            while TEMPERATURA < 10:
                try:
                    GPIO.output(ACTFOCO, GPIO.LOW)
                    TEMPERATURA = SENSOR.temperature
                except RuntimeError as error:
                    pass
                time.sleep(120)
                print ("\n\tCalentando, por favor espere...")
            GPIO.output(ACTFOCO, GPIO.HIGH)
            print ("\nTemperatura: ", TEMPERATURA, "°")
            print ("\n\n\tLa temperatura se ha reestablecido\n")
            mensaje = """
                ESTE ES UN  MENSAJE DE ACTUALIZACION:
```

```
            LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

            GRACIAS!!

            """
        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
        server = smtplib.SMTP("smtp.office365.com", 587)
        server.starttls()
        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
        server.quit()
    elif TEMPERATURA > 25:
        print ("\n\tTemperatura muy alta, iniciando proceso de enfriamiento, por favor espere...")
        mensaje = """
            ESTE ES UN  MENSAJE DE ALERTA:

            LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION, SE DESCRIBE LA FALLA:

            PARAMETRO AFECTADO: Temperatura
            ESTADO: Muy alta
            ACCIONES TOMADAS: Sistema de enfriamiento activado

            POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

            GRACIAS!

            """
        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
        server = smtplib.SMTP("smtp.office365.com", 587)
        server.starttls()
        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
        server.quit()
        while TEMPERATURA > 25:
            try:
                GPIO.output(ACTVEN, GPIO.LOW)
                GPIO.output(ACTRIEGO, GPIO.LOW)
                TEMPERATURA = SENSOR.temperature
            except RuntimeError as error:
                pass
            time.sleep(10)
            print ("\n\tEnfriando, por favor espere...")
        GPIO.output(ACTVEN, GPIO.HIGH)
        GPIO.output(ACTRIEGO, GPIO.HIGH)
        print ("Temperatura: ", TEMPERATURA, "°")
        print ("\n\n\tLa temperatura se ha reestablecido\n")
        mensaje = """
            ESTE ES UN  MENSAJE DE ACTUALIZACION:

            LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

            GRACIAS!!

            """
        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
        server = smtplib.SMTP("smtp.office365.com", 587)
        server.starttls()
        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
        server.quit()
    else:
        print ("\n\tLa temperatura es correcta")
    return TEMPERATURA
  except RuntimeError as error:
    pass

def hum():
  try:
```

```
HUMEDAD = SENSOR.humidity
# Se asigna un valor de 60 % a la  humedad cuando el sensor no retorna valor y se tiene un None
if HUMEDAD is None:
    HUMEDAD = 60
print ("\nHumedad: ", HUMEDAD, "%")
if HUMEDAD < 50:
    print ("\n\tHumedad muy baja, iniciando riego, por favor espere...")
    mensaje = """
        ESTE ES UN  MENSAJE DE ALERTA:

        LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION SE DESCRIBE LA FALLA:

        PARAMETRO AFECTADO:Humedad
        ESTADO: Muy baja
        ACCIONES TOMADAS: Sistema de riego activado

        POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

        GRACIAS!

        """
    mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
    server = smtplib.SMTP("smtp.office365.com", 587)
    server.starttls()
    server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
    server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
    server.quit()
    while HUMEDAD < 50:
        try:
            GPIO.output(ACTRIEGO, GPIO.LOW)
            HUMEDAD = SENSOR.humidity
        except RuntimeError as error:
            pass
        time.sleep(10)
        print ("\n\tReajustando humedad, por favor espere...")
    GPIO.output(ACTRIEGO, GPIO.HIGH)
    print ("Humedad: ", HUMEDAD, "%")
    print ("\n\n\tLa humedad se ha reestablecido\n")
    mensaje = """
        ESTE ES UN  MENSAJE DE ACTUALIZACION:

        LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

        GRACIAS!!

        """
    mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
    server = smtplib.SMTP("smtp.office365.com", 587)
    server.starttls()
    server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
    server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
    server.quit()
elif HUMEDAD > 75:
    print ("\n\tHumedad muy alta, activando ventilación, por favor espere...")
    mensaje = """
        ESTE ES UN  MENSAJE DE ALERTA:

        LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION SE DESCRIBE LA FALLA:

        PARAMETRO AFECTADO: Humedad
        ESTADO: Muy alta
        ACCIONES TOMADAS: Sistema de ventilacion activado

        POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

        GRACIAS!

        """
```

```
            mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
            server = smtplib.SMTP("smtp.oficce365.com", 587)
            server.starttls()
            server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
            server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
            server.quit()
            while HUMEDAD > 75:
               try:
                  GPIO.output(ACTVEN, GPIO.LOW)
                  HUMEDAD = SENSOR.humidity
               except RuntimeError as error:
                  pass
               time.sleep(10)
               print ("\n\tReajustando humedad, por favor espere...")
            GPIO.output(ACTVEN, GPIO.HIGH)
            print ("Humedad: ", HUMEDAD, "%")
            print ("\n\n\tLa humedad se ha reestablecido\n")
            mensaje = """
               ESTE ES UN  MENSAJE DE ACTUALIZACION:

               LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

               GRACIAS!!

               """
            mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
            server = smtplib.SMTP("smtp.office365.com", 587)
            server.starttls()
            server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
            server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
            server.quit()
         else:
            print ("\n\tEl porcentaje de humedad es correcto")
         return HUMEDAD
      except RuntimeError as error:
         pass

while True:

            try:
               GPIO.output(ACTVEN, GPIO.HIGH)
               GPIO.output(ACTFOCO, GPIO.HIGH)
               GPIO.output(ACTRIEGO, GPIO.HIGH)
               asunto = "Notificacion Invernadero IoT"

               TEMPERATURA = temp()
               # Se asigna un valor de 20 °C cuando la variable está vacía, Null, y no se puede guardar en MySQL
               if TEMPERATURA is None:
                  TEMPERATURA = 20
               HUMEDAD = hum()
               # Se asigna un valor de 40 % cuando la variable está vacía, Null, y no se puede guardar en MySQL
               if HUMEDAD is None:
                  HUMEDAD = 60
               FECHA = time.strftime("%y/%m/%d")
               HORA = time.strftime("%H:%M:%S")

               mycursor = mibd.cursor()
               sql = "INSERT INTO DHT22(Fecha, Hora, Temperatura, Humedad) VALUES (%s, %s, %s, %s)"
               val = (FECHA, HORA, TEMPERATURA, HUMEDAD)
               mycursor.execute(sql, val)
               mibd.commit()

               mensaje = """
                  ESTE ES UN  MENSAJE DE NOTIFICACION:

                  TODOS LOS PARAMETROS SE ENCUENTRAN EN ORDEN!

                  """
```

```
        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
        server = smtplib.SMTP("smtp.office365.com", 587)
        server.starttls()
        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
        server.quit()

        time.sleep(900)

    except OSError as error:
        pass
```

En este caso se resaltó la parte del código que se agregó para resolver los errores encontrados. Después de agregar las líneas de código nuevamente se ejecutó por un tiempo el programa para comprobar que resolvían los problemas y monitorear nuevamente en busca de errores al momento de la ejecución.

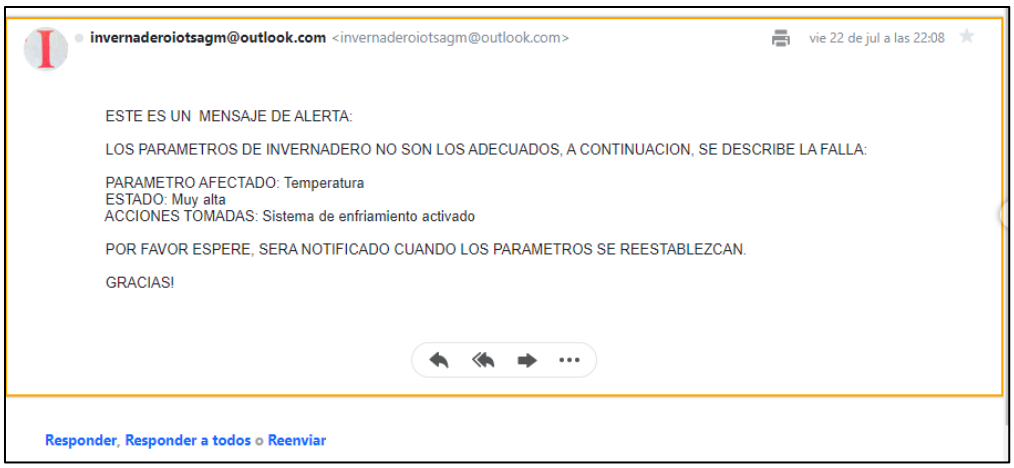
Durante el monitoreo del funcionamiento del sistema, se hicieron diferentes tipos de prueba. La primera prueba consistió en generar calor con una secadora de pelo para comprobar que el sistema enfriaba correctamente.

10.3.1.16: Prueba del sistema de enfriamiento

Después de unos segundos el sensor comenzó a notar el aumento de temperatura, y la Raspberry comenzó el proceso de enfriamiento. Primero se notificó en a través de un mensaje en la pantalla y por correo electrónico.

```
Temperatura:  28.9 °
        Temperatura muy alta, iniciando proceso de enfriamiento, por favor espere...
        Enfriando, por favor espere...
        Enfriando, por favor espere...
        Enfriando, por favor espere...
        Enfriando, por favor espere...
        Enfriando, por favor espere...
```

10.3.1.17: Mensaje sobrecalentamiento

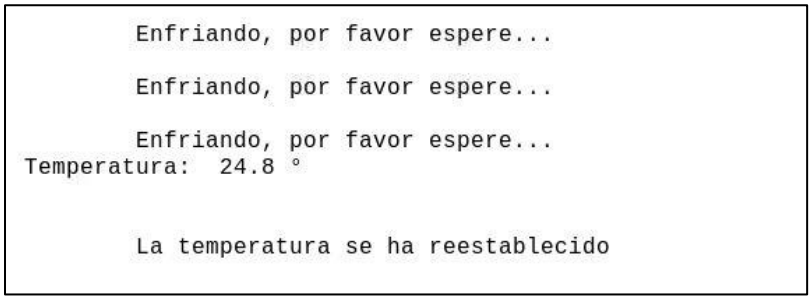


10.3.1.18: Correo de notificación de alta temperatura

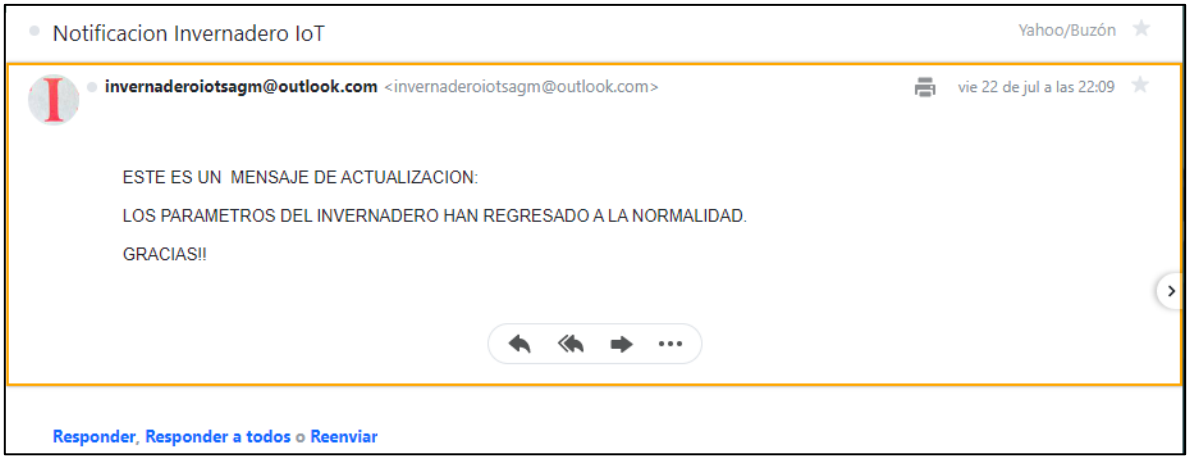
También, como está previsto, se activaron la ventilación y la bomba de riego.

10.3.1.19: Inicio de enfriamiento

El sistema tardó alrededor de tres minutos en enfriar el ambiente. Cuando la temperatura regreso al rango adecuado se desactivaron automáticamente los ventiladores y la bomba, y se notificó por correo electrónico.



10.3.1.20: Mensaje después de ajustar la Temperatura

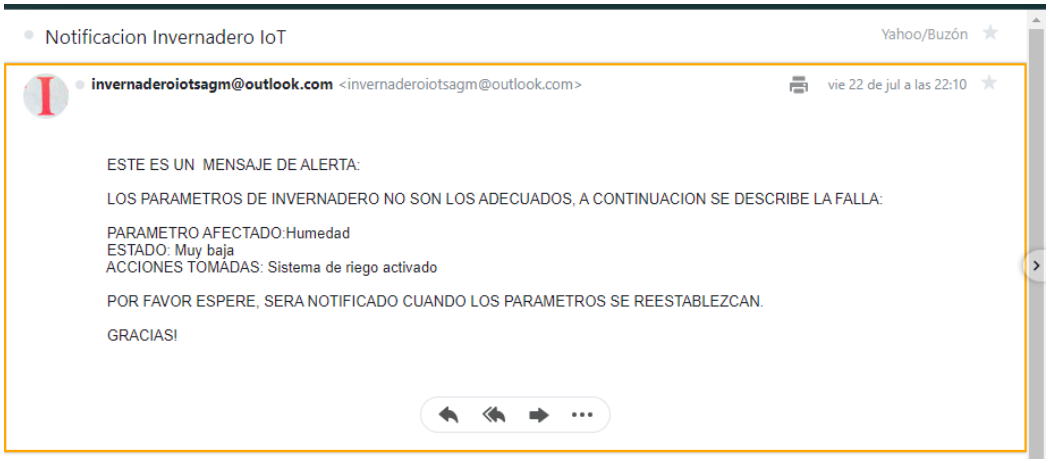


10.3.1.21: Notificación de ajuste de Temperatura vía correo electrónico

El ajuste de temperatura provocó un desajuste en los niveles de humedad, por lo que se alertó igualmente vía mensaje en pantalla y correo electrónico.

```
Humedad:  34.9 %

        Humedad muy baja, iniciando riego, por favor espere...

        Reajustando humedad, por favor espere...
Humedad:  35.2 %


        La humedad se ha reestablecido
```

10.3.1.22: Mensaje de humedad baja



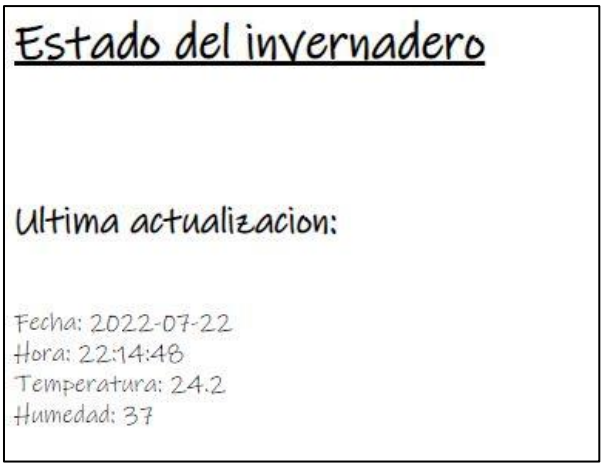
10.3.1.23: Alerta por correo electrónico sobre humedad baja

Aunque el ajuste de la humedad fue casi inmediato, sirvió para comprobar que el sistema de aumento de humedad funcionaba correctamente debido a que la bomba de riego se activó al detectar la baja humedad.

10.3.1.24: Activación de bomba de riego

Igualmente, cuando se corrigió la humedad se envió un correo electrónico de aviso, en este caso fue un correo similar al mostrado en la imagen 10.3.1.21.

En la última ejecución del programa se esperó a que los parámetros de humedad y temperatura estuvieran en orden. De esta manera serían enviados a la base de datos MySQL y desplegados en la página Web.



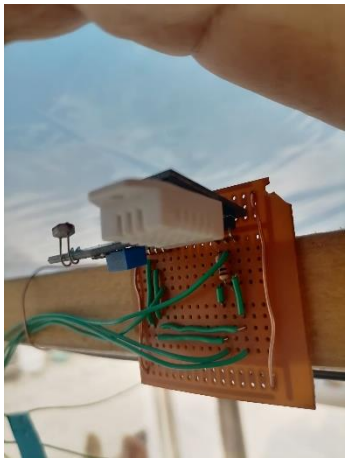
10.3.1.25: Página Web

Así mismo, si todo estaba en orden, se tenía que notificar por correo electrónico. El correo recibido fue el siguiente:

10.3.1.26: Correo que informa que todos los parámetros de encuentran en orden

Con la ejecución del sistema por algunas horas se pudo comprobar que los dos errores encontrados fueron resueltos, y esto, junto con las pruebas realizadas se pudo comprobar que el sistema de control de Temperatura y Humedad ya se encontraba listo para ser instalado en el invernadero. Por lo que a continuación se presentar las fotografías de la instalación:

Sensor de humedad y temperatura:



10.3.1.27: Sensor DTH22

El sensor fue colocado en una placa perforada junto al sensor LDR, ambos sensores se encuentran en la parte superior del invernadero. Para facilitar la instalación y reducir el número de conexiones se conectaron ambos sensores a la misma fuente de alimentación de 5[V].

Sistema de ventilación:

10.3.1.28: Ventilador para ingreso de aire

10.3.1.29: Ventilador para salida de aire

En las imágenes anteriores se pueden observar los dos ventiladores utilizados, el ventilador para el ingreso de aire frio se encuentra colocado en la parte inferior del invernadero, mientras el ventilador para la extracción de aire caliente se encuentra en la parte superior, esto debido a las diferentes densidades del aire según la temperatura.

Sistema de iluminación y riego:

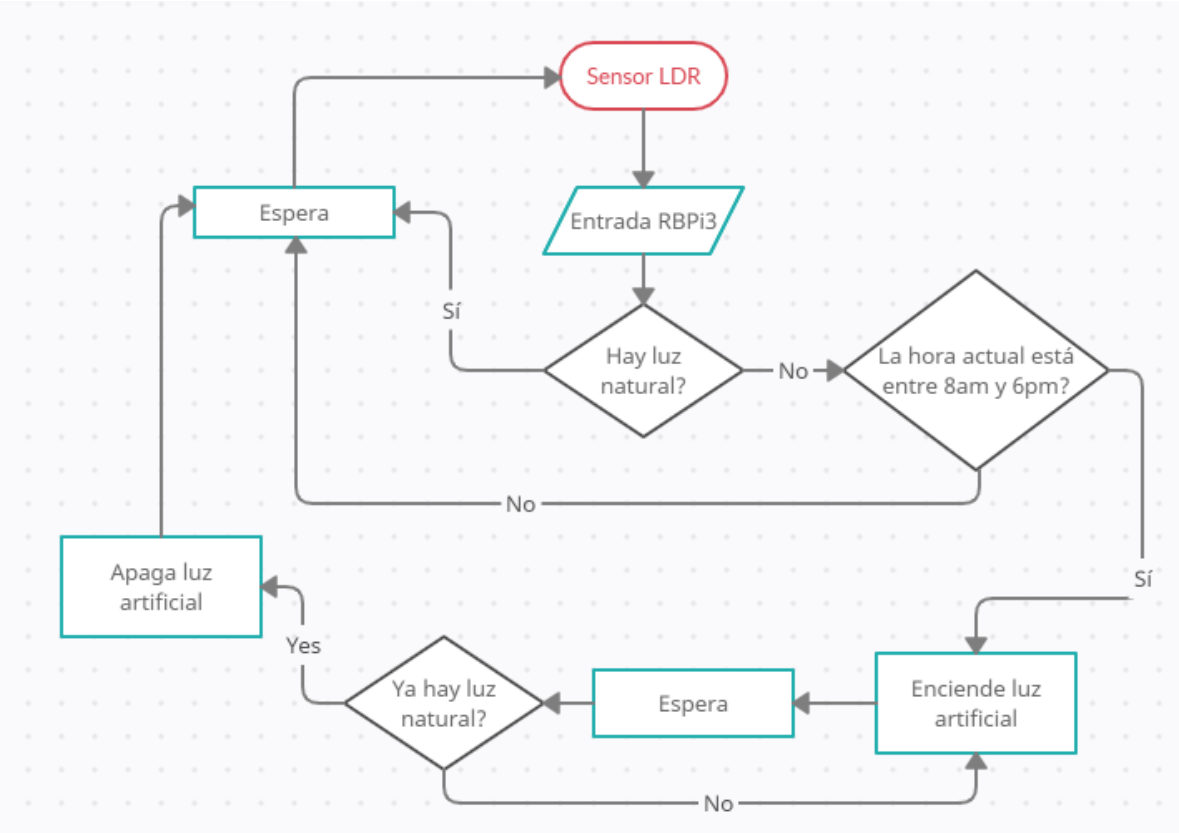
10.3.1.30: Tira LED y riego

La tira LED de iluminación y las mangueras de riego fueron instaladas en la parte central superior del invernadero, de esta manera se garantiza que el riego y la iluminación estarán distribuidos uniformemente en todo el invernadero.

10.3.2 Sistema de medición de Iluminación

Este sistema fue mucho más sencillo que el de Temperatura y Humedad. En este caso consta de un sensor LDR que al no tener la luz del sol envía una señal a la Raspberry para activar una luz artificial. De esta manera se garantiza la luz suficiente para que las plantas realicen su fotosíntesis.

El diagrama de flujo del sistema fue el siguiente:



Como se mencionó anteriormente, el funcionamiento del sistema de iluminación solo consta de activar o desactivar una luz artificial según la cantidad de luz natural. En este caso, la única condición es que el sistema solo funcione entre las 8am y 6pm, que es la parte del día en que las plantas suelen recibir la luz del sol. El objetivo de la condición fue proveer de luz a las plantas solo durante el día, dejándolas descansar durante la noche, que es lo que realmente pasa en la naturaleza. De esta manera las plantas crecen más saludables debido a que se evita el estrés por exceso de luz.

A diferencia de la Temperatura y la Humedad, en este caso no se guardó información en MySQL debido a que sería irrelevante en este proyecto guardar la información del encendido de la iluminación artificial. El sistema solo fue diseñado para enviar notificaciones vía email informando que la luz artificial se había activado.

El código utilizado en el sistema es el siguiente, en el cual se han omitido los comentarios detallados en las líneas similares a las del código del Sistema de Control de Temperatura y Humedad.

```
# Se importa la biblioteca que permite trabajar con los sensores de luz
from gpiozero import LightSensor
import RPi.GPIO as GPIO
# Debido a que se trabajará con fechas se importa datetime
from datetime import datetime
# Importación de biblioteca smtplib, para enviar alertas vía correo electrónico
import smtplib
import time

# En la variable ldr, se guarda el pin de entrada de información proveniente del sensor, en este caso el pin 26
ldr = LightSensor(26)
# Configuración del pin que controla al actuador, el pin 2
ACTFOCO = 2

GPIO.setmode(GPIO.BCM)
GPIO.setup(ACTFOCO, GPIO.OUT)

# Se pretende que el sensor trabaje las 24 hrs, por lo que se ingresa en un ciclo while infinito
while True:

        # Se apaga la luz artificial
        GPIO.output(ACTFOCO, GPIO.HIGH)

        # Cuando el sensor envía un valor mayor a 0 significa que no hay luz, por lo tanto, se indica que hacer en este caso
        if ldr.value > 0:

                # Se despliega mensaje indicando que no hay luz natural y se pide la hora del servidor
                print ("\nNo hay luz natural...\n")
                FECHA = datetime.now()

                # Condición de actuar solo entre las 8am y las 18pm, es decir, mientras es de día
                if (FECHA.hour > 8) and (FECHA.hour < 18):

                        # Si la condición se cumple, se imprime mensaje en pantalla y se notifica vía correo electrónico
                        print ("\tAccion: Se activa luz artificial!...\n")
                        asunto = "Notificacion Invernadero IoT"
                        mensaje = """
                                ESTE ES UN MENSAJE DE NOTIFICACION:

                                SE HA ACTIVADO LA ILUMINACION ARTIFICIAL!

                                """
                        mensaje = "Subject : {} \n\n {}".format(asunto, mensaje)
                        server = smtplib.SMTP("smtp.office365.com", 587)
                        server.starttls()
                        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
                        server.sendmail("invernaderoiotsagm@outlook.com", "marcko_antonio_966@yahoo.com", mensaje)
                        server.quit()
                        # Se informa que el correo ha sido enviado
                        print ("\tMensaje de alerta enviado\n")

                        # Se agrega un ciclo while en el que se indica que se active la luz artificial mientras el valor del sensor
                        # sea mayor a 0, o, en otras palabras, mientras la luz solar no incida sobre el sensor.
                        while ldr.value > 0:

                                GPIO.output(ACTFOCO, GPIO.LOW)

                        # Si el sensor envía un valor 0, la condición de while ya no se cumple, entonces termina su proceso y
                        # se envía una señal para desactivar la luz artificial debido a que nuevamente hay luz solar
                        GPIO.output(ACTFOCO, GPIO.HIGH)
```

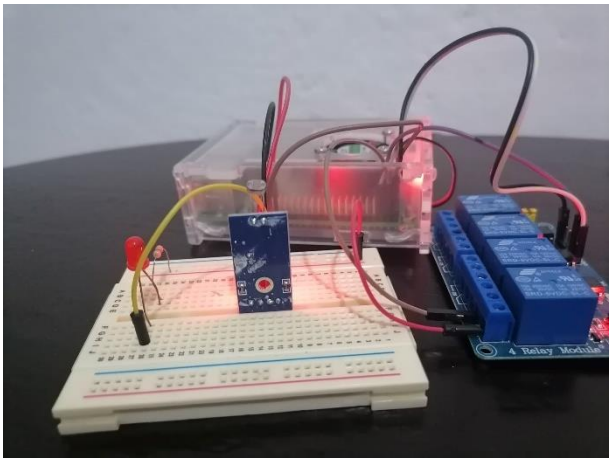
Como se pudo observar en el código, este sistema consta de un sensor que si envía valores igual a cero significa que la luz solar incide sobre el sensor LDR y se informa a través de mensajes en pantalla que es de día. En caso de que no haya luz incidiendo sobre el sensor el programa pide la hora al servidor, si la hora se encuentra entre las 8am y 18 pm significa que es de día y que probablemente es un día nublado y

por eso la luz solar no llega al sensor, y por lo tanto a las plantas. En este caso se activa la luz artificial que proveerá de una iluminación a las plantas. Esta acción es notificada a través de un correo electrónico.

Si el sensor no recibe luz y la hora del servidor se encuentra fuera del rango establecido significa que es de noche y en este caso no se activa la luz artificial debido a que el propósito del invernadero es que las plantas crezcan en un ambiente lo más similar posible a su entorno natural. En este caso, solo se informa a través de un mensaje en pantalla que el sensor no recibe luz debido a que es de noche.

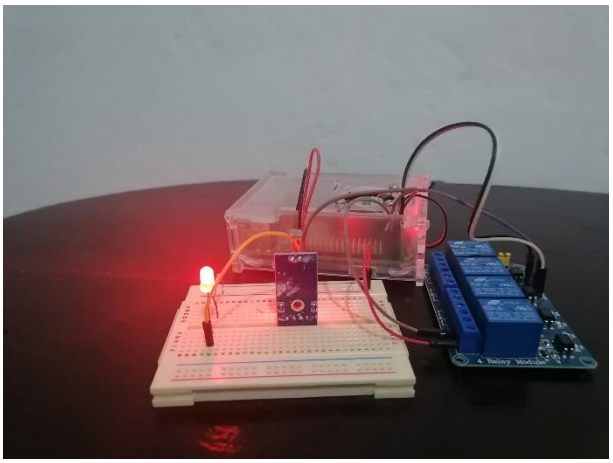
Para este sistema también se hicieron las primeras pruebas de funcionamiento con ayuda de LEDs y una Protoboard. El módulo del sensor LDR se conectó a la Raspberry para enviar la información que captaba, un pin de salida de la Raspberry se conectó a un relé para activar o desactivar un LED según la entrada del sensor.

A continuación, se presentan las imágenes de las pruebas:



10.3.2.1: Simulación con un LED

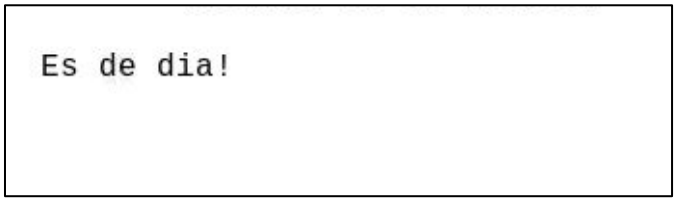
La imagen anterior muestra el LED rojo apagado debido a que un haz de luz estaba incidiendo sobre el sensor LDR, cuando la luz se apagó el LED rojo se prendió:



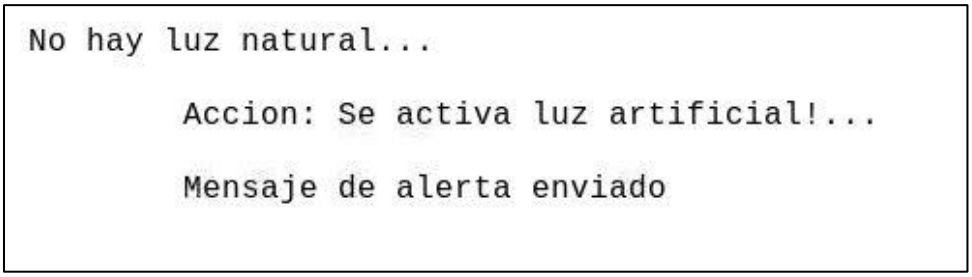
10.3.2.2: LED encendido debido a la ausencia de luz

El LED se mantuvo encendido en la oscuridad, y cuando se incidió nuevamente la luz sobre el sensor el LED se apagó, lo que demostró que el módulo del sensor LDR funcionaba correctamente.

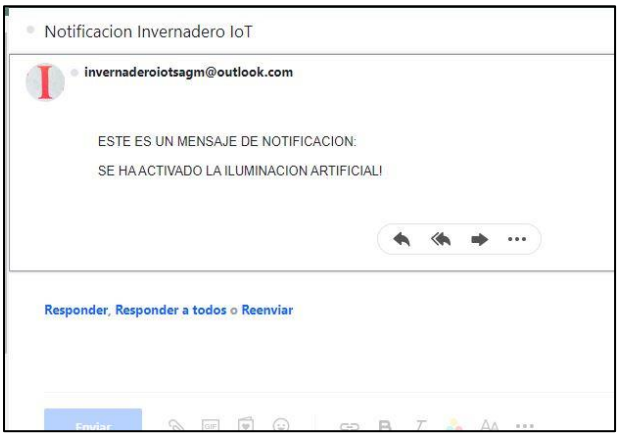
El programa del sistema fue ejecutado varias veces, dos veces durante el día, una vez con luz, y una vez más donde se bloqueó la luz para comprobar que el LED se encendía y que se enviaba la alerta por correo electrónico. En la parte de abajo se muestran las imágenes con los mensajes desplegados en pantalla, así como el correo de alerta.



10.3.2.3: Mensaje durante el día mientras hay luz natural

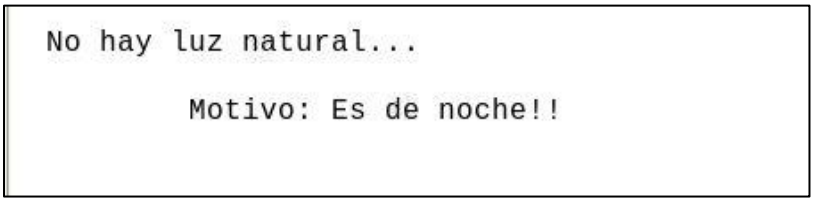


10.3.2.4: Mensaje cuando en día deja de incidir luz en el sensor LDR



10.3.2.5: Mail de notificación

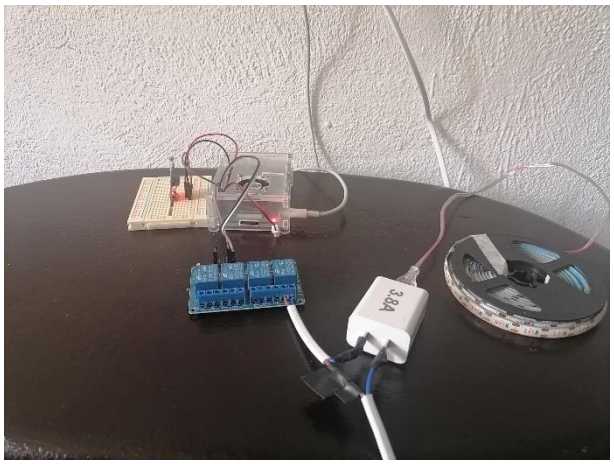
La tercera ejecución del código fue durante la noche, cuando el sensor no recibía luz pero se sabía que la luz artificial no encendería debido a que no era de día.



10.3.2.6: Mensaje desplegado durante la noche

Por último, después de comprobar que el sistema se comportaba como se esperaba, se pasó de la simulación con LEDs a la prueba con el equipo real. Para este caso se utilizó la misma tira LED utilizada en el sistema de control de Humedad y Temperatura debido a que además de producir calor, también emite luz en el mismo espectro de frecuencia que el Sol.

Antes de instalar el sistema en el invernadero, primero se hicieron pruebas con todo el equipo:



10.3.2.7: Prueba del Sistema de Iluminación

En la imagen se observa la tira LED conectada a un módulo relé, y a su vez, el módulo conectado a la Raspberry. También se observa la Protoboard en la que se encuentra conectado el sensor LDR. Al momento de tomar la foto había luz natural, por lo que la tira LED se encontraba apagada.

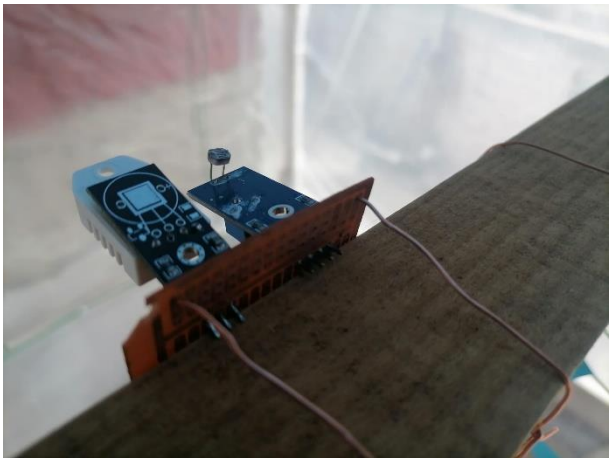
Al oscurecer la habitación, la tira LED se encendió automáticamente.

10.3.2.8: Activación de la Iluminación artificial

Ahora sí, después de comprobar que el sistema funcionaba correctamente con los elementos reales, el paso final consistió en la instalación del sistema en el invernadero y comprobar su funcionamiento.

A continuación, se presentan las fotografías de la instalación en el invernadero:

Sensor LDR:



10.3.2.9: Sensores LDR y DTH22

En la fotografía anterior nuevamente se observa la placa perforada que contiene a los sensores LDR y DTH22, en este caso fue tomada desde la parte superior para mostrar que la resistencia del sensor LDR fue colocada hacia arriba para recibir directamente los rayos del sol.

Tira LED:

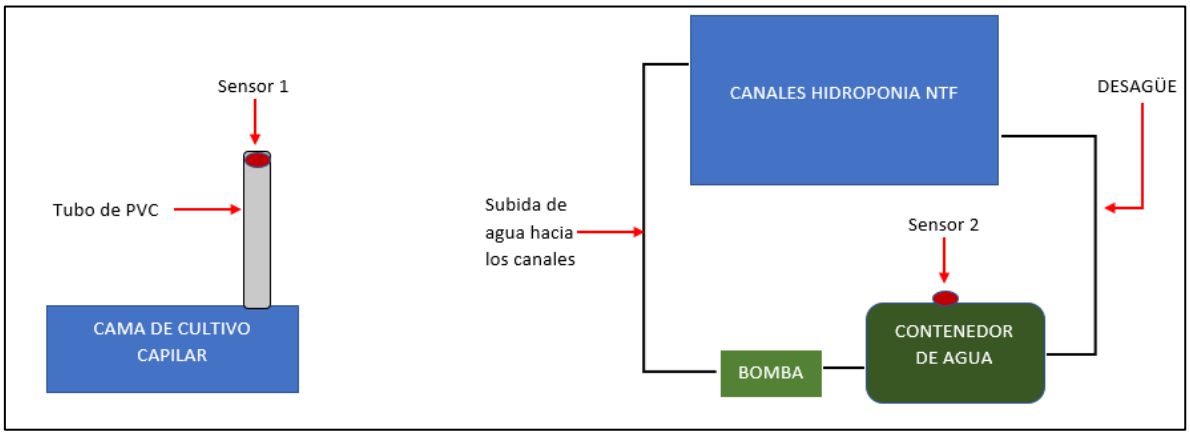
10.3.2.10: Tira LED

Aquí se puede observar más cerca la tira LED que fue colocada junto a la manguera de riego.

10.3.3 Sistema de Nivel de Agua en los contenedores

La cama de siembra capilar cuenta con un tubo de PVC donde se almacena agua para que las plantas la vayan absorbiendo según la cantidad que requieran. Para el caso de la hidroponía NFT el agua se almacena en un contenedor que se localiza debajo de los canales. En ambos casos es importante siempre contar con agua para que las plantas puedan crecer. Por lo que en esta sección se explica el sistema realizado para verificar los niveles de agua tanto en el tubo de PVC de la cama capilar, como del contenedor de la hidroponía NFT.

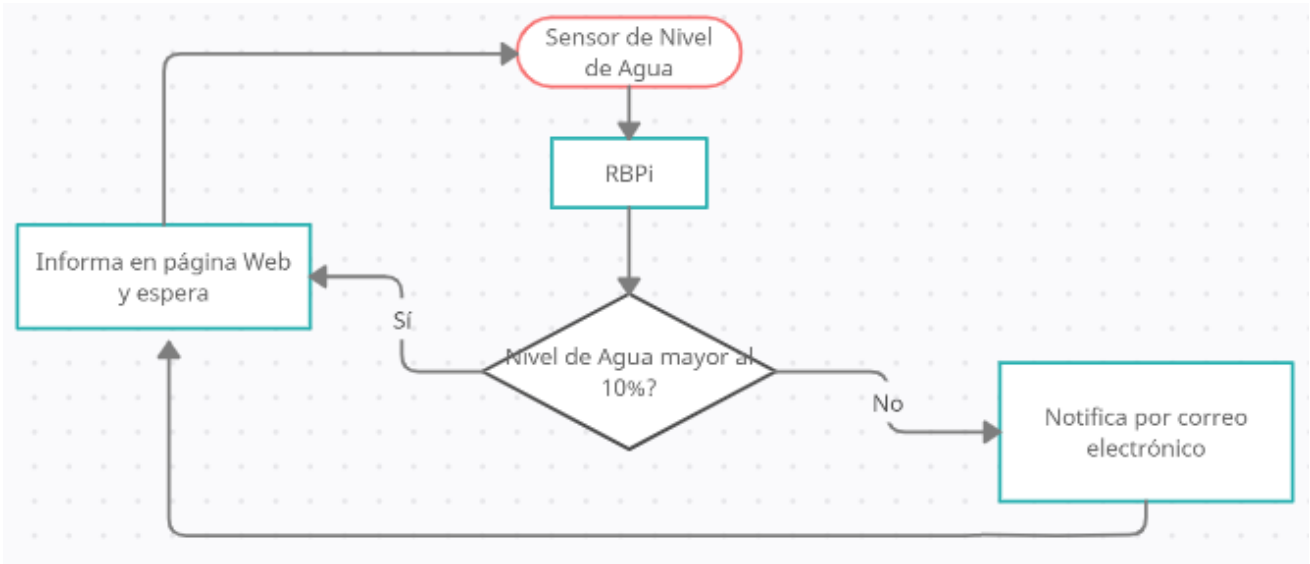
El sistema consistió en un par de sensores ultrasónicos *HC-SR04*, uno se instaló en la tapa del tubo de PVC y el otro en la tapa del contenedor como se muestra en la siguiente imagen:



10.3.3.1: Diagrama de los tipos de cultivo

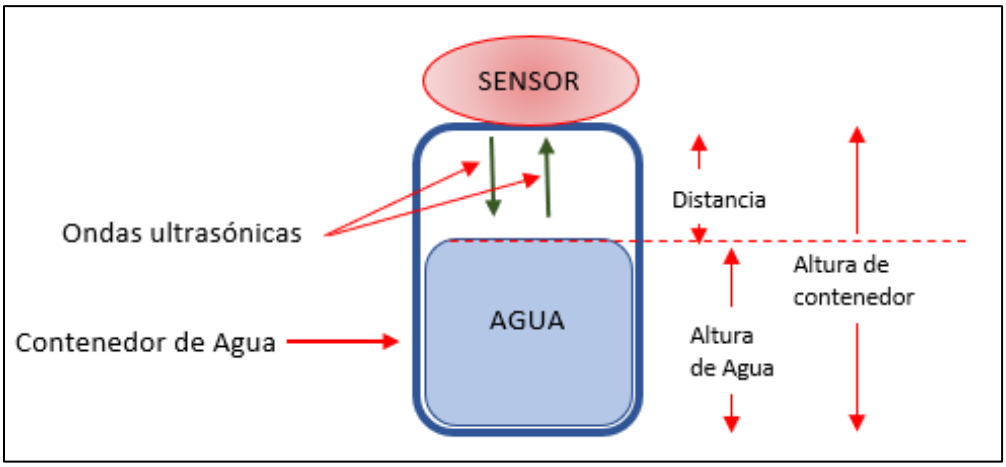
El objetivo de los sensores es informar por correo electrónico y en la página Web la cantidad de agua en cada uno de los sistemas de cultivo, de esta manera se garantiza que en ningún momento se tenga desabasto del líquido.

Para la creación de Sistema de Alerta de Nivel de Agua se utilizó el siguiente diagrama de flujo. El diagrama solo se realizó para un sensor debido a que el proceso es similar para ambos sensores.



El funcionamiento del Sistema de Nivel de Agua consiste en que el sensor mida el nivel de agua dentro del contenedor, si el nivel es mayor al 10% el valor se almacena en una base de datos y posteriormente es desplegado en la página Web solo con fines informativos. Cuando el nivel es inferior se notifica por correo electrónico para que el contenedor sea llenado nuevamente. Explicado el funcionamiento del sistema se pasa al diseño del mismo.

Para esta parte del proyecto se utilizó un sensor ultrasónico, el cual mide la distancia del objeto que tenga enfrente, con esta lógica se hizo el siguiente análisis:



10.3.3.2: Diagrama de Sensor

Si el sensor se colocaba en la parte superior del contenedor de agua sería capaz de medir la distancia entre la superficie del recipiente y el agua, en la imagen anterior se le denomina *Distancia*. Si se sabe la altura del contenedor y la *Distancia*, es posible saber la altura del agua con una simple resta:

$$Altura\ Agua = Altura\ de\ contenedor - Distancia$$

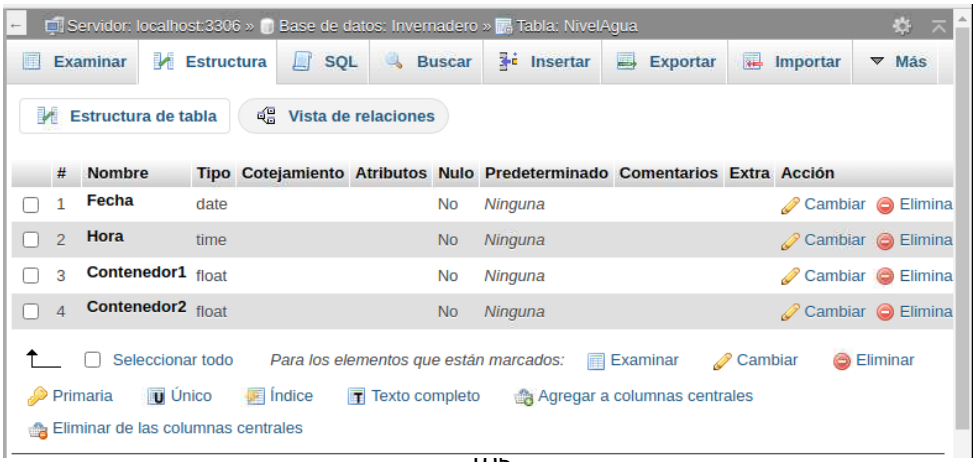
Para saber el porcentaje de agua dentro del contenedor solo de realizó una regla de tres. Si se sabía que la altura del agua era igual a la altura del contenedor, entonces el contenedor estaba al 100% de su capacidad, por lo que la regla de tres quedaba de la siguiente manera:

$$Altura\ de\ contenedor\ \rightarrow 100\%\ capacidad$$

$$Altura\ de\ Agua\ \rightarrow ¿?\ \%\ capacidad$$

$$¿?\ \% = \frac{Altura\ de\ Agua * 100\%\ capacidad}{Altura\ contenedor}$$

Con el análisis anterior y el diagrama de flujo se realizó el Sistema de Nivel de Agua, el primer paso fue crear una nueva tabla llamada *NivelAgua* en la base de datos *Invernadero*, esta tabla sirve para almacenar los datos del nivel de agua en cada uno de los contenedores. La tabla fue creada con cuatro columnas, la primera llamada *Fecha*, después *Hora*, *Contenedor1* y *Contenedor2*. El *Contenedor1* representa al tuvo de PVC de la capa de siembra capilar, y el *Contenedor2* al contenedor del cultivo hidropónico NFT.

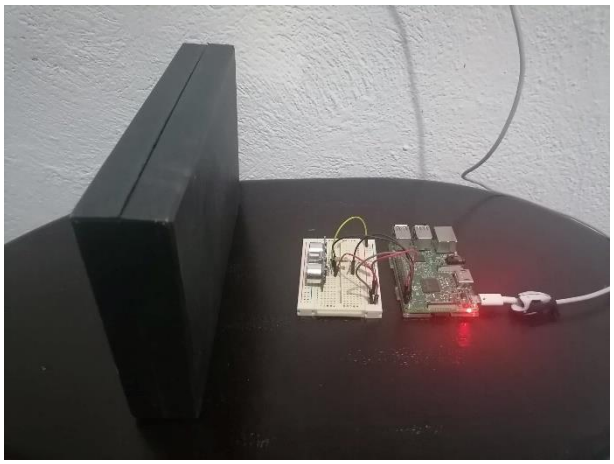


10.3.3.3: Creación de la tabla NivelAgua en la base de datos Invernadero

Después de tener la base de datos, el siguiente paso fue crear el código en Python. El código para verificar el funcionamiento de los dos sensores fue:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

#Se definen los pines a los que se conectará el sensor, Tx será el trigger y Rx el echo
Tx = 2
Rx = 3

#El pin Tx funcionará como transmisor y el Rx como receptor, es decir, como salida y entrada respectivamente
GPIO.setup(Tx, GPIO.OUT)
GPIO.setup(Rx, GPIO.IN)

# Se define la función dis(), que obtendrá la distancia entre el sensor y un objeto
def dis():

        # Primero se activa trigger para mandar su señal ultrasónica por 0.1 [uS] y posteriormente se apaga
        GPIO.output(Tx, GPIO.LOW)
        time.sleep(0.0001)
        GPIO.output(Tx, GPIO.HIGH)

        # Se le pide al pin echo que guarde la hora ultima hora antes de que le llegue la señal de Tx
        while GPIO.input(Rx) == 0:

                INICIO = time.time()

        # Se le pide a echo que guarde la ultima hora en recibir la señal de Tx
        while GPIO.input(Rx) > 0:

                FIN = time.time()

        # La duración de la señal es la resta de las dos horas guardadas, antes y después de recibir la señal
        DURACION = FIN – INICIO
        # La formula de la distancia es igual a Distancia = Tiempo * Velocidad, en este caso la velocidad del sonido, distancia será
        # la que recorrió la onda hasta chocar con el objeto y regresar al sensor, por lo que muestra el doble de la distancia entre el
        # sensor y el objeto, para resolver esto la distancia obtenida se debe dividir entre dos
        DISTANCIA = (DURACION * 34300)/2
        return DISTANCIA

print ("Prueba de distancia:\n")
# En el código principal se llama a la función dis() para calcular la distancia
dist = dis()
# Con la función round() se limita a 2 el número de decimales mostrados en el cálculo de la distancia
dist1 = round(dist,2)
print ("\tLa distancia es de: ", dist1, "[cm]")
```

La prueba de funcionamiento consistió en poner un obstáculo frente al sensor, obtener mediante el código la distancia y después comprobarlo con un flexómetro.



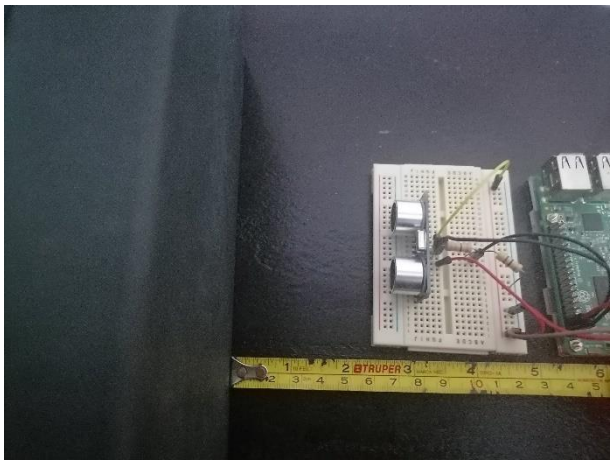
10.3.3.4: Prueba de sensores ultrasónicos

La primera prueba con el sensor 1, arrojó una distancia de 6.51 [cm].

```
Prueba de distancia:

        La distancia es de:  6.51 [cm]
```

10.3.3.5: Mensaje en pantalla

Se comprobó la distancia con el flexómetro:

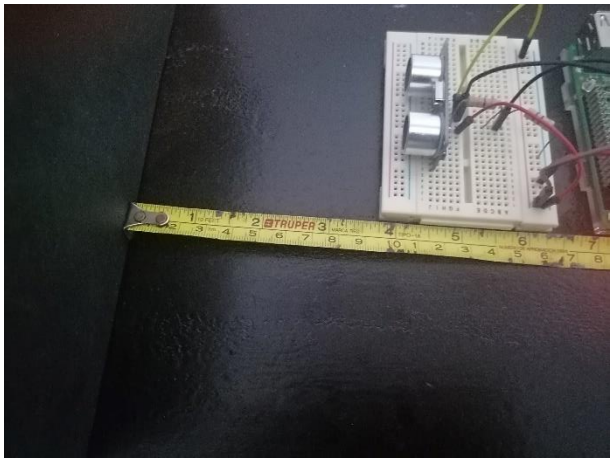


10.3.3.6: Comprobación de distancia

Después se intercambió el sensor 1 por el sensor 2, se alejó el obstáculo y nuevamente se ejecutó el programa, en este caso dio como resultado una distancia de 10.09 [cm], misma que fue comprobada con el flexómetro.

```
Prueba de distancia:

        La distancia es de:  10.09 [cm]
```

10.3.3.7: Mensaje en pantalla



10.3.3.8: Comprobación de distancia

Con la prueba de que los dos sensores funcionaban correctamente se hicieron las modificaciones para el código final del Sistema de Nivel de Agua, el resultado fue el siguiente:

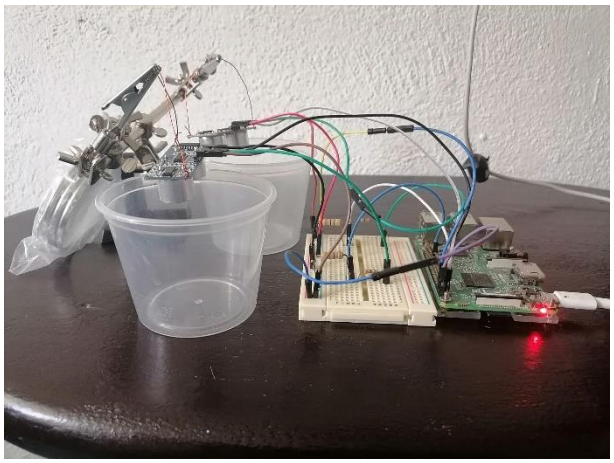
```
import RPi.GPIO as GPIO
import time
import mysql.connector
import smtplib

GPIO.setmode(GPIO.BCM)
# Inicio de sesión en la base de datos Invernadero
mibd = mysql.connector.connect(host = "localhost", user = "pi", passwd = "TesisIoT2021", database = "Invernadero")

# Configuración de los pines para los dos sensores ultrasónicos
Tx1 = 2
Rx1 = 3
Tx2 = 4
Rx2 = 5

# Configuración de los pines como entradas o salidas
GPIO.setup(Rx1, GPIO.IN)
GPIO.setup(Rx2, GPIO.IN)
GPIO.setup(Tx1, GPIO.OUT)
GPIO.setup(Tx2, GPIO.OUT)

# Función para calcular la altura del agua en un contenedor
def faltura(TxX, RxX):

        GPIO.output(TxX, GPIO.LOW)
        time.sleep(0.0001)
        GPIO.output(TxX, GPIO.HIGH)

        while GPIO.input(RxX) == 0:

                INICIO = time.time()

        while GPIO.input(RxX) > 0:

                FIN = time.time()

        DURACION = FIN - INICIO
        DISTANCIA = (DURACION * 34300)/2
        # Despues de calcular la distancia entre el sensor y el agua se procede a calcular la altura con la formula
        # Altura = Altura del contenedor – DISTANCIA
        ALTURA = 5.5 - DISTANCIA
        return ALTURA

# Al saber la altura del agua y del contenedor se puede obtener el porcentaje de agua dentro del contenedor
def fporcentaje():

        PORCENTAJE = (VALTURA*100)/5.5
        return PORCENTAJE

# El programa principal se ejecuta dentro de un ciclo while True para que funcione infinitamente
while True:

        print ("== NIVEL DE AGUA ==\n")
        # Asunto del correo de notificación
        asunto = "Notificación invernadero IoT"
        # Primero se informa del contenedor 1, el tubo de PVC
        print( "\n\tCONTENEDOR 1 (CAMA CULTIVO CAPILAR)\n")
        # Se llama a la función de altura del agua, enviándole los valores de los pines del sensor 1
        VALTURA = faltura(Tx1, Rx1)
        # Se calcula el porcentaje de agua dentro del contenedor
        VPORCENTAJE = fporcentaje()
        # El porcentaje se trunca a dos decimales y se guarda en la variable VP1 que será utilizada más adelante para enviar los datos
        # la base de datos
        VP1 = round(VPORCENTAJE, 2)
```

```
# Se imprime en pantalla el nivel de agua
print ("\n\t\tEl nivel del agua se encuentra al ", VP1, "% de su capacidad.\n")
# Cuando es menor al 10% se debe enviar notificación por correo electrónico, en este caso se utiliza la condición if
if VPORCENTAJE < 10:

        # Se alerta a través de un mensaje en la pantalla
        print("\t\tNivel de agua muy bajo, es necesario rellenar el contenedor!\n")
        # Cuerpo del mensaje que será enviado
        mensaje = """
                ESTE ES UN MENSAJE DE ALERTA:

                        ESTADO: Nivel de agua menor al 10%
                        CONTENEDOR: Tubo de PVC
                        RECOMENCACION: Rellenar urgentemente
        """
        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
        server = smtplib.SMTP("smtp.office365.com", 587)
        server.starttls()
        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
        server.quit()
        print ("\t\tCorreo de notificacion enviado.\n")

else:

        # En caso de que el nivel sea mayor al 10% solo se notifica a través de un mensaje en pantalla
        print("\t\tLos niveles de agua son correctos!\n")

# Para el contenedor 2 el proceso es similar al del contenedor 1, por lo que se omiten los comentarios
print( "\n\tCONTENEDOR 2 (Hidroponia NTF):\n")
VALTURA = faltura(Tx2, Rx2)
VPORCENTAJE = fporcentaje()
VP2 = round(VPORCENTAJE, 2)
print ("\n\t\tEl nivel del agua se encuentra al ", VP2, "% de su capacidad.\n")
if VPORCENTAJE < 10:

        print("\t\tNivel de agua muy bajo, es necesario rellenar el contenedor!\n")
        mensaje = """
                ESTE ES UN MENSAJE DE ALERTA:

                        ESTE ES UN MENSAJE DE ALERTA:
                        ESTADO: Nivel de agua menor al 10%
                        CONTENEDOR: Contenedor hidroponia NTF
                        RECOMENCACION: Rellenar urgentemente
        """
        mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
        server = smtplib.SMTP("smtp.office365.com", 587)
        server.starttls()
        server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
        server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
        server.quit()
        print ("\t\tCorreo de notificacion enviado.\n")

else:

        print("\t\tLos niveles de agua son correctos!\n")

# Después de calcular el nivel de agua en los dos contenedores se procede a guardar la información en MySQL
# En la tabla NivelAgua existen las columnas de fecha y hora, por lo que se solicita dicha información al servidor
FECHA = time.strftime("%y/%m/%d")
HORA = time.strftime("%H:%M:%S")
mycursor = mibd.cursor()
# Se informa a MySQL que en la tabla NivelAgua de la base de datos Invernadero se guardará información en las columnas
# Fecha, Hora, Contenedor1 y Contenedor2, respectivamente, así como el tipo de valor que se guardará
sql = "INSERT INTO NivelAgua(Fecha, Hora, Contenedor1, Contenedor2) VALUES (%s, %s, %s, %s)"
# Se indica que los valores que queremos ingresar se encuentran en las variables FECHA, HORA, VP1 y VP2
val = (FECHA, HORA, VP1, VP2)
mycursor.execute(sql, val)
```

```
mibd.commit()
# Se notifica en la pantalla que los datos han sido guardados correctamente en MySQL
print("\n\nDatos guardados correctamente.\n")
# El nivel de agua se medira cada hora, por lo que se duerme el proceso 3,600 segundos, el equivalente a una hora
time.sleep(3600)
```

Para el código mostrado es importante mencionar que las pruebas se hicieron en dos recipientes de plástico, ambos de 6 [cm] de altura, pero en el código se especificaron solo 5.5 [cm] para evitar que el agua llegara a los sensores.

La primera prueba se realizó con los contenedores vacíos.



10.3.3.9: Prueba con contenedores vacíos

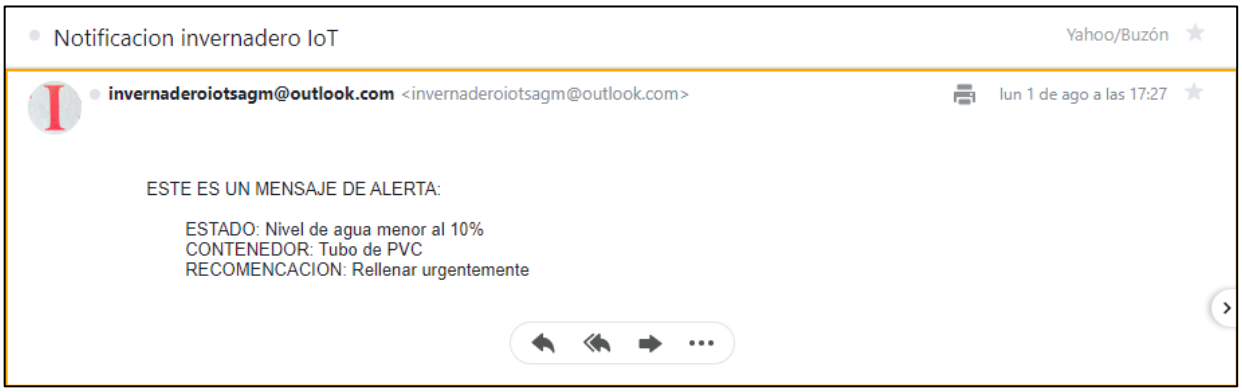
Debido a que los contenedores se encontraban vacíos se desplegó el siguiente mensaje en pantalla:

```
== NIVEL DE AGUA ==

        CONTENEDOR 1 (CAMA CULTIVO CAPILAR)

                El nivel del agua se encuentra al  -2.12 % de su capacidad.

                Nivel de agua muy bajo, es necesario rellenar el contenedor!

                Correo de notificacion enviado.

        CONTENEDOR 2 (Hidroponia NTF):

                El nivel del agua se encuentra al  9.0 % de su capacidad.

                Nivel de agua muy bajo, es necesario rellenar el contenedor!

                Correo de notificacion enviado.

Datos guardados correctamente.
```

10.3.3.10: Mensajes desplegados en pantalla

Se observan valores muy cercanos a cero, y no exactamente cero a pesar de tener los contenedores vacíos, esto es debido al error presente en los sensores y al tamaño de los contenedores.

En los mensajes desplegados se observa que se enviaron correos de notificación y que los valores de los niveles de agua fueron guardados en la tabla *NivelAgua* correctamente.



10.3.3.11: Correo de notificación del contenedor de hidroponía NTF

10.3.3.12: Correo de notificación del tubo de PVC

Fecha	Hora	Contenedor1	Contenedor2
2022-08-01	17:27:43	-2.12	9

10.3.3.13: Llenado de la tabla NivelAgua

La segunda prueba consistió en agregar agua a los contenedores. Al primero se le agrego aproximadamente la mitad de su capacidad, y el segundo contenedor quedó casi lleno.

```
== NIVEL DE AGUA ==

        CONTENEDOR 1 (CAMA CULTIVO CAPILAR)

                El nivel del agua se encuentra al  54.55 % de su capacidad.

                Los niveles de agua son correctos!

        CONTENEDOR 2 (Hidroponia NTF):

                El nivel del agua se encuentra al  66.49 % de su capacidad.

                Los niveles de agua son correctos!

Datos guardados correctamente.
```

10.3.3.14: Mensaje desplegado en la segunda prueba

Después de comprobar que el código Python funcionaba correctamente se inició con el código php para la página Web donde se desplegaría la información del nivel de agua.

```
<html>
        <head>
                // Titulo que tendrá la pestaña de la página Web
                <title>Nivel de Agua</title>
                <meta charset="utf-8">
        </head>
        <body>
                <?php
                        // Se le pide a HTML iniciar sesion en MySQL
                        $link = mysqli_connect("localhost", "pi", "TesisIoT2021");
                        // Selección de la base de datos en la que se trabajará
                        mysqli_select_db($link, "Invernadero");
                        // Selección de la table en la base de datos
                        $result = mysqli_query($link, "SELECT * FROM NivelAgua");
                        $renglones = mysqli_num_rows($result);
                        $ultima = $renglones-1;
                        // Se selecciona la última fila de la tabla, es decir el ultimo registro guardado, que es el que se
                        // mostrará
                        mysqli_data_seek($result, $ultima);
                        $extraido = mysqli_fetch_array($result);
                        // Se comienza con el diseño de la página, en este caso el texto que se va a desplegar
                        echo "<h1><u>Nivel de los contenedores de agua</u></h1><br>";
                        echo "<br><h2>Ultima actualizacion:</h2><br>";
                        // Se comienza a sustraer y mostrar la información guardada en la última fila
                        echo "Fecha: ".$extraido["Fecha"]."<br>";
                        echo "Hora: ".$extraido["Hora"]."<br>";
                        echo "Tubo PVC: ".$extraido["Contenedor1"]." %<br>";
                        // Se pide que en caso de que el nivel sea menor al 10% muestre un mensaje de alerta en color rojo
                        if ($extraido["Contenedor1"] < 10) {
                                echo "<font color=red>IMPORTANTE: URGE RELLENAR CONTENEDOR 1!!</font><br>";}
                        echo "Contenedor NTF: ".$extraido["Contenedor2"]." %<br>";
                        if ($extraido["Contenedor2"] < 10){
                                echo "<font color=red>IMPORTANTE: URGE RELLENAR CONTENEDOR 2!!</font><br>";}
                ?>
        </body>
</html>
```

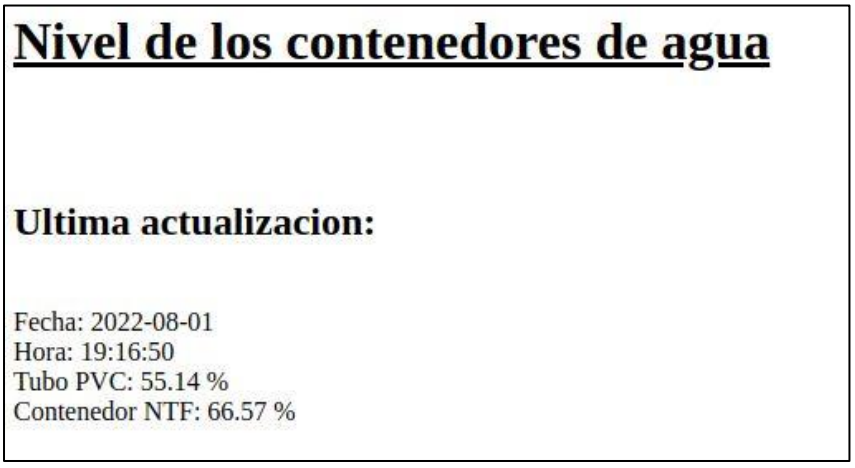
En este caso se diseñó la página para mostrar la última información guardada en la base de datos, así como la fecha y hora. Además, se pide a HTML muestre un mensaje de alerta en color rojo para avisar que es necesario rellenar los contenedores cuando están al 10% o menos de su capacidad.

Para comprobar el funcionamiento de la página nuevamente se hicieron dos pruebas, una con los contenedores con agua, el resultado fue el siguiente:

```
== NIVEL DE AGUA ==

        CONTENEDOR 1 (CAMA CULTIVO CAPILAR)

                El nivel del agua se encuentra al  55.14 % de su capacidad.
                Los niveles de agua son correctos!

        CONTENEDOR 2 (Hidroponia NTF):

                El nivel del agua se encuentra al  66.57 % de su capacidad.
                Los niveles de agua son correctos!

Datos guardados correctamente.
```

10.3.3.15: Mensajes mostrados en pantalla en la prueba 1

Al ingresar a la página 192.168.100.42/NIVEL.php se observó lo siguiente:



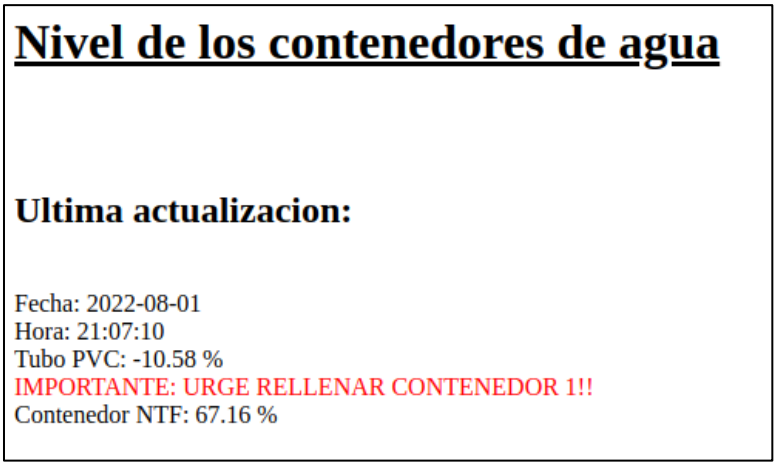
10.3.3.16: Página Web con la primera prueba

Y en ambas imágenes se observa el mismo resultado, por lo que se demuestra que cuando los contenedores tienen más del 10% de su capacidad la página funciona correctamente.

En la segunda prueba se vació el agua del contenedor 1 y los resultados fueron:

```
== NIVEL DE AGUA ==

        CONTENEDOR 1 (CAMA CULTIVO CAPILAR)

                El nivel del agua se encuentra al  -10.58 % de su capacidad.

                Nivel de agua muy bajo, es necesario rellenar el contenedor!

                Correo de notificacion enviado.

        CONTENEDOR 2 (Hidroponia NTF):

                El nivel del agua se encuentra al  67.16 % de su capacidad.

                Los niveles de agua son correctos!

Datos guardados correctamente.
```

10.3.3.17: Mensajes de pantalla en la 2da prueba

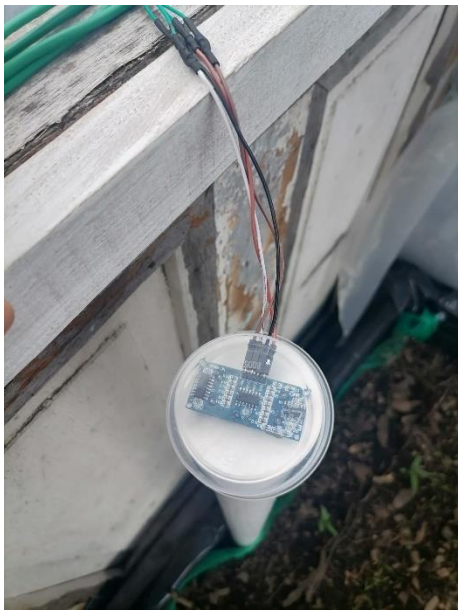


10.3.3.18: Pagina Web en la segunda prueba

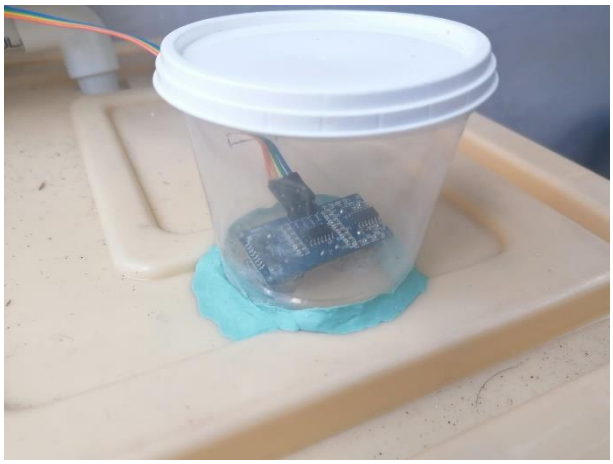
Para este caso se observa debido a que el contenedor 1 tenía menos del 10% de su capacidad se desplego un mensaje de alerta en color rojo en la página. Por lo que con estos dos ejemplos se demuestra el funcionamiento completo del Sistema de Nivel de Agua en los contenedores.

Al igual como se hizo en los sistemas pasados, después de comprobar el funcionamiento del sistema se procedió a la instalación de los elementos en el invernadero.

Los sensores supersónicos fueron instalados en las tapas de los dos contenedores, el contenedor de la solución nutritiva NFT y en el tubo de riego de la capa capilar.



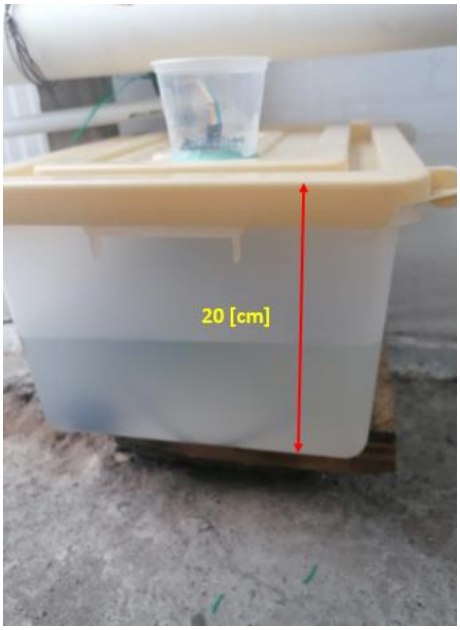
10.3.3.19: Sensor en el tubo de riego de la cama capilar



10.3.3.20: Sensor en el contenedor de solución nutritiva NFT

Los sensores fueron cubiertos con recipientes de plástico trasparente para evitar que se mojaran con el riego o cualquier otro líquido. Los cuatro pines fueron conectados por medio de jumpers que a por medio de cable para Protoboard se conectaron a la Raspberry y a la fuente de 5 [V].

10.3.3.21: Altura del tubo de riego de la cama capilar



10.3.3.22: Altura del contenedor de solución nutritiva

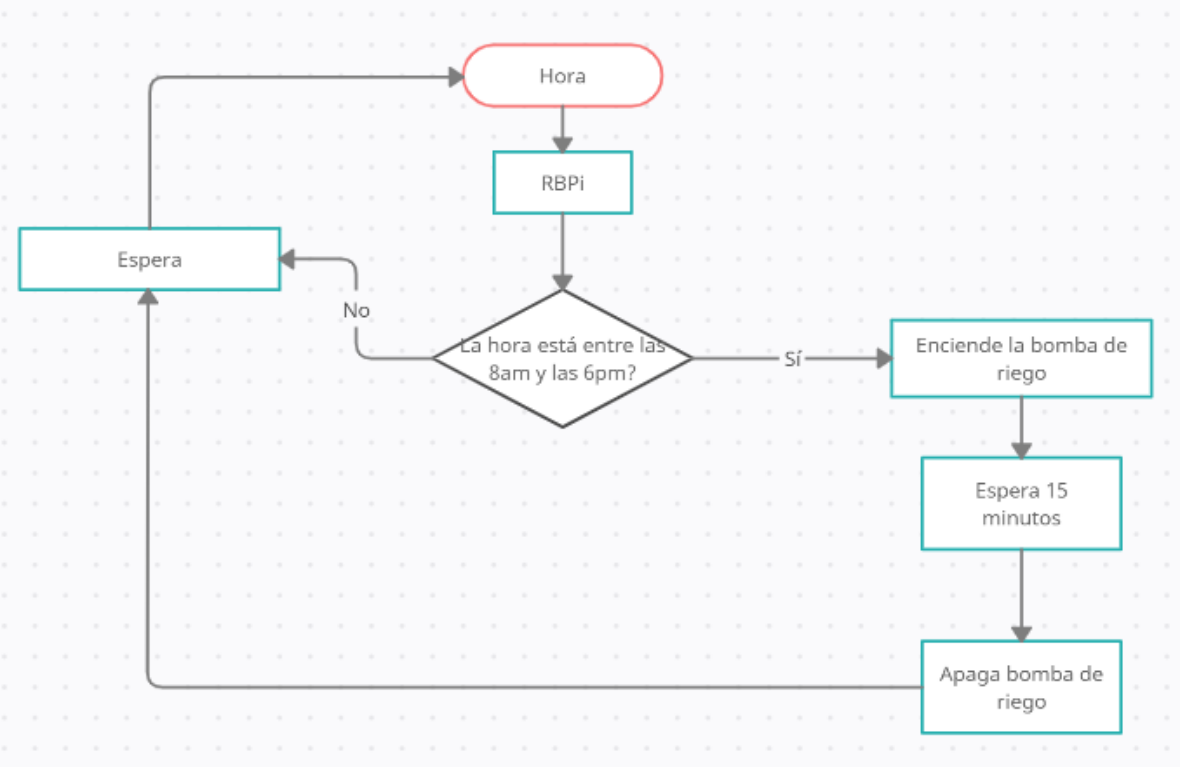
Lo último que se hizo en la instalación fue medir las alturas de los contenedores debido a que es necesario conocerlas para que el sistema determine la cantidad de agua en cada contenedor. Las imágenes anteriores muestran que el tubo de riego de la cama hidropónica tiene una altura de 80 [cm], mientras que

la altura del contenedor de solución nutritiva es de 20 [cm]. Las modificaciones con estas alturas se observarán en el subcapítulo 10.4, donde se unirán todos los sistemas para obtener el sistema final.

10.3.4 Sistema de Autorriego para el cultivo hidropónico NFT

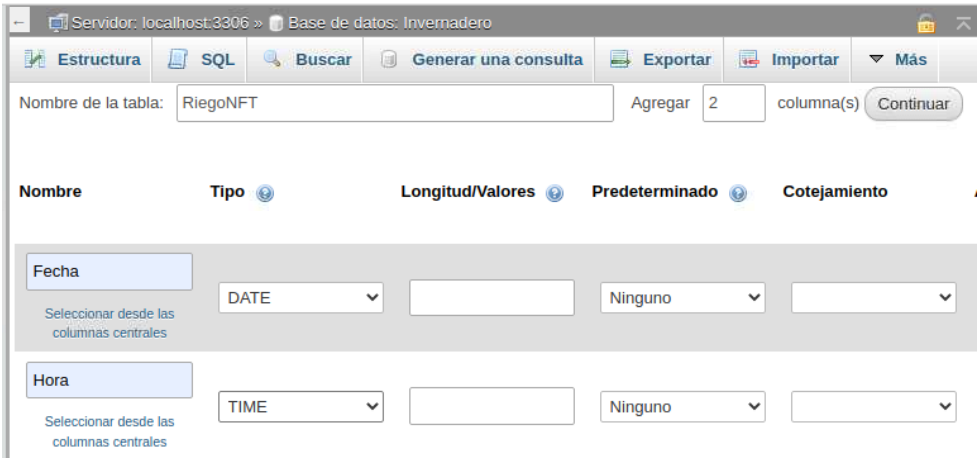
Este sistema consiste en activar automáticamente la bomba que regará los canales del cultivo NFT, la bomba llevará hasta el canal más alto a la solución para que recorra todos los canales hasta regresar nuevamente al contenedor, de esta manera se tendrá la recirculación de la solución nutritiva que alimentará y oxigenará a las plantas.

La bomba se activará cada hora por quince minutos, pero solo durante el día, brindándole un descanso al cultivo durante la noche. El diagrama de flujo del sistema es el siguiente:



Para este caso el sistema necesita que la Raspberry solo actúe cada hora activando por quince minutos la bomba de riego, siempre y cuando sea de día. Una vez desactivada la bomba se guarda la fecha y hora del riego para informar a través de una página Web.

Primero en la base de datos *Invernadero* se creó una nueva tabla, en este caso llamada *RiegoNFT*.



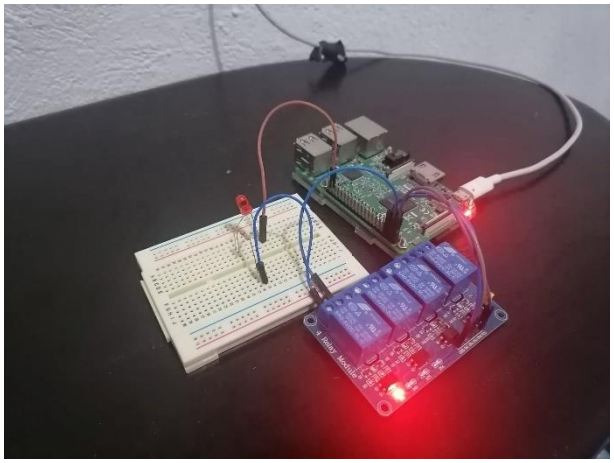
10.3.4.1: Creación de la tabla RiegoNFT

Posteriormente se diseñó el código en Python, y quedó como a continuación se muestra:

```
import RPi.GPIO as GPIO
from datetime import datetime
import time
import mysql.connector

# Se configura el pin que activará el riego
RiegoNFT = 2

GPIO.setmode(GPIO.BCM)
GPIO.setup(RiegoNFT, GPIO.OUT)
# Inicio de sesión en la base de datos que contiene la tabla en la que se trabajará
mibd = mysql.connector.connect(host = "localhost", user = "pi", passwd = "TesisIoT2021", database = "Invernadero")

# El ciclo while servirá para que el sistema funcione las 24 hrs
while True:

        # Se desactiva la salida del pin de riego
        GPIO.output(RiegoNFT, GPIO.HIGH)
        # La hora y fecha del servidor son guardadas en la variable fecha
        fecha = datetime.now()
        # Solo se imprime la hora
        print("Hora: ", fecha.strftime("%H:%M:%S"))
        # El sistema solo debe funcionar entre las 8am y 6pm, por lo que se agrega una condicion para activar el riego
        if (fecha.hour >= 8) and (fecha.hour <= 18):

                # Si se cumple la condicion se notifica en la pantalla que el riego se va a activar
                print ("Se han activado el riego automatico y recirculacion, por favor espere...\n")
                GPIO.output(RiegoNFT, GPIO.LOW)
                # Se mantiene el riego activo por 15 minutos que equivalen a 900 segundos
                time.sleep(900)
                GPIO.output(RiegoNFT, GPIO.HIGH)
                # La fecha y hora del riego se deben guardar en MySQL para ser mostrados en una página Web
                mycursor = mibd.cursor()
                FECHA = fecha.date()
                HORA = fecha.strftime("%H:%M:%S")
                # Se indica la tabla en la que será guardada la información
                sql = "INSERT INTO RiegoNFT(Fecha, Hora) VALUES (%s, %s)"
                # Se indica que la informacion que se guardará se encuentra en las variables FECHA y HORA, respectivamente
                val = (FECHA, HORA)
                mycursor.execute(sql, val)
                mibd.commit()
                # Después de guardar los datos se notifica en pantalla
                print("\nDatos guardados correctamente.\n")
```

```
                # Tambien se notifica que el riego ha terminado
                print ("El riego y recirculacion han finalizado. Gracias!\n")

        # Si la condicion no se cumple solo se imprime un mensaje en pantalla
        else:

                print ("El riego solo se activa durante el dia. Gracias!\n")

        # El sistema se duerme por una hora antes de volver a ejecutarse
        time.sleep(3600)
```

Como se ha venido manejando, después de crear el código se procede a realizar pruebas, en este caso se ejecutó el código y para probar su funcionamiento se ocupó un LED.



10.3.4.2: Prueba de bomba de riego

La primera prueba fue durante la noche fuera del horario establecido en la condición, y debido a que no se cumplió no se activó la bomba, pero se desplego el mensaje que estaba programado:

```
Hora:  00:22:34
El riego solo se activa durante el dia. Gracias!
```

10.3.4.3: Mensaje del Sistema de riego durante la noche

Para fines prácticos, el código se modificó un poco solo para comprobar que funcionaba correctamente sin la necesidad de esperar cada hora o los quince minutos de activación de la bomba. Para las pruebas se configuro que la bomba solo encendiera 5 segundos, y esperara por 10 segundos apagada, en lugar de una hora. El resultado fue el siguiente:

```
>>> %Run RiegoNFT.py
 Hora:  17:27:13
 Se han activado el riego automatico y recirculacion, por favor espere...


 Datos guardados correctamente.

 El riego y recirculacion han finalizado. Gracias!
```


10.3.4.4: Mensajes desplegados de activación de la bomba de riego

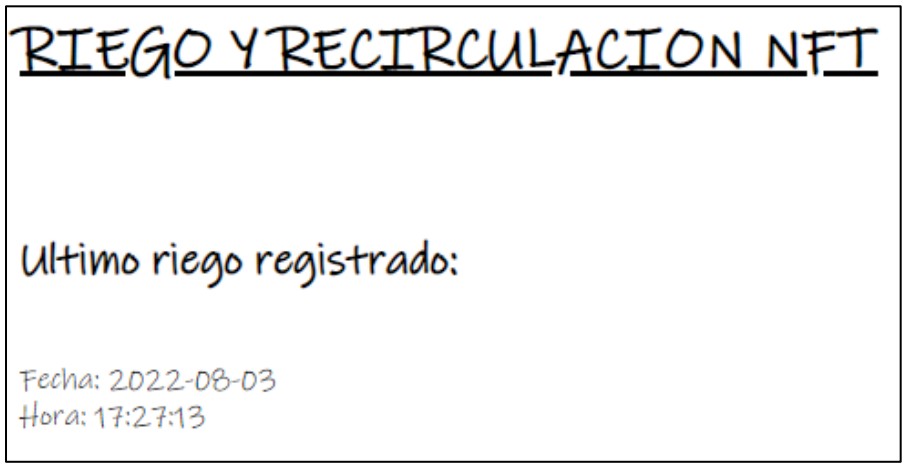
Se observa que el ciclo *while* se ejecutó una vez y la base de datos *RiegoNFT* guardó los valores:

10.3.4.5: Valores guardados en la tabla RiegoNFT

Ahora se mostrará el código PHP para que los valores de la tabla *RiegoNFT* se muestren en una página de Internet con la dirección 192.168.100.42/NFT.php:

```
<html>
	<head>
		<title>Riego NFT</title>
		<meta charset="utf-8">
	</head>
	<body>
		<?php
			$link = mysqli_connect("localhost", "pi", "TesisIoT2021");
			mysqli_select_db($link, "Invernadero");
			$result = mysqli_query($link, "SELECT * FROM RiegoNFT");
			$renglones = mysqli_num_rows($result);
			$ultima = $renglones-1;
			mysqli_data_seek($result, $ultima);
			$extraido = mysqli_fetch_array($result);
			echo "<h1><u>RIEGO Y RECIRCULACION NFT</u></h1><br>";
			echo "<br><h2>Ultimo riego registrado:</h2><br>";
			echo "Fecha: ".$extraido["Fecha"]."<br>";
			echo "Hora: ".$extraido["Hora"]."<br>";
		?>
	</body>
</html>
```

El código no se comentó debido a que es mucho más sencillo a los mostrados anteriormente, solo consiste en tomar el valor de la última fila de la tabla *RiegoNFT* y posteriormente es desplegada en la página de la siguiente manera:



10.3.4.6: Pagina Web del Riego y Recirculación

Después de comprobar el funcionamiento del código Python y el código HTML, se probó el circuito con la bomba y se comprobó que solo encendía cada hora durante el día.

Con las pruebas realizadas fue posible determinar que el sistema estaba listo por lo que se pasó a su implementación en el invernadero. En este caso, al sistema de canales de PVC instalado previamente se le colocó el contenedor de solución nutritiva, en las imágenes se observa que es transparente, contrario a lo que se mencionó en capítulos anteriores, esto con el objetivo de visualizar su interior para el informe, posteriormente fue pintado de negro para evitar el paso de a luz solar.

A la tapa del contenedor se le realizaron dos orificios extra para el tubo de desagüe de los canales, la maguera y el cable de la bomba sumergible colocada al interior del contenedor.

10.3.4.7: Bomba sumergible dentro del contenedor

10.3.5 Contador de días para cambio de solución nutritiva:

El contador es el parámetro más sencillo que se diseñó para el invernadero, simplemente consististe en un programa en Python que informa de cuantos días lleva en el contenedor la solución nutritiva del cultivo NFT, esto debido a que se recomienda que la solución no se utilice por más de 15 días debido a que con el paso del tiempo y la recirculación comienza a perder sus nutrientes dejando sin alimento a las plantas.

El programa cuenta hasta quince días, en el día 14 envía la primera notificación informando que falta un día para cambiar la solución, y en el día 15 envía la segunda notificación informando que es día de cambio de solución. Pasados los 15 días el contador se reinicia. También, los días que pasan son guardados en una tabla para que sean mostrados en una página Web.

Primero se realizó una tabla en MySQL llamada *ContadorNFT*, donde se guarda la información de los días y la fecha de comienzo de uso de la solución, la tabla solo posee tres columnas llamadas *días, INICIO* y *FINAL*. Posteriormente se realizó el código Python siguiente:

```
import smtplib
import mysql.connector
import time

asunto = "Notificacion Invernadero IoT"
mibd = mysql.connector.connect(host = "localhost", user = "pi", passwd = "TesisIoT2021", database = "Invernadero")
```

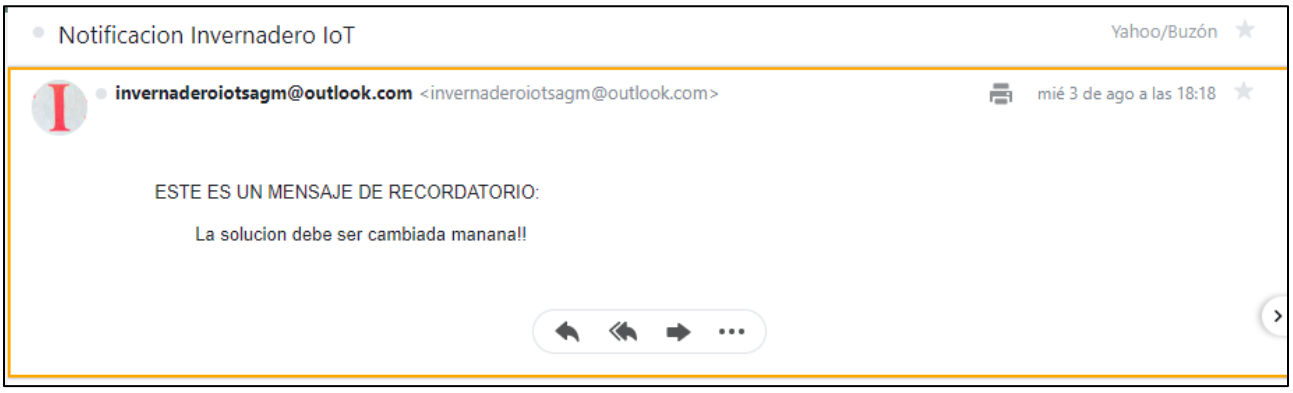
```
# El programa principal se ingresa a un ciclo infinito
while True:

        # Se crea la variable que llevara la cuenta de los días transcurridos
        dia = 1
        # Con un ciclo while se crea el contador de 15 días
        while dia <= 15:

        print ("==DIAS DE USO DE LA SOLUCION NUTRITIVA==\n")
        # Se imprime en pantalla los dias que se ha usado la solución nutritiva
        print ("\tDias: ", dia, "\n")
        # En el dia 1 se guarda la feha inicial y la fecha de cambio
        if dia == 1:
                FECHA = date.today()
                AUMENTO = timedelta(15)
                FINAL = FECHA + AUMENTO
        # En el día 14 se envía mensaje notificando que falta un día para cambiar la solución nutritiva
        if dia == 14:
                print ("\tFalta un dia para el cambio de la solucion.\n")
                mensaje = """
                        ESTE ES UN MENSAJE DE RECORDATORIO:

                        La solucion debe ser cambiada manana!!

                        """
                mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
                server = smtplib.SMTP("smtp.office365.com", 587)
                server.starttls()
                server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
                server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
                server.quit()
                print ("\t\tCorreo de notificacion enviado.\n")
        # En el dia 15 nuevamente se manda una alerta por correo, en este caso para informar que se tiene que cambiar la solución
        if (dia == 15):
                print ("\tLa solucion debe ser cambiada HOY!!.\n")
                mensaje = """
                        ESTE ES UN MENSAJE DE ALERTA:

                        La solucion debe ser cambiada HOY!!

                        """
                mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
                server = smtplib.SMTP("smtp.office365.com", 587)
                server.starttls()
                server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
                server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
                server.quit()
                print ("\t\tCorreo de notificacion enviado.\n")
        # El valor de la variable dia es guardado en MySQL para ser desplegado en la página de internet
        mycursor = mibd.cursor()
        # La informacion se guarda en la tabla llamada ContadorNFT
        sql = "INSERT INTO ContadorNFT(Dias, INICIAL, FINAL) VALUES (%s, %s, %s)"
        val = (dia, FECHA, FINAL)
        mycursor.execute(sql, val)
        # Al guardar terminar todo el proceso del código, se suma un día a la variable
        dia = dia+1
        # El paso final es poner en modo reposo al programa por un día completo, es decir, 86,400 segundos
        time.sleep(86400)
```

Una vez terminada la programación del Sistema, se procedió a ejecutarlo, y quedó de la siguiente manera:

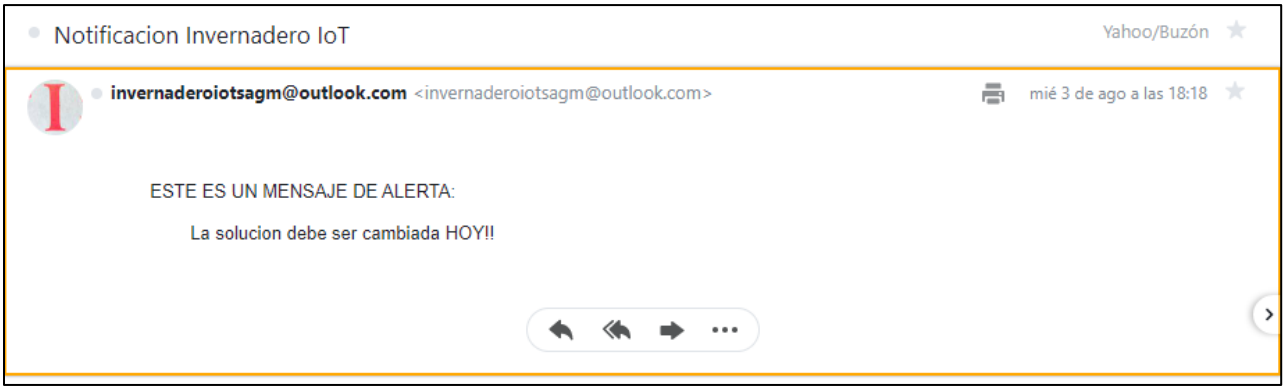
```
==DIAS DE USO DE LA SOLUCION NUTRITIVA==

        Dias:  12

==DIAS DE USO DE LA SOLUCION NUTRITIVA==

        Dias:  13

==DIAS DE USO DE LA SOLUCION NUTRITIVA==

        Dias:  14

        Falta un dia para el cambio de la solucion.

                Correo de notificacion enviado.

==DIAS DE USO DE LA SOLUCION NUTRITIVA==

        Dias:  15

        La solucion debe ser cambiada HOY!!.

                Correo de notificacion enviado.
```

10.3.5.1: Prueba del Sistema que contabiliza los días de uso de la solución nutritiva

En este caso se decidió mostrar solo a partir del día 12 debido a que el mensaje mostrado en pantalla del día 1 al 11 es similar al mostrado en el día 12. En la imagen se observa que en el día 14 apareció el mensaje en pantalla informando que faltaba un día para el cambio de la solución nutritiva, y en el día 15 el aviso informaba que ese día se tenía que cambiar la solución. En ambos días, el 14 y el 15, se observa que se recibieron notificaciones por correo electrónico y a continuación son presentadas:



10.3.5.2: Correo de notificación en el día 14

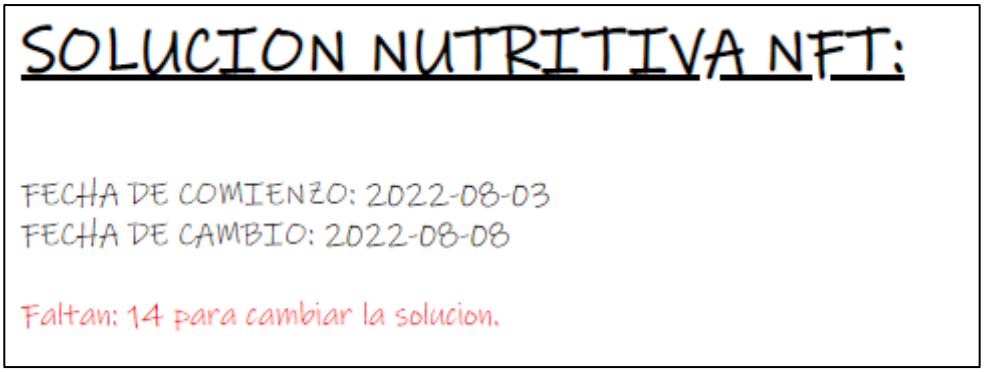


10.3.5.3: Correo de notificación en el día 15

Para finalizar con este sistema, se muestra el código HTML diseñado para mostrar los días de uso de la solución nutritiva:

```
<html>
        <head>
                # Nombre de la pestaña de la página Web
                <title>CONTADOR</title>
                <meta charset="utf-8">
        </head>
        <body>
                <?php
                        $link = mysqli_connect("localhost", "pi", "TesisIoT2021");
                        mysqli_select_db($link, "Invernadero");
                        # El código se conecta a la tabla llamada ContadorNFT
                        $result = mysqli_query($link, "SELECT * FROM ContadorNFT");
                        $renglones = mysqli_num_rows($result);
                        $ultima = $renglones-1;
                        # El puntero se posiciona en la última línea de información
                        mysqli_data_seek($result, $ultima);
                        $extraido = mysqli_fetch_array($result);
                        echo "<h1><u>SOLUCION NUTRITIVA NFT:</u></h1><br>";
                        # Se pide extraer las fechas de inicio y cambio de la solución nutritiva
                        echo "FECHA DE COMIENZO: ".$extraido["INICIAL"]."<br>";
                        echo "FECHA DE CAMBIO: ".$extraido["FINAL"]."<br><br>";
                        # Se imprime aviso de los días que faltan para cambiar la solución nutritiva
                        $ultimo = 15- $extraido["Dias"];
                        echo "<font color = red>Faltan: ".$ultimo." para cambiar la solucion.</font>";
                ?>
        </body>
</html>
```

El código se probó en un navegador Web accediendo a la dirección 192.168.100.42/CONTADOR.php:



10.3.5.4: Pagina Web del contador de cambio de la solución nutritiva

En la imagen se observa de color rojo el recordatorio de los días faltantes para el cambio de la solución, el cual se actualiza automáticamente cada día, mientras que las fechas se cambiaran igualmente en automático cada 15 días.

Con este último sistema se termina el tema 10.3 en el que se diseñó y se realizaron pruebas con cada sistema por separado para comprobar su funcionamiento. Todos los sistemas y el control de los parámetros serán unidos en el próximo tema, en el cual también se mostrarán las fotografías de los sistemas trabajando ya instalados en el invernadero, en esta sección se omitió esta parte debido a que el funcionamiento de los sistemas antes y después de ser instalados seria el mismo y las imágenes hubieran sido bastante similares.

10.4 SISTEMA HIDROPÓNICO INTELIGENTE

Para iniciar la parte final del proyecto primero se reestructuraron todos los sistemas, esto debido a que cada sistema se trabajó por separado sin tener en cuenta los pines GPIO usados ni tener en consideración los verdaderos tamaños de algunos elementos. Por ejemplo, la altura de los contenedores de agua.

Para comenzar se presenta una tabla en la cual se muestran los pines utilizados para unir todos los proyectos. El objetivo de esta tabla fue evitar utilizar dos veces el mismo pin, además de que sirvió de guía al momento de las conexiones:

PIN GPIO	CONFIGURACION	ELEMENTO CONECTADO
2	Entrada	Sensor DHT22
3	Salida	Ventiladores
4	Salida	Bomba para riego
5	Salida	Tira LED
6	Entrada	Sensor LDR
12	Entrada	Echo de Sensor ultrasónico 1
13	Salida	Trigger de Sensor ultrasónico 1
16	Salida	Trigger de Sensor ultrasónico 2
17	Entrada	Echo de Sensor ultrasónico 2
18	Salida	Bomba para NFT

También, fue necesario definir los valores finales de cada uno de los sistemas para ser configurados en sus respectivos programas Python.

SISTEMA DE TEMPERATURA Y HUMEDAD:

Rango de Temperatura:	10 – 25 [°C]
Rango de Humedad:	50 – 75 [%]
Tiempo de tira LED encendida para calentar el invernadero antes de medir la temperatura nuevamente:	5 [minutos] = 300 [s]
Tiempo de riego y ventiladores encendidos para enfriar el invernadero antes de medir nuevamente la temperatura:	1 [minuto] = 60 [s]

Tiempo de ventiladores encendidos para eliminar el exceso de humedad dentro del invernadero:	5 [minutos] = 300 [s]
Tiempo de riego encendido para aumentar la humedad al interior del invernadero:	1 [minuto] = 60 [s]
Tiempo de espera para solicitar nuevamente la temperatura y la humedad:	15 [minutos] = 900 [s]

SISTEMA DE ILUMINACIÓN:

Horario de funcionamiento de la iluminación artificial:	**08:00 – 18:00**

SISTEMA DE NIVEL DE AGUA EN LOS CONTENEDORES:

Altura de tubo de cama capilar:	80 [cm]
Altura de contenedor de riego NFT:	20 [cm]

En este sistema además de reconfigurar las distancias de los contenedores, también se tuvo que realizar una modificación en el código Python. La modificación fue en la función que calculaba la altura de agua dentro del contenedor. Para las pruebas se utilizaron contenedores con la misma altura y en la tabla anterior se observa que para la instalación real se utilizaron contenedores con alturas diferentes. En el programa de Python de las pruebas la altura se pasó a la función como una constante debido a que era la misma. Para solucionarlo se modificó la función y el valor de la altura se pasó a la función como un parámetro, de esta manera la función trabaja con cualquier altura pasada como parámetro. A continuación, se presenta el fragmento del código modificado:

```
def faltura(TxX, RxX, AltCont):

        GPIO.output(TxX, GPIO.LOW)
        time.sleep(0.0001)
        GPIO.output(TxX, GPIO.HIGH)

        while GPIO.input(RxX) == 0:

                INICIO = time.time()

        while GPIO.input(RxX) > 0:

                FIN = time.time()

        DURACION = FIN - INICIO
        DISTANCIA = (DURACION * 34300)/2
        ALTURA = AltCont - DISTANCIA
        return ALTURA
```

Lo mismo ocurrió con la función del cálculo del porcentaje de agua dentro de los contenedores:

```
def fporcentaje(VALTURA, AltCont):

        PORCENTAJE = (VALTURA*100)/AltCont
        return PORCENTAJE
```

Las modificaciones se verán más adelante cuando se muestren todos los códigos finales.

SISTEMA DE RIEGO NFT:

Intervalo de tiempo entre riegos:	1 [hora] = 3600 [s]
Tiempo de riego:	15 [minutos] = 900 [s]
Horario de riego:	08:00 – 18:00

SISTEMA DE CONTADOR DE SOLUCION NUTRITIVA:

Número de días de uso de la solución nutritiva:	15 [días]
Tiempo de espera del contador para sumar un día:	24 [horas] = 86400 [s]

Después de definir que pines se iban a utilizar y conocer los tiempos de espera y funcionamiento de los sistemas, se procedió a configurar esta información en los códigos Python finales, quedando de la siguiente manera. Al igual que en la modificación anterior de la función del Sistema de Nivel de Agua, se presenta el código resaltando las modificaciones realizadas.

SISTEMA DE TEMPERATURA Y HUMEDAD:

```
import adafruit_dht
import board
import time
import RPi.GPIO as GPIO
import mysql.connector
import smtplib

SENSOR = adafruit_dht.DHT22(board.D2, use_pulseio = False)
mibd = mysql.connector.connect(host = "localhost", user = "pi", passwd = "TesisIoT2021", database = "Invernadero")
ACTVEN = 3
ACTFOCO = 5
ACTRIEGO = 4

GPIO.setmode(GPIO.BCM)
GPIO.setup(ACTVEN, GPIO.OUT)
GPIO.setup(ACTFOCO, GPIO.OUT)
GPIO.setup(ACTRIEGO, GPIO.OUT)

def temp():
    try:
        TEMPERATURA = SENSOR.temperature
        if TEMPERATURA is None:
            TEMPERATURA = 20
        print ("\nTemperatura: ", TEMPERATURA, "°")
        if TEMPERATURA < 10:
            print ("\n\tTemperatura baja, iniciando proceso de calentamiento, por favor espere...")
            mensaje = """
                ESTE ES UN  MENSAJE DE ALERTA:

                LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION, SE DESCRIBE LA FALLA:

                PARAMETRO AFECTADO: Temperatura
                ESTADO: Muy baja
                ACCIONES TOMADAS:Sistema de calefaccion activado

                POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.
```

```
        GRACIAS!

        """
    mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
    server = smtplib.SMTP("smtp.office365.com", 587)
    server.starttls()
    server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
    server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
    server.quit()
    while TEMPERATURA < 10:
        try:
            GPIO.output(ACTFOCO, GPIO.LOW)
            TEMPERATURA = SENSOR.temperature
        except RuntimeError as error:
            pass
        time.sleep(300)
        print ("\n\tCalentando, por favor espere...")
    GPIO.output(ACTFOCO, GPIO.HIGH)
    print ("\nTemperatura: ", TEMPERATURA, "°")
    print ("\n\n\tLa temperatura se ha reestablecido\n")
    mensaje = """
        ESTE ES UN  MENSAJE DE ACTUALIZACION:

        LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

        GRACIAS!!

        """
    mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
    server = smtplib.SMTP("smtp.office365.com", 587)
    server.starttls()
    server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
    server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
    server.quit()
elif TEMPERATURA > 25:
    print ("\n\tTemperatura muy alta, iniciando proceso de enfriamiento, por favor espere...")
    mensaje = """
        ESTE ES UN  MENSAJE DE ALERTA:

        LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION, SE DESCRIBE LA FALLA:

        PARAMETRO AFECTADO: Temperatura
        ESTADO: Muy alta
        ACCIONES TOMADAS: Sistema de enfriamiento activado

        POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

        GRACIAS!

        """
    mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
    server = smtplib.SMTP("smtp.office365.com", 587)
    server.starttls()
    server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
    server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
    server.quit()
    while TEMPERATURA > 25:
        try:
            GPIO.output(ACTVEN, GPIO.LOW)
            GPIO.output(ACTRIEGO, GPIO.LOW)
            TEMPERATURA = SENSOR.temperature
        except RuntimeError as error:
            pass
        time.sleep(60)
        print ("\n\tEnfriando, por favor espere...")
    GPIO.output(ACTVEN, GPIO.HIGH)
    GPIO.output(ACTRIEGO, GPIO.HIGH)
    print ("Temperatura: ", TEMPERATURA, "°")
```

```
            print ("\n\n\tLa temperatura se ha reestablecido\n")
            mensaje = """
               ESTE ES UN  MENSAJE DE ACTUALIZACION:

               LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

               GRACIAS!!

               """
            mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
            server = smtplib.SMTP("smtp.office365.com", 587)
            server.starttls()
            server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
            server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
            server.quit()
         else:
            print ("\n\tLa temperatura es correcta")
         return TEMPERATURA
      except RuntimeError as error:
         pass

def hum():
   try:
      HUMEDAD = SENSOR.humidity
      if HUMEDAD is None:
         HUMEDAD = 60
      print ("\nHumedad: ", HUMEDAD, "%")
      if HUMEDAD < 50:
         print ("\n\tHumedad muy baja, iniciando riego, por favor espere...")
         mensaje = """
            ESTE ES UN  MENSAJE DE ALERTA:

            LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION SE DESCRIBE LA FALLA:

            PARAMETRO AFECTADO:Humedad
            ESTADO: Muy baja
            ACCIONES TOMADAS: Sistema de riego activado

            POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

            GRACIAS!

            """
         mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
         server = smtplib.SMTP("smtp.office365.com", 587)
         server.starttls()
         server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
         server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
         server.quit()
         while HUMEDAD < 50:
            try:
               GPIO.output(ACTRIEGO, GPIO.LOW)
               HUMEDAD = SENSOR.humidity
            except RuntimeError as error:
               pass
            time.sleep(60)
            print ("\n\tReajustando humedad, por favor espere...")
         GPIO.output(ACTRIEGO, GPIO.HIGH)
         print ("Humedad: ", HUMEDAD, "%")
         print ("\n\n\tLa humedad se ha reestablecido\n")
         mensaje = """
            ESTE ES UN  MENSAJE DE ACTUALIZACION:

            LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

            GRACIAS!!

            """
```

```
            mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
            server = smtplib.SMTP("smtp.office365.com", 587)
            server.starttls()
            server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
            server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
            server.quit()
        elif HUMEDAD > 75:
            print ("\n\tHumedad muy alta, activando ventilación, por favor espere...")
            mensaje = """
                ESTE ES UN  MENSAJE DE ALERTA:

                LOS PARAMETROS DE INVERNADERO NO SON LOS ADECUADOS, A CONTINUACION SE DESCRIBE LA FALLA:

                PARAMETRO AFECTADO: Humedad
                ESTADO: Muy alta
                ACCIONES TOMADAS: Sistema de ventilacion activado

                POR FAVOR ESPERE, SERA NOTIFICADO CUANDO LOS PARAMETROS SE REESTABLEZCAN.

                GRACIAS!

                """
            mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
            server = smtplib.SMTP("smtp.office365.com", 587)
            server.starttls()
            server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
            server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
            server.quit()
            while HUMEDAD > 75:
                try:
                    GPIO.output(ACTVEN, GPIO.LOW)
                    HUMEDAD = SENSOR.humidity
                except RuntimeError as error:
                    pass
                time.sleep(300)
                print ("\n\tReajustando humedad, por favor espere...")
            GPIO.output(ACTVEN, GPIO.HIGH)
            print ("Humedad: ", HUMEDAD, "%")
            print ("\n\n\tLa humedad se ha reestablecido\n")
            mensaje = """
                ESTE ES UN  MENSAJE DE ACTUALIZACION:

                LOS PARAMETROS DEL INVERNADERO HAN REGRESADO A LA NORMALIDAD.

                GRACIAS!!

                """
            mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
            server = smtplib.SMTP("smtp.office365.com", 587)
            server.starttls()
            server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
            server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
            server.quit()
        else:
            print ("\n\tEl porcentaje de humedad es correcto")
        return HUMEDAD
    except RuntimeError as error:
        pass

while True:

            try:
                        GPIO.output(ACTVEN, GPIO.HIGH)
                        GPIO.output(ACTFOCO, GPIO.HIGH)
                        GPIO.output(ACTRIEGO, GPIO.HIGH)
                        asunto = "Notificacion Invernadero IoT"

                        TEMPERATURA = temp()
```

```
                if TEMPERATURA is None:
                TEMPERATURA = 20
                HUMEDAD = hum()
                if HUMEDAD is None:
                HUMEDAD = 60
                FECHA = time.strftime("%y/%m/%d")
                HORA = time.strftime("%H:%M:%S")

                mycursor = mibd.cursor()
                sql = "INSERT INTO DHT22(Fecha, Hora, Temperatura, Humedad) VALUES (%s, %s, %s, %s)"
                val = (FECHA, HORA, TEMPERATURA, HUMEDAD)
                mycursor.execute(sql, val)
                mibd.commit()

                mensaje = """
                ESTE ES UN  MENSAJE DE NOTIFICACION:

                TODOS LOS PARAMETROS SE ENCUENTRAN EN ORDEN!

                """
                mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
                server = smtplib.SMTP("smtp.office365.com", 587)
                server.starttls()
                server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
                 server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
                server.quit()

                time.sleep(900)

        except OSError as error:
                pass
```

SISTEMA DE ILUMINACION:

```
from gpiozero import LightSensor
import RPi.GPIO as GPIO
from datetime import datetime
import smtplib
import time

ldr = LightSensor(6)
ACTFOCO = 5

GPIO.setmode(GPIO.BCM)
GPIO.setup(ACTFOCO, GPIO.OUT)

while True:

   GPIO.output(ACTFOCO, GPIO.HIGH)

   if ldr.value > 0:

      print ("\nNo hay luz natural...\n")
      FECHA = datetime.now()

      if (FECHA.hour > 8) and (FECHA.hour < 18):

         print ("\tAccion: Se activa luz artificial!...\n")
         asunto = "Notificacion Invernadero IoT"
         mensaje = """
            ESTE ES UN MENSAJE DE NOTIFICACION:

            SE HA ACTIVADO LA ILUMINACION ARTIFICIAL!

            """
         mensaje = "Subject : {} \n\n {}".format(asunto, mensaje)
```

```
            server = smtplib.SMTP("smtp.office365.com", 587)
            server.starttls()
            server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
            server.sendmail("invernaderoiotsagm@outlook.com", "marcko_antonio_966@yahoo.com", mensaje)
            server.quit()
            print ("\tMensaje de alerta enviado\n")

            while ldr.value > 0:

                GPIO.output(ACTFOCO, GPIO.LOW)

            GPIO.output(ACTFOCO, GPIO.HIGH)

        else:

            print ("\tMotivo: Es de noche!!\n")

    else:

        print ("Es de dia!\n")
```

SISTEMA DE NIVEL DE AGUA EN LOS CONTENEDORES:

```
import RPi.GPIO as GPIO
import time
import mysql.connector
import smtplib

GPIO.setmode(GPIO.BCM)
mibd = mysql.connector.connect(host = "localhost", user = "pi", passwd = "TesisIoT2021", database = "Invernadero")

Tx1 = 13
Rx1 = 12
Tx2 = 16
Rx2 = 17

GPIO.setup(Rx1, GPIO.IN)
GPIO.setup(Rx2, GPIO.IN)
GPIO.setup(Tx1, GPIO.OUT)
GPIO.setup(Tx2, GPIO.OUT)

def faltura(TxX, RxX, AltCont):

    GPIO.output(TxX, GPIO.LOW)
    time.sleep(0.0001)
    GPIO.output(TxX, GPIO.HIGH)

    while GPIO.input(RxX) == 0:

        INICIO = time.time()

    while GPIO.input(RxX) > 0:

        FIN = time.time()

    DURACION = FIN - INICIO
    DISTANCIA = (DURACION * 34300)/2
    ALTURA = AltCont - DISTANCIA
    return ALTURA

def fporcentaje(VALTURAX, AltCont):

    PORCENTAJE = (VALTURAX*100)/AltCont
    return PORCENTAJE

while True:
```

```
print ("== NIVEL DE AGUA ==\n")
asunto = "Notificacion invernadero IoT"
print( "\n\tCONTENEDOR 1 (CAMA CULTIVO CAPILAR)\n")
VALTURA1 = faltura(Tx1, Rx1, 80)
VALTURA = VALTURA1
VPORCENTAJE1 = fporcentaje(VALTURA, 80)
VP1 = round(VPORCENTAJE1, 2)
print ("\n\t\tEl nivel del agua se encuentra al ", VP1, "% de su capacidad.\n")
if VP1 < 10:

    print("\t\tNivel de agua muy bajo, es necesario rellenar el contenedor!\n")
    mensaje = """
            ESTE ES UN MENSAJE DE ALERTA:

                ESTADO: Nivel de agua menor al 10%
                CONTENEDOR: Tubo de PVC
                RECOMENCACION: Rellenar urgentemente
        """
    mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
    server = smtplib.SMTP("smtp.office365.com", 587)
    server.starttls()
    server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
    server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
    server.quit()
    print ("\t\tCorreo de notificacion enviado.\n")

else:

    print("\t\tLos niveles de agua son correctos!\n")


print( "\n\tCONTENEDOR 2 (Hidroponia NTF):\n")
VALTURA2 = faltura(Tx2, Rx2, 35)
VPORCENTAJE2 = fporcentaje(VALTURA2, 35)
VP2 = round(VPORCENTAJE2, 2)
print ("\n\t\tEl nivel del agua se encuentra al ", VP2, "% de su capacidad.\n")
if VP2 < 10:

    print("\t\tNivel de agua muy bajo, es necesario rellenar el contenedor!\n")
    mensaje = """
            ESTE ES UN MENSAJE DE ALERTA:

                ESTE ES UN MENSAJE DE ALERTA:
                ESTADO: Nivel de agua menor al 10%
                CONTENEDOR: Contenedor hidroponia NTF
                RECOMENCACION: Rellenar urgentemente
        """
    mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
    server = smtplib.SMTP("smtp.office365.com", 587)
    server.starttls()
    server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
    server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
    server.quit()
    print ("\t\tCorreo de notificacion enviado.\n")

else:

    print("\t\tLos niveles de agua son correctos!\n")

FECHA = time.strftime("%y/%m/%d")
HORA = time.strftime("%H:%M:%S")
mycursor = mibd.cursor()
sql = "INSERT INTO NivelAgua(Fecha, Hora, Contenedor1, Contenedor2) VALUES (%s, %s, %s, %s)"
val = (FECHA, HORA, VP1, VP2)
mycursor.execute(sql, val)
mibd.commit()
print("\nDatos guardados correctamente.\n")
```

```
    time.sleep(86400)
```

SISTEMA DE RIEGO NFT:

```
import RPi.GPIO as GPIO
from datetime import datetime
import time
import mysql.connector

RiegoNFT = 18

GPIO.setmode(GPIO.BCM)
GPIO.setup(RiegoNFT, GPIO.OUT)
mibd = mysql.connector.connect(host = "localhost", user = "pi", passwd = "TesisIoT2021", database = "Invernadero")

while True:

    GPIO.output(RiegoNFT, GPIO.HIGH)
    fecha = datetime.now()
    print("Hora: ", fecha.strftime("%H:%M:%S"))
    if (fecha.hour >= 8) and (fecha.hour <= 18):

        print ("Se han activado el riego automatico y recirculacion, por favor espere...\n")
        GPIO.output(RiegoNFT, GPIO.LOW)
        time.sleep(900)
        GPIO.output(RiegoNFT, GPIO.HIGH)
        mycursor = mibd.cursor()
        FECHA = fecha.date()
        HORA = fecha.strftime("%H:%M:%S")
        sql = "INSERT INTO RiegoNFT(Fecha, Hora) VALUES (%s, %s)"
        val = (FECHA, HORA)
        mycursor.execute(sql, val)
        mibd.commit()
        print("\nDatos guardados correctamente.\n")
        print ("El riego y recirculacion han finalizado. Gracias!\n")

    else:

        print ("El riego solo se activa durante el dia. Gracias!\n")

    time.sleep(3600)
```

SISTEMA DE CONTADOR DE SOLUCION NUTRITIVA:

```
import smtplib
import mysql.connector
import time
from datetime import date, timedelta

asunto = "Notificacion Invernadero IoT"
mibd = mysql.connector.connect(host = "localhost", user = "pi", passwd = "TesisIoT2021", database = "Invernadero")

while True:
    dia = 1
    while dia <= 15:

        print ("==DIAS DE USO DE LA SOLUCION NUTRITIVA==\n")
        print ("\tDias: ", dia, "\n")
        if dia == 1:
            FECHA = date.today()
            AUMENTO = timedelta(15)
            FINAL = FECHA + AUMENTO
        if dia == 14:
```

```
    print ("\tFalta un dia para el cambio de la solucion.\n")
    mensaje = """
          ESTE ES UN MENSAJE DE RECORDATORIO:

                La solucion debe ser cambiada manana!!

       """
    mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
    server = smtplib.SMTP("smtp.office365.com", 587)
    server.starttls()
    server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
    server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
    server.quit()
    print ("\t\tCorreo de notificacion enviado.\n")
if (dia == 15):
    print ("\tLa solucion debe ser cambiada HOY!!.\n")
    mensaje = """
          ESTE ES UN MENSAJE DE ALERTA:

                La solucion debe ser cambiada HOY!!

       """
    mensaje = "Subject: {}\n\n {}".format(asunto, mensaje)
    server = smtplib.SMTP("smtp.office365.com", 587)
    server.starttls()
    server.login("invernaderoiotsagm@outlook.com", "TesisIoT2021")
    server.sendmail("invernaderoiotsagm@outlook.com","marcko_antonio_966@yahoo.com", mensaje)
    server.quit()
    print ("\t\tCorreo de notificacion enviado.\n")
mycursor = mibd.cursor()
sql = "INSERT INTO ContadorNFT(Dias, INICIAL, FINAL) VALUES (%s, %s, %s)"
val = (dia, FECHA, FINAL)
mycursor.execute(sql, val)
mibd.commit()
dia = dia+1
time.sleep(86400)
```

Para unir todos los códigos Python en uno solo se utilizaron los *Hilos*, una herramienta que permite multiplexa varios procesos para simular que se están realizando al mismo tiempo. Esta herramienta es muy útil en los dispositivos actuales para realizar varias tareas en paralelo.

Para utilizar los *Hilos* en Python se debe importar la biblioteca *threading* y después se configura como un hilo cada proceso que se quiere realizar.

Para el código final además de utilizar los hilos también se utilizó la función *exec(open())*, que ejecuta scripts de otros archivos. De esta manera el código final llamará a los códigos de cada sistema y los ejecutará, esto evitará reescribir en un solo archivo todos los códigos Python escritos anteriormente, lo que facilitará el manejo de los sistemas y las reconfiguraciones futuras.

Para utilizar la función *exec(open())* fue necesario tener a la mano los nombres de cada uno de los códigos de los sistemas. En la siguiente tabla se muestra el sistema y el nombre final de su código Python.

Sistema de Temperatura y Humedad:	TYHFINAL.py
Sistema de Iluminación:	LDRFINAL.py
Sistema de Nivel de Agua en los Contenedores:	NIVELAGUAFINAL.py
Sistema de Riego NFT:	RIEGONFTFINAL.py
Sistema de Contador de Solución Nutritiva:	CONTADORFINAL.py

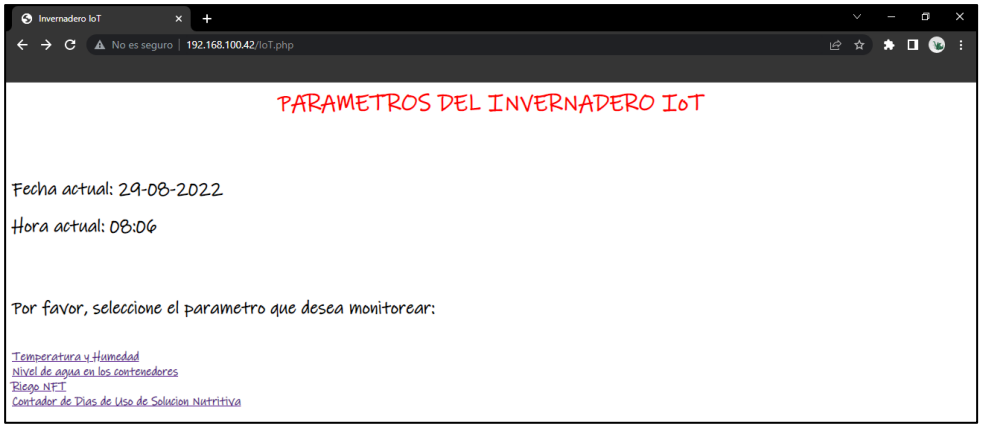
```
# Importación de todas las bibliotecas utilizadas por el Sistema completo
import threading
import adafruit_dht
import board
import time
import RPi.GPIO as GPIO
import mysql.connector
import smtplib
from gpiozero import LightSensor
from datetime import datetime

# Configuración de todos los pines utilizados
SENSOR = adafruit_dht.DHT22(board.D2, use_pulseio = False)
ACTVEN = 3
ACTRIEGO = 4
ACTFOCO = 5
ldr = LightSensor(6)
Rx1 = 12
Tx1 = 13
Tx2 = 16
Rx2 = 17
RiegoNFT = 18

# Debido a que todos los correos de notificación tendrán el mismo asunto, se declara una variable global
asunto = "Notificacion Invernadero IoT"

# Cada uno de los sistemas se abre a través de una función
def f1():
        exec(open("TYHFINAL.py").read())

def f2():
        exec(open("LDRFINAL.py").read())

def f3():
        exec(open("NIVELAGUAFINAL.py").read())

def f4():
        exec(open("RIEGONFTFINAL.py").read())

def f5():
        exec(open("CONTADORFINAL.py").read())

# Las funciones para abrir los códigos de los sistemas se convierten en hilos
l1 = threading.Thread(target = f1)
l2 = threading.Thread(target = f2)
l3 = threading.Thread(target = f3)
l4 = threading.Thread(target = f4)
l5 = threading.Thread(target = f5)

# Inicialización de los hilos, a partir de este momento comienzan a ejecutarse en “paralelo”
l1.start()
l2.start()
l3.start()
l4.start()
l5.start()

# Bloque de los hilos de llamada
l1.join()
l2.join()
l3.join()
l4.join()
l5.join()
```

Algo similar ocurrió con el código HTML final para monitorizar los parámetros a través de una página Web. En este caso, se creó una página Web principal con hipervínculos hacia los parámetros. A continuación, se muestra el código:

```
<html>

    <head>
        <title>Invernadero IoT</title>
        <meta charset="utf-8">
    </head>
    <body>
        <?php
            date_default_timezone_set("America/Mexico_City");
            echo "<center><h1><font color = red>PARAMETROS DEL INVERNADERO IoT</font></h1></center></br></br>";
            $fecha = date("d-m-Y");
            $hora = date("h:i");
            echo "<h2>Fecha actual: ".$fecha."</h2>";
            echo "<h2>Hora actual: ".$hora."</h2></br></br>";
            echo "<h2>Por favor, seleccione el parametro que desea monitorear:</h2></br>";
        ?>
          # Hipervínculos hacia los parámetros
        <a href="HyTFinal.php"><b>Temperatura y Humedad</b></a></br>
        <a href="NIVEL.php"><b>Nivel de agua en los contenedores</b></a></br>
        <a href="NFT.php"><b>Riego NFT</b></a></br>
        <a href="CONTADOR.php"><b>Contador de Dias de Uso de Solucion Nutritiva</b></a></br>
    </body>
</html>
```

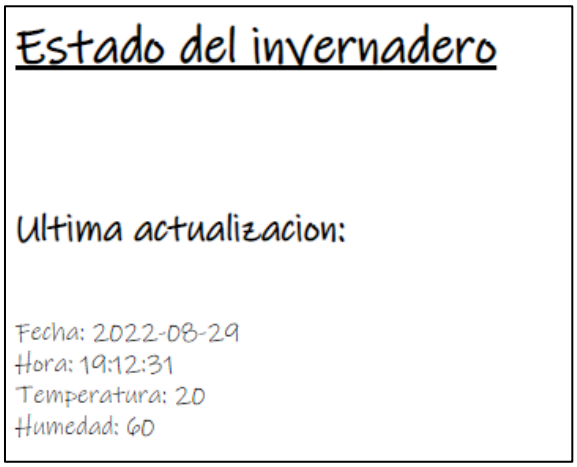
El código fue guardado en la ruta 192.168.100.42/IoT.php.

Para comprobar el funcionamiento del Sistema Final se ejecutó el código Python y después comenzó el monitoreo a través de la página Web y las notificaciones por correo electrónico. Las siguientes imágenes muestran el resultado final.

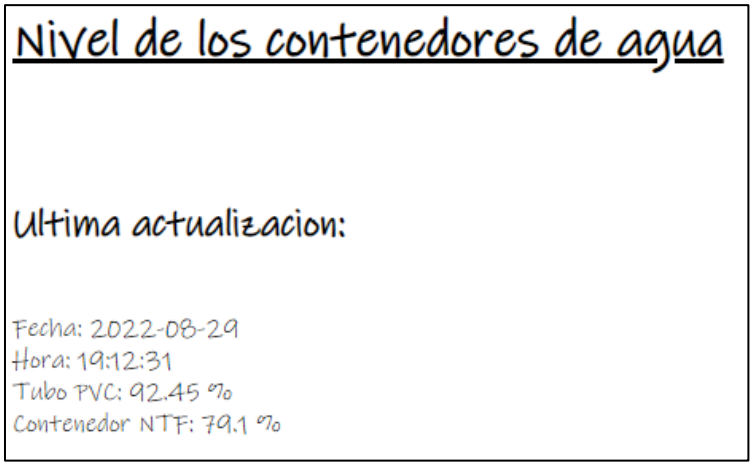


10.4.1: Página principal de monitoreo

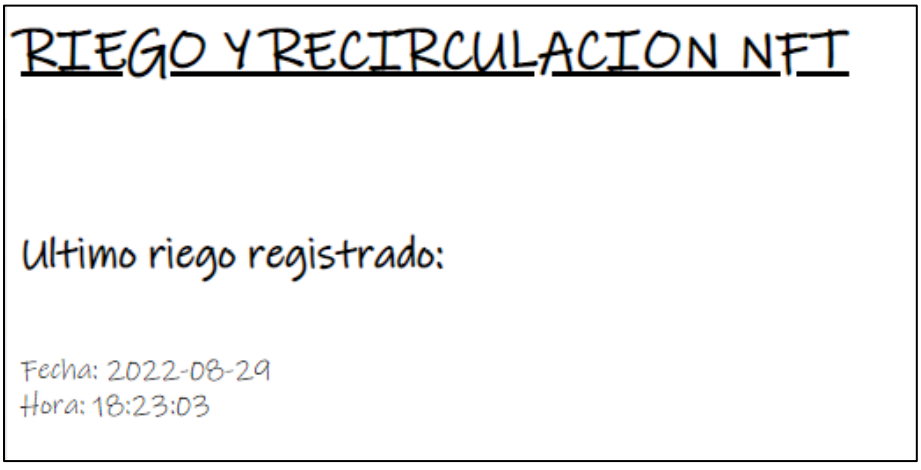
Haciendo clic en cada uno de los parámetros de obtuvo lo siguiente:



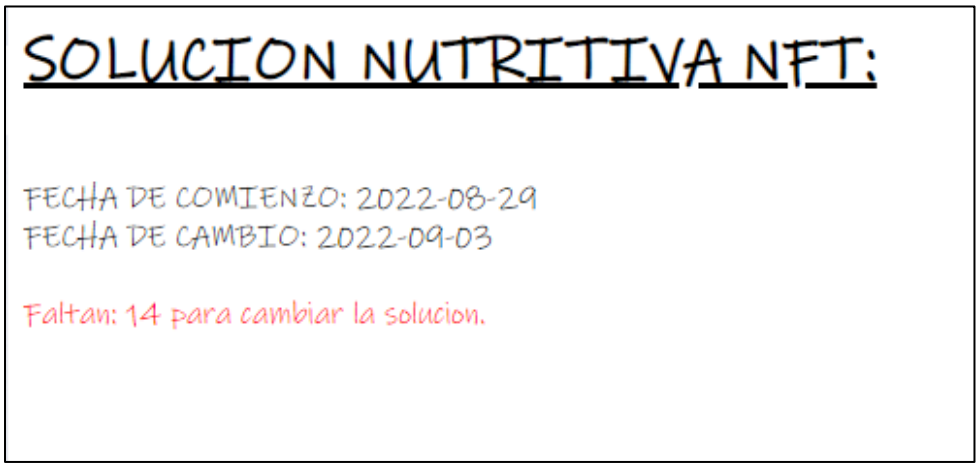
10.4.2: Monitoreo de Humedad y Temperatura



10.4.3: Monitoreo de los Niveles de Agua en los Contenedores



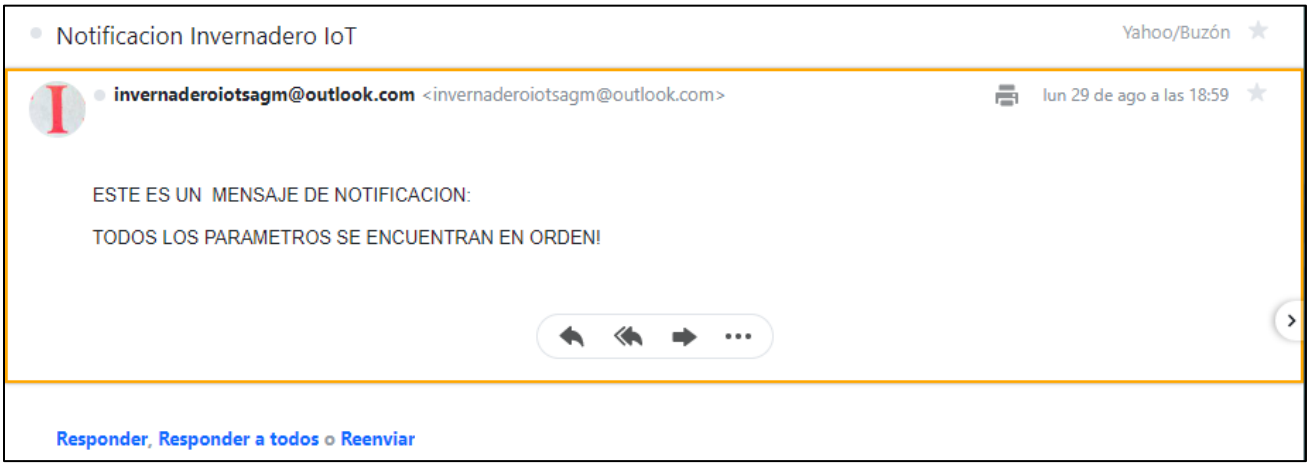
10.4.4: Monitoreo del Riego NFT



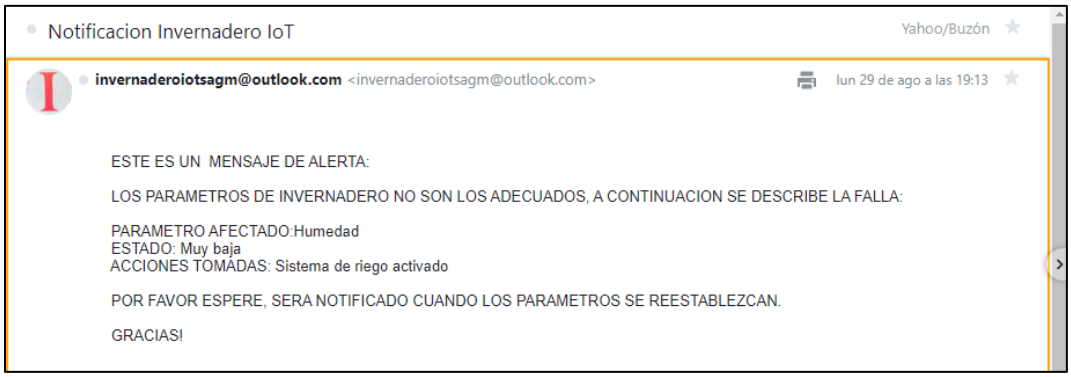
10.4.5: Monitoreo de la solución nutritiva

También, se recibieron algunas notificaciones por correo electrónico sobre el estado del Invernadero:

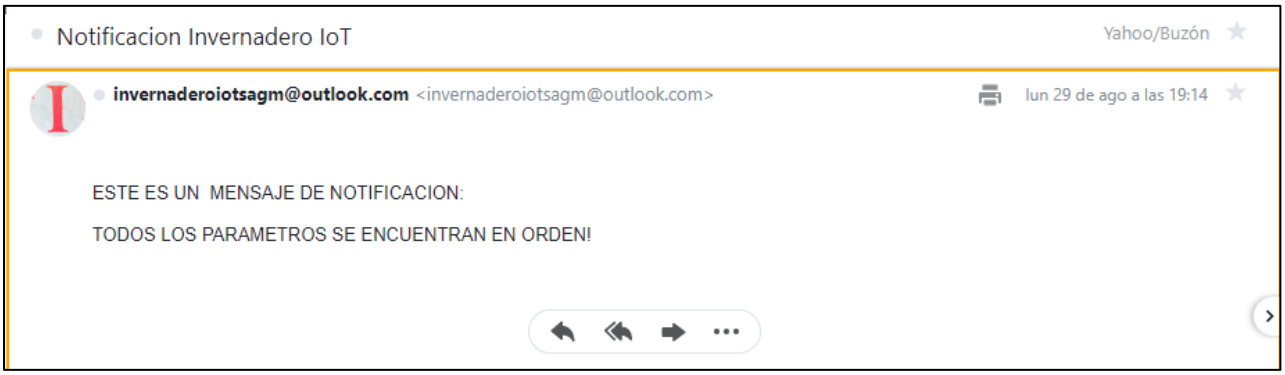
10.4.6: Correo de notificación de alta temperatura



10.4.7: Correo de notificación después de reestablecer la temperatura



10.4.8: Correo de notificación de baja humedad



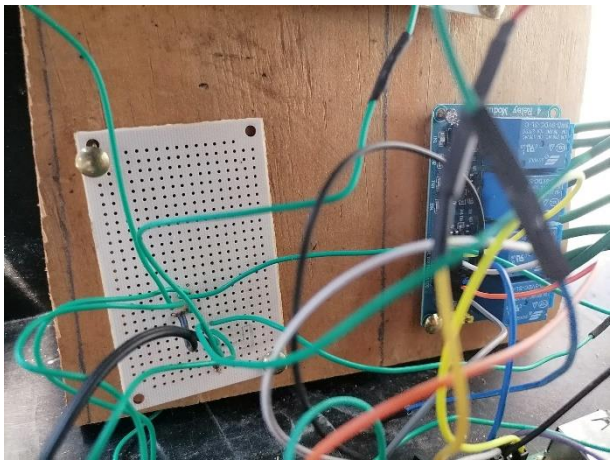
10.4.9: Correo de notificación después de restablecer el porcentaje de humedad

Por último, y después de comprobar el funcionamiento del sistema, se agregan fotos de las conexiones finales sobre el invernadero, de algunos actuadores funcionando y el invernadero final con las primeras plantas.

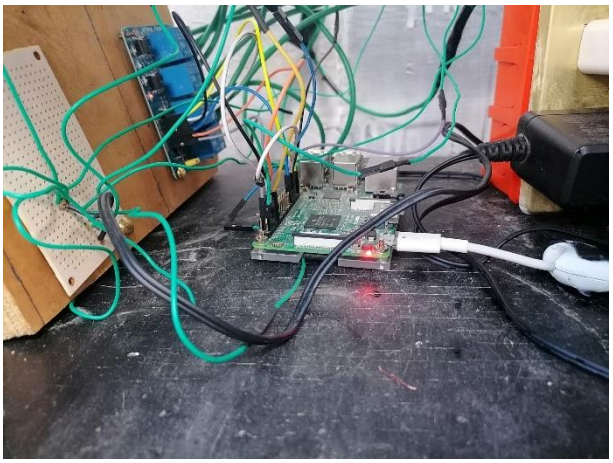
Para aumentar la seguridad de la instalación se agregó un centro de carga, en el cual se instaló una pastilla termomagnética de 120 [V] a 15 [A].

10.4.10: Centro de carga

Para las conexiones finales se utilizó una tabla fenólica para conectar una fuente de 5 [V], y de la fuente, los dispositivos que requerían ese voltaje. Por seguridad, la tabla fue empotrada a una tabla de madera, al igual que el módulo de relevadores para mantenerse fijos en un solo lugar y evitar problemas y accidentes. También, se colocó un contacto para la Raspberry y el eliminador de la fuente de 5 [V].



10.4.11: Tabla de instalación de la fuente de 5 [V] y los relés



10.4.12: Raspberry con conexiones finales

Para finalizar se agregan algunos actuadores funcionando. No se colocan todos debido a que su funcionamiento no se puede apreciar correctamente en fotografías.

Tira LED:

10.4.13: Funcionamiento de la tira LED

Sensores:

10.4.14: Funcionamiento de los sensores DTH22 y LDR

Ventiladores:

10.4.15: Funcionamiento de los ventiladores

El invernadero con los cultivos se ve de la siguiente manera:

10.4.16: Cama de cultivo capilar

10.4.17: Cultivo hidropónico NFT

Con los resultados obtenidos, y mostrados en este capítulo se demuestra que se logró el objetivo del proyecto. Mas adelante, en las conclusiones se hará un análisis de los resultados mostrados.

Capítulo 11: Costo del sistema

A continuación, se presenta una tabla de gastos, esta tabla presenta el material en el orden en el que fue utilizado en el capítulo 10:

Nombre:	No. De piezas:	Precio por pieza:	Precio Total:
Raspberry Pi 3 B	1	$ 2,400	$ 2,400
Cargador Raspberry	1	$ 139	$ 139
Memoria MicroSD 32 [GB]	1	$ 120	$ 120
Teclado	1	$ 210	$ 210
Ratón	1	$ 175	$ 175
Monitor Raspberry	1	$ 760	$ 760
Conector HDMI – HDMI	1	$ 50	$ 50
Tiras de madera	20	$ 35	$ 700
Escuadras	50	$ 1	$ 50
Impermeabilizante 4 [L]	1	$ 265	$ 265
Pijas 1 [”]	300	$ 0.2	$ 60
Clavos 1 [”]	20	$ 0.3	$ 6
Hoja de triplay	1	$ 300	$ 300
Plástico *“Frost King”*	1	$ 390	$ 390
Bisagras para puerta	2	$ 15	$ 30
Pasador	1	$ 10	$ 10
Tabla 35 [cm] de ancho por 2.5 [m] largo	2	$ 100	$ 200
Plástico negro [m]	4	$ 40	$ 160
Grava [bulto]	1	$ 50	$ 50
Tubo PVC 2 [“] por metro	2	$ 35	$ 70
Malla mosquitera por metro	4	$ 13	$ 52
Tierra para sembrar por costal	2	$ 100	$ 100
Sensor DTH22	1	$ 115	$ 75
Módulo de 4 relés	1	$ 75	$ 75
LEDs	5	$ 2	$ 10
Jumpers por paquete	1	$ 79	$ 79
Protoboard tamaño estándar	1	$ 90	$ 90
Protoboard pequeña	1	$ 55	$ 55
Resistencias varias	20	$ 0.25	$ 5
Ventilador 110 V	2	$ 709	$ 1,418
Cable calibre 12	100 [m]	$ 400	$ 400
Bomba sumergible de 3 [m] de elevación	2	$ 298	$ 298
Tubo de riego diámetro 4 [mm]	10 [m]	$ 209	$ 209
Boquillas de rociado	10	$ 199	$ 199
Tira LED de calentamiento e iluminación	3 [m]	$ 291	$ 291
Sensor LDR	1	$ 50	$ 50
Sensor ultrasónico HC-SR04	2	$ 35	$ 70
Tubo PVC 3’’ por metro	8	$ 60	$ 480
Tubo PVC ½’ por metro	3	$ 15	$ 45

Tapones PVC 3''	8	$ 20	$ 160
Cople ½'	4	$ 3.50	$ 14
Codos ½'	8	$ 2	$ 16
Manguera ½' por metro	2	$ 15	$ 30
Polines	2	$ 50	$ 100
Pegamento PVC	1	$ 55	$ 55
Cautín	1	$ 79	$ 79
Contenedor de 20 [L]	1	$ 100	$ 100
Solución Nutritiva 1 [kg]	1	$ 229	$ 229
Paquete de vasos de plástico	1	$ 17	$ 17
Varias semillas para sembrar	10	$ 10	$ 100
Cable para Raspberry	100 [m]	$ 1.8	$ 180
Paquete de grapas para cable	1	$ 30	$ 30
Tabla fenólica perforada	1	$ 5	$ 5
Centro de carga para una pastilla	1	$ 90	$ 90
Pastilla termomagnética 110 [V] a 15 [A]	1	$ 40	$ 40
Cinta de aislar	2	$ 20	$ 40
TOTAL:			$ 11, 431
SUBTOTAL:			$ 5, 480

En la tabla de costos se puede observar la lista de materiales incluidos, así como el precio. También, se observan algunos materiales resaltados en color rojo, con los que ya se contaba debido a que fueron adquiridos a lo largo de la carrera o era material que tenía en casa.

Al final de la tabla se tienen dos renglones, el primero corresponde al costo total del material, incluyendo el precio del que ya se tenía, mientras que la última columna es la suma de los costos del material que fue comprado especialmente para este proyecto.

Conclusiones

Al inicio del proyecto se establecieron los objetivos de este trabajo, que después de analizar el desarrollo y los resultados obtenidos se puede concluir que fueron cumplidos. El principal objetivo fue crear un sistema inteligente dentro de un invernadero, capaz de monitorear y mantener estables los parámetros necesarios para el desarrollo de un cultivo hidropónico. Las imágenes mostradas en los capítulos 9 y 10 en los que se desarrollaron los sistemas, se probaron y posteriormente se instalaron, se puede observar que el sistema completo es capaz de mantener estables los parámetros de temperatura, humedad e iluminación. De acuerdo a los resultados obtenidos, al llegar a una temperatura de 30 [°C] el sistema tarda alrededor de 18 [minutos] en bajar a menos de 25 [°C] la temperatura activando los ventiladores y el riego, para el caso del porcentaje de humedad el sistema actúa más rápido, si baja la humedad alrededor del 40 [%], se activa el riego y aproximadamente 2 [minutos] después incrementa al 50 [%]. Gracias al sistema de iluminación que activa o desactiva la tira LED según la cantidad de luz en el ambiente las plantas cuentan con cantidad de luz suficiente para desarrollar la fotosíntesis durante el día y descansando durante la noche.

Además de los parámetros ya mencionados, el sistema es capaz de completar el autorriego del cultivo NFT, generando una recirculación en los canales de PVC durante 15 minutos cada hora. El agua en el contenedor de autorriego y en la cama capilar siempre esta monitoreada para evitar una escasez, y en caso de vaciarse alguno de los contenedores se envía un correo electrónico al usuario. También, se cuenta con un contador de días para el uso de la solución nutritiva NFT, debido a que después de 15 días la solución deja de ser útil y se tiene que cambiar.

En caso de alguna falla o algún evento extraordinario, el sistema tiene la inteligencia suficiente para notificar vía correo electrónico, además de que se cuenta con la información del estado del invernadero en bases de datos que pueden ser consultadas en cualquier momento en la página Web diseñada especialmente para el proyecto.

Entre los objetivos también se encontraba la idea de crear un invernadero dirigido especialmente a los habitantes de la Ciudad de México apoyándose en la tecnología con la que cuenta la ciudad y adaptándose a espacios pequeños dentro de los hogares. El proyecto fue realizado en un espacio libre de aproximadamente 5 [m^2] en la azotea y se utilizó el modem Wifi utilizado para proveer de internet al hogar, por lo que, con estas medidas, y aprovechando que buena parte de los habitantes de la CDMX cuentan con acceso a internet a través de un modem, el invernadero inteligente podría implementarse fácilmente en cualquier hogar.

En cuanto a lo económico, se buscó que el invernadero fuera de bajo costo para volverlo accesible a toda la población. En la tabla mostrada en el capítulo 11 se muestra la lista de materiales utilizados y el precio. En este caso se obtuvieron dos totales, uno que corresponde a la suma de todos los materiales empleados y otro más donde se eliminó el material que ya se tenía. Si se comparan ambas cifras totales con el precio de un invernadero inteligente en el mercado se podrá observar que el precio está muy por debajo del precio de un invernadero inteligente profesional, y que tan solo la estructura metálica de un invernadero pequeño sin tecnología ronda los 10,000 pesos. Si se quisiera minimizar aún más los costos se podría utilizar material reciclado para la construcción del invernadero como madera, plástico, metal, etc., teniendo únicamente que gastar en los dispositivos electrónicos.

El proyecto se considera innovador debido a que se utilizó una tarjeta Raspberry cuando en la mayoría de proyectos similares se utiliza Arduino, lo que permite una mayor velocidad en los procesos y decisiones

tomadas por el sistema, además de que puede realizar tareas más complejas. Debido a su rendimiento y velocidad, la Raspberry es ideal para comenzar con proyectos pequeños que poco a poco se pueden ir escalando y convertirse en proyectos industriales en los que el nivel de proceso es muy alto. La Raspberry cuenta con módulos integrados como Wifi y Bluetooth, además de fuentes de voltaje y bastantes pines que facilitan las conexiones al momento de implementar los proyectos.

Los objetivos alcanzados en la creación del Sistema Inteligente para un Invernadero Hidropónico fueron creados principalmente con los conocimientos adquiridos a lo largo de la carrera universitaria. Las materias cursadas me brindaron las bases teóricas y prácticas para poder aplicar mis conocimientos a una problemática de la vida real. Además de los conocimientos tecnológicos enfocados a mi carrera, la UNAM me dio la oportunidad de conocer sobre otros campos y tener una perspectiva más general de la sociedad y la vida. También, aprendí a conocer mis habilidades, como el autoaprendizaje y la investigación, que son bastante útiles para el desarrollo de nuevos proyectos.

Bibliografía

LIBROS, REVISTAS, ARTICULOS y TESIS:

[1] Izquierdo Juan, ***¿Qué es la Hidroponía?***, Chile, Oficina Regional de Producción Vegetal de la FAO, 2003.

[2] Beltrano José, ***Cultivo en Hidroponía***, Argentina, Universidad Nacional de la Plata, 2015.

[3] Inca Sánchez Saul Adrián, ***Tesis: Automatización y Control del Sistema NFT para Cultivo Hidropónico***, Escuela Profesional de Ingeniería Electrónica, Universidad Ricardo Palma, Perú, 2013.

[4] Resh H.M., ***Cultivos Hidropónicos***, 5ta Edición, España, Ediciones Mundi-Prensa, 2002.

[5] Evans Dave, ***How the Next Evolution of the Internet Is Changing Everything***, CISCO, Internet of Things White Paper.

[6] García Muelas Carlos, ***Proyecto: Integración de Redes Telemáticas IoT con Raspberry Pi***, Universidad Oberta de Cataluña, España.

[7] Ma Hua-Dong, ***Internet of Things: Objectives and Scientific Challenges***, Journal of Computer Science and Technology, China, 2011.

[8] Mathur Puneet, ***IoT Machine Learning Applications in Telecom, Energy, and Agriculture***, India, Apress, 2020.

[9] Salazar Jordi, ***Internet de las Cosas***, Europa, Erasmus+.

[10] Shovic John, ***Raspberry Pi IoT Projects***, EUA, Apress, 2016.

[11] Alcaraz Marcelo, ***Internet de las Cosas***, Paraguay, Universidad Católica Nuestra Señora de la Asunción.

[12] Rose, Karen, ***La Internet de las Cosas – Una breve reseña***, Internet Society, 2015.

[13] Corona Ramírez Leonel German, ***Sensores y Actuadores Aplicaciones con Arduino***, México, Instituto Politécnico Nacional, Grupo Editorial Patria, 2014.

[14] Pallás Areny Ramon, ***Sensores y Acondicionadores de Señal***, 4ta Edición, España MARCOBO, 2003.

[15] Neri Vela Rodolfo, ***Líneas de Transmisión***, México, McGraw-Hill, 1999.

[16] Hanes, David, ***IoT Fundamentals***, CISCO, 2017

[17] Watkiss Stewart, ***Learn Electronics with Raspberry Pi*****,** Reino Unido, Apress, 2016.

FUENTES DE INTERNET:

[I] ***Hidroponía, ¿Sabes qué es y cómo funciona?*****, Gobierno de México:**

https://www.gob.mx/siap/articulos/hidroponia-sabes-que-es-y-como-funciona

[II] ***Internet de las Cosas*****, Instituto de Ingeniería, UNAM:**

http://www.ii.unam.mx/es-mx/AlmacenDigital/CapsulasTI/Paginas/internetdelascosas.aspx

[III] ***Introducción al Internet de las Cosas*****, Redes de Ingeniería, Universidad Distrital "Francisco José de Caldas", Colombia:**

https://revistas.udistrital.edu.co/index.php/REDES/article/view/8505/10464

[IV] ***Beneficios y características del cultivo bajo plástico*****, Editorial Informativo Agrícola de México:**

https://mexico.infoagro.com/beneficios-y-caracteristicas-del-cultivo-bajoplastico/#:~:text=En%20su%20concepto%2C%20un%20invernadero,el%20desarrollo%20de%20las%20plantaciones

[V] ***Invernaderos de Plástico*****, tecnología y manejo. Ediciones multiprensa:**

https://ebookcentral.proquest.com/lib/bibliodgbsp/reader.action?docID=3206880

[VI] ***¿Qué es un Invernadero?*****, Ministerio del Trabajo y Economía Social, Gobierno de España:**

https://www.insst.es/-/-que-es-un-invernader-1

[VII] ***¿Qué es el sistema NFT?*****, Comercializadora Hydro Environment S.A. de C.V.:**

https://www.hydroenv.com.mx/catalogo/index.php?main_page=page&id=101

[VIII] ***Iluminación horticultural y floral: la importancia de la luz en los invernaderos*****, Lumínica: Revista Profesional de la Iluminación y el Alumbrado:**

https://www.revistaluminica.es/horticultural-invernaderos/#:~:text=La%20cantidad%20de%20luz%20diaria,para%20niveles%20altos%20de%20luz.

[IX] ***La luz dentro de los invernaderos*****, ACEA, Invernaderos para el mundo:**

https://acea.com.mx/articulos-tecnicos/alex-j-pacheco/60-los-factores-ambientales-y-su-influencia-en-invernaderos-313-la-luz-dentro-del-invernadero

[X] ***Monografías de productos agroalimentarios mexicano*****, Servicio de Información Agroalimentaria y Pesquera, Gobierno de México:**

https://www.gob.mx/siap/documentos/monografias

[XI] ***DTH11 Sensor with Raspberry Pi and Python*****, Peppe8o, Raspberry Pi, Arduino and Electronics made simple:**

https://peppe8o.com/using-raspberry-pi-with-dht11-temperature-and-humidity-sensor-and-python/?fbclid=IwAR3umJ9WrUGI-Jq40F9sPTONIc5fPINtMXZPxVM2zUf42iz2kDmb-tokGDo

[XII] ***Raspberry Pi Humidity Sensor using the DTH22*****, Emmet Young, PiMyLifeUp:**

https://pimylifeup.com/raspberry-pi-humidity-sensor-dht22/

[XIII] ***Módulos fotorresistivos LDR*****, TD Electrónica**

https://tdelectronica.com/producto/sensores/modulo-fotorresistivo-ldr/#:~:text=Descripci%C3%B3n,su%20resistencia%20es%20muy%20alta.

[XIV] ***Como conectar una Fotorresistencia (LDR) a Raspberry Pi*****, Saúl García, 330 Ohms:**

https://blog.330ohms.com/2020/02/06/ejemplo-fotoresistencia-ldr-raspberry-pi/

[XV] ***Encender una bomba de agua con Arduino*****, Luis Llamas, "Ingeniería, Informática y Diseño":**

https://www.luisllamas.es/bomba-de-agua-con-arduino/

[XVI] ***Arduino: Sensor ultrasónico HC-SR04*****, Blog de Tecnología, Gobierno de Canarias**

https://www3.gobiernodecanarias.org/medusa/ecoblog/fsancac/2018/02/06/arduino-sensor-ultrasonico-hc sr04/#:~:text=Su%20funcionamiento%20consiste%20en%20emitir,demora%20en%20llegar%20el%20eco.

[XVII] ***Como conectar un sensor ultrasónico HC-SR04 a Raspberry Pi*****, 330Ohms Tienda de Electrónica:**

https://blog.330ohms.com/2020/06/17/como-conectar-un-sensor-ultrasonico-a-raspberry-pi/

[XVIII] ***Using a Raspberry P distance sensor (Ultrasonic sensor HC-SR04)*****, Tutorials for RASPBERRY PI:**

https://tutorials-raspberrypi.com/raspberry-pi-ultrasonic-sensor-hc-sr04/

[XIX] ***Instalar Raspbian con Raspberry Pi Imager de forma sencilla*****, Joan Carles, Geekland Blog de Tecnología:**

https://geekland.eu/instalar-raspbian-con-raspberry-pi-imager/

[XX] ***Cómo configurar VNC en Raspberry Pi*****, Ionos Guía Digital:**

https://www.ionos.mx/digitalguide/servidores/configuracion/como-instalar-y-configurar-vnc-en-raspberry-pi/

[XXI] ***Como crear un servidor casero en 10 minutos con Raspberry Pi*****, Pedro Pablo Moral, El buen SEOmaritano:**

https://pedropablomoral.com/raspberrypi/proyectos/servidor-casero/

[XXII] ***Cómo usar un servidor web Raspberry Pi*****, Ionos Guía Digital:**

https://www.ionos.mx/digitalguide/servidores/configuracion/como-configurar-un-servidor-web-raspberry-pi-con-lamp/

[XXIII] ***Enviando emails con Python. Guía completa*****, Hever Rubio, Loop:**

https://loopgk.com/2021/03/05/enviando-emails-con-python-guia-completa/

OTRAS FUENTES CONSULTADAS:

1° Nixon Robin, ***Aprende PHP, MySQL y JavaScript***, 5ta Edición, España, MARCOMBO, 2018.

2° ***Cómo gestionar bases de datos en tu Rasberry Pi*****, Lorenzo Carbonell, Atareao:**

https://atareao.es/tutorial/raspberry-pi-primeros-pasos/gestionar-bases-de-datos/

3° ***Errores y excepciones*****, Python Software Foundation:**

https://docs.python.org/es/3/tutorial/errors.html

4° ***Guía de funciones de Python con ejemplos*****, freeCodeCamp:**

https://www.freecodecamp.org/espanol/news/guia-de-funciones-de-python-con-ejemplos/

5° ***Manual: num_rows*****, Grupo PHP:**

https://www.php.net/manual/es/mysqli-result.num-rows.php

6° ***Operaciones con fechas: Sumar y restar días en Python*****, Python de Principiante a Experto, J2LOGO:**

https://j2logo.com/operaciones-con-fechas-en-python/

7° ***Python MySQL INSERT INTO*****, BigData Analytics:**

https://bigdata-analytics.es/python/mysql-insert-into/

8° ***Sentencias Try y Except de Python: Como manejar excepciones de Python*****, freeCodeCamp:**

https://www.freecodecamp.org/espanol/news/sentencias-try-y-except-de-python-como-menejar-excepciones-en-python/

9° ***Uso de PHP con MySQL. Consultas básicas*****, Aprende a Programar: Didáctica y divulgación de la Programación:**

https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=612:php-consultas-mysql-mysqliconnect-selectdb-query-fetcharray-freeresult-close-ejemplos-cu00841b&catid=70&Itemid=193