



DEPARTAMENTO DE CIENCIAS DE DATOS E INFORMÁTICA

FACULTAD DE INGENIERÍA

UNIVERSIDAD DE PLAYA ANCHA

PROTOTIPO DE SISTEMA PARA MONITOREO, CONTROL Y
AUTOMATIZACIÓN DE VARIABLES MEDIOAMBIENTALES EN
INVERNADERO.

Proyecto de Título para optar al Título Ingeniero Informático
con mención en la Gestión de Información

YANKO DELGADO CANEO

Profesor Guía: Javier Castillo Allaria

San Felipe, Chile
2024

Agradecimientos

Agradezco a los profesores que estuvieron acompañándome en este proceso, al profesor Javier Castillo Allaria quien me brindó el financiamiento para poder llevar a cabo el proyecto, además de guiarme con sus recomendaciones. Al profesor Ricardo Contreras que siempre me apoyó y me orientó en diversos temas relacionados con el proceso de mi tesis.

Y por último agradezco profundamente a mi pareja que estuvo presente en todo este proceso apoyándome y dándome ánimos para seguir adelante y no decaer, y sin dejar de lado a toda mi familia que siempre creyó en mí y a la distancia me brindaba su apoyo y buenos deseos.

TABLA DE CONTENIDOS

Resumen	i
Abstract	ii
Introducción	iii
CAPÍTULO I.....	1
Formulación del Problema.....	1
1.1 Descripción del Problema.....	1
1.2 Estado del Arte	3
1.3 Objetivos del Proyecto.....	7
1.3.1 Objetivo General.....	7
1.3.2 Objetivos específicos.....	7
CAPÍTULO II.....	8
Metodología	8
2.1 Metodología de Software	9
CAPÍTULO III.....	12
Tecnologías de Desarrollo.....	12
3.1 Arduino IDE	12
3.2 Autodesk Fusion 360	12
3.4 PrusaSlicer	13
3.5 PHP.....	13
3.6 MySQL	13
3.7 JavaScript.....	13
3.8 Bootstrap 5.3	14
3.9 HTML5.....	14
CAPÍTULO IV	15
Desarrollo Propuesta.....	15
4.1 Requerimientos.....	15
4.1.1 Funcionales	15
4.1.2 No funcionales	15
4.2 Propuestas de Solución.....	15
4.3 Diagramas del Proyecto	27

4.3.1	Base de datos	27
4.3.2	Caso de uso	28
4.3.3	Diagramas de conexión.....	29
CAPÍTULO V		32
Desarrollo		32
5.1 Tecnologías Implementadas en el Desarrollo.....		32
5.2.1 Armado de hardware prototipo inicial.....		33
5.2.2 Creación códigos primer prototipo.....		36
5.3 Primera Iteración Y Retroalimentación		46
5.4 Mejoras Del Prototipo		48
5.4.1 Hardware		48
5.4.2 Creación nuevo código.....		50
5.4.3 Código recolector de datos.....		54
5.4.4 Configuración Raspberry pi		55
5.4.5 Creación modelo 3D		59
5.4.6 Pruebas Prototipo mejorado.....		63
5.5 Segunda Iteración Y Retroalimentación		65
5.6 Interfaz De Usuario		66
CAPÍTULO VI		77
Prototipo Final.....		77
CAPÍTULO VII		80
Conclusión.....		80
REFERENCIAS		82

TABLA DE ILUSTRACIONES

Ilustración 1 Metodología basada en prototipos.....	10
Ilustración 2 Sensor DHT11	16
Ilustración 3 Sensor DHT22.....	17
Ilustración 4 Sensor SEN0193	20
Ilustración 5 Sensor MQ-135	21
Ilustración 6 Sensor DS18B20	22
Ilustración 7 NodeMCU ESP32	23
Ilustración 8 Raspberry pi 2	24
Ilustración 9 Diagrama Base de datos.....	27
Ilustración 10 Diagrama caso de uso usuario general	29
Ilustración 11 Conexión DHT22.....	29
Ilustración 12 Conexión DS18B20	30
Ilustración 13 Conexión MQ-135	30
Ilustración 14 Conexión sensor capacitivo.....	31
Ilustración 15 Conexión sistema inalámbrico módulo de cuatro sensores	31
Ilustración 16 Pinout placa ESP32 devkit v1	34
Ilustración 17 Configuración Stepdown.....	36
Ilustración 18 Cuadro de preferencias para la instalación de package ESP32.....	37
Ilustración 19 URL package ESP32	37
Ilustración 20 Librerías sensor DHT22	39
Ilustración 21 Librería función HTTPClient ESP32.....	39
Ilustración 22 Librería sensor DS18B20	40
Ilustración 23 Librería función WiFi ESP32	40
Ilustración 24 Vista valores principales código sensores.....	41
Ilustración 25 Vista configuración función void setup()	42
Ilustración 26 Variables para calibración sensor capacitivo	43
Ilustración 27 Librerías sensor DS18B20.....	43
Ilustración 28 Instanciar clases OneWire y DallasTemperature	43
Ilustración 29 Código para envío de datos por método POST	44
Ilustración 30 código para obtener lecturas sensor capacitivo	45

Ilustración 31 Mejora de código para trabajar 4 sensores juntos	50
Ilustración 32 Implementación toma de lecturas para todos los sensores	52
Ilustración 33 Nueva estructuración para enviar datos por método POST	53
Ilustración 34 Código receptor y de inserción de datos	54
Ilustración 35 Referencia de donde sacar S.O Raspberry	55
Ilustración 36 Comando para actualizar paquetes de Raspberry	56
Ilustración 37 Comando para instalar apache	56
Ilustración 38 Comandos para obtener IP e instalar PHP	57
Ilustración 39 Comando para instalar MySQL.....	57
Ilustración 40 Comando para reiniciar servicios de apache	57
Ilustración 41 Comando para configuración inicial MySQL	58
Ilustración 42 Cuadro selección servidor web	59
Ilustración 43 Modelado 3D caja módulo compacto	60
Ilustración 44 Cuadro informativo del proceso de impresión	62
Ilustración 45 Tabla de lecturas	64
Ilustración 46 Carta de últimas lecturas	67
Ilustración 47 Cartas de últimas lecturas junto a sistema semáforo	67
Ilustración 48 Estructuración gráfico de promedios.....	68
Ilustración 49 Gráfico de promedios.....	69
Ilustración 50 Código gráficos principales	70
Ilustración 51 Consulta SQL para filtro de datos	70
Ilustración 52 Código gráficos con y sin filtro	71
Ilustración 53 Vista usuario página de gráficos históricos.....	71
Ilustración 54 Código de obtención de datos y creación tabla	72
Ilustración 55 Formulario creación archivo Excel	73
Ilustración 56 Tabla datos sensores con descarga.....	73
Ilustración 57 Iniciación código descarga	74
Ilustración 58 Preparación columnas de archivo Excel.....	75
Ilustración 59 Código contenido del archivo Excel	75
Ilustración 60 Parte final creación archivo Excel	76
Ilustración 61 Interfaz visual de inicio	78

Ilustración 62 Interfaz visual gráficos históricos	78
Ilustración 63 Interfaz visual tablas de los últimos 10 datos	79
Ilustración 64 Interfaz visual en dispositivo móvil	79

ÍNDICE DE TABLAS

Tabla 1 Requerimientos Funcionales	15
Tabla 2 Requerimientos no Funcionales	15
Tabla 3 Cuadro comparativo sensores DHT11 y DHT22	17
Tabla 4 Cuadro comparativo sensores capacitivo y SEN13322	18
Tabla 5 Características Raspberry pi 2	24
Tabla 6 Detalle para tabla de lecturas	27
Tabla 7 Detalle para tabla sensores	27
Tabla 8 Detalle para tabla módulos	28
Tabla 9 Softwares principales en el desarrollo	32

Resumen

El presente trabajo tiene como objetivo desarrollar un sistema automatizado para la recolección de variables ambientales, integrando sensores IoT con almacenamiento de datos en una base de datos estructurada. Este sistema está diseñado para facilitar el monitoreo continuo y confiable de parámetros como la temperatura, la humedad y otros indicadores ambientales relevantes para diferentes aplicaciones.

El método propuesto combina tecnologías de hardware y software. Los sensores IoT recolectan datos en tiempo real y los envían a un servidor central mediante protocolos de comunicación como lo puede ser MQTT o HTTP. Posteriormente, estos datos son procesados y almacenados en una base de datos estructurada, diseñada para optimizar la consulta y análisis posterior de la información. Además, se desarrolla una interfaz gráfica que permite la visualización de datos y el acceso remoto al sistema por parte de los usuarios.

Los resultados obtenidos muestran que el sistema es capaz de operar de manera eficiente, proporcionando datos precisos y confiables en tiempo real. La arquitectura modular facilita su escalabilidad y adaptación a diferentes entornos y necesidades específicas. Además, el uso de tecnologías de bajo costo contribuye a la viabilidad del proyecto para aplicaciones de amplio alcance.

En conclusión, el sistema desarrollado representa una solución efectiva para el monitoreo ambiental, integrando tecnología moderna con un enfoque práctico y accesible. Este enfoque tiene el potencial de ser aplicado en diversos ámbitos, promoviendo la sostenibilidad y la gestión eficiente de los recursos naturales.

Abstract

This work aims to develop an automated system for collecting environmental variables by integrating IoT sensors with structured database storage. The system is designed to facilitate continuous and reliable monitoring of parameters such as temperature, humidity, and other relevant environmental indicators for various applications.

The proposed method combines hardware and software technologies. IoT sensors collect real-time data and send it to a central server using communication protocols such as MQTT or HTTP. These data are then processed and stored in a structured database, optimized for subsequent query and analysis. Additionally, a graphical interface is developed to allow users remote access and data visualization.

The results show that the system can operate efficiently, providing accurate and reliable real-time data. Its modular architecture facilitates scalability and adaptation to different environments and specific needs. Moreover, the use of low-cost technologies enhances the project's feasibility for broader applications.

In conclusion, the developed system represents an effective solution for environmental monitoring, integrating modern technology with a practical and accessible approach. This solution has the potential to be applied across various fields, promoting sustainability and efficient natural resource management.

Introducción

El presente trabajo se enmarca en la necesidad de contar con herramientas tecnológicas que permitan monitorear y gestionar variables ambientales de manera eficiente. Este tema adquiere especial relevancia en un contexto donde el cambio climático y la sostenibilidad ambiental se han convertido en prioridades globales. La elección de este tema está motivada por la creciente demanda de sistemas que faciliten la recopilación, el almacenamiento y el análisis de datos ambientales. Elementos fundamentales para la toma de decisiones en diversas áreas.

La justificación de este trabajo radica en su contribución al desarrollo de soluciones tecnológicas que promuevan una mayor comprensión y control de las condiciones ambientales. Al implementar un sistema que permita obtener datos en tiempo real y almacenarlos en una base de datos, se busca facilitar el análisis y la generación de conocimiento que pueda ser aplicado para mejorar procesos operativos y promover la sostenibilidad.

El objetivo principal del proyecto es diseñar e implementar un sistema para la adquisición y almacenamiento de variables ambientales, utilizando tecnologías modernas y accesibles. Como objetivos específicos, se propone:

1. Seleccionar los sensores y dispositivos adecuados para la recopilación de datos.
2. Diseñar un esquema eficiente de base de datos para almacenar la información recopilada.
3. Desarrollar una interfaz que permita visualizar y analizar los datos de manera intuitiva.

Partiendo de la hipótesis de que un sistema integrado que combine hardware y software permitirá mejorar significativamente el acceso y análisis de

datos ambientales en tiempo real, el desarrollo de este trabajo se enfoca en probar esta afirmación mediante un prototipo funcional.

La metodología utilizada para resolver el problema incluye varias etapas clave. En primer lugar, se realiza un análisis de los requisitos técnicos y funcionales del sistema, seguido por la selección de herramientas y tecnologías adecuadas, como sensores de bajo costo y plataformas de desarrollo de software. Posteriormente, se lleva a cabo el diseño del sistema, que incluye diagramas de conexión de sensores y un esquema de base de datos. Finalmente, se implementa el sistema, probando su funcionalidad en un entorno controlado para garantizar que cumple con los objetivos planteados.

CAPÍTULO I

Formulación del Problema

1.1 Descripción del Problema

La gestión de las condiciones medioambientales es fundamental para garantizar el óptimo desarrollo de cultivos en invernaderos a baja escala. El control adecuado de variables como la temperatura y la humedad, tanto del suelo como del ambiente, así como la concentración de dióxido de carbono (CO_2), resulta crucial para alcanzar los objetivos establecidos.

En un invernadero estas variables afectan directamente el crecimiento, la productividad y la calidad de los cultivos. La falta de control puede llevar a condiciones desfavorables que impactan en la salud de las plantas, provocando pérdidas económicas y un uso ineficiente de recursos como el agua.

A pesar de su importancia, muchas instalaciones carecen de sistemas automatizados o adecuados para la monitorización y control en tiempo real de estas variables, lo que dificulta la toma de decisiones oportunas y precisas. Por ello, se identifica la necesidad de desarrollar una solución tecnológica que permita:

- Monitorear continuamente las condiciones ambientales y del suelo.
- Controlar de manera activa los parámetros críticos para mantener el entorno óptimo.
- Proveer datos accesibles para análisis, predicción y mejora en la gestión medioambiental.

Adicionalmente, se busca implementar esta solución con una inversión que no implique costos exorbitantes, haciendo posible su adopción tanto en entornos domésticos como agrícolas, promoviendo el acceso a tecnologías que

optimicen recursos y mejoren los resultados sin comprometer la viabilidad económica del proyecto.

1.2 Estado del Arte

En 2017, Sergio Ossa Duque publicó un artículo titulado Monitoreo y control de variables ambientales mediante una red inalámbrica para agricultura de precisión en invernaderos en la revista Vector. En este trabajo, el autor describe el diseño de una plataforma para el monitoreo remoto y el control de variables ambientales en invernaderos. Esta plataforma utiliza tecnologías basadas en redes inalámbricas que emplean el protocolo de comunicación Zigbee, integrando software y hardware de uso libre.

El proyecto incorporó diversos tipos de hardware, entre ellos:

- Arduino Nano, un microcontrolador compacto y económico.
- Sensores DHT11, diseñados para medir temperatura y humedad.
- Módulo XBee 2mW RPSMA – Series 2 Pro, utilizado para la comunicación inalámbrica mediante Zigbee.
- Arduino WiFi Shield CC3000, un módulo para proporcionar conectividad WiFi.
- Sensor de humedad de suelo SEN-13322, empleado para medir la humedad del suelo.

En las conclusiones, Ossa Duque destaca que estos sistemas son altamente accesibles debido a dos factores principales: su bajo costo y el uso de software libre. Este último elimina la necesidad de adquirir licencias comerciales, lo que reduce significativamente los costos asociados a su implementación.

En 2019, se presentó un proyecto en Colombia titulado Red de Sensores Inalámbricos para el Monitoreo de Variables Agroecológicas en Cultivos Bajo Invernadero. El objetivo principal fue implementar una red inalámbrica en cultivos bajo invernadero, utilizando placas NodeMCU (modelo ESP8266) junto con sensores DHT11 (para temperatura y humedad) y DS18B20 (sensor sumergible para temperatura).

Este sistema permitió registrar variables clave como la humedad, la temperatura y la luminosidad en plantas ornamentales cultivadas en macetas. Una de las conclusiones del estudio destacó que las redes de sensores inalámbricos son esenciales para aumentar la competitividad en el sector agropecuario, especialmente cuando se emplean tecnologías de bajo costo. Sin embargo, los investigadores alertaron sobre la necesidad de seleccionar componentes adecuados, ya que el sensor DHT11 tiende a sulfatarse rápidamente, afectando su desempeño y precisión. (Aguirre, Pira, Rodríguez, Mesa, & Castro, 2019)

Chile también ha desarrollado proyectos relacionados con la monitorización de variables en invernaderos. Uno de ellos fue realizado en la Universidad de Tarapacá, bajo el título Monitoreo y control de variables físicas en un invernadero a través del Internet de las Cosas. Este proyecto implementó una red inalámbrica utilizando placas NodeMCU8266, sensores DHT11 y DHT22, junto con una microcomputadora Raspberry Pi 3. El sistema permitió monitorear de forma remota diversas variables físicas, facilitando el control dentro del invernadero.

Aunque el proyecto destacó por el bajo costo y la accesibilidad de sus componentes, los investigadores señalaron que no se logró implementar completamente debido a las restricciones provocadas por la pandemia de COVID-19. Sin embargo, se concluyó que una conexión a internet estable es crucial para evitar errores en la transmisión de datos en redes de sensores inalámbricos. (Arancio Fernández, 2020)

Aunque se disponga de la tecnología necesaria para el monitoreo, es fundamental comprender la importancia de supervisar variables ambientales para garantizar el crecimiento y desarrollo óptimo de las plantas en un invernadero. Cada variable desempeña un papel esencial en el rendimiento de los cultivos.

Según el Informativo n.º 36 del Instituto de Investigaciones Agropecuarias (INIA), los principales factores limitantes para la producción hortícola incluyen la temperatura, la falta de luz, la humedad, el exceso de viento y el dióxido de carbono.

- Temperatura

- La temperatura es un factor determinante para el crecimiento y desarrollo de las plantas en un invernadero.
- Un control preciso de la temperatura permite adaptar las condiciones ambientales a las necesidades específicas de los cultivos.
- Las temperaturas extremas pueden afectar negativamente la fotosíntesis, la transpiración, la respiración y otros procesos vitales de las plantas.
- Además de manera estándar la temperatura que se espera en los invernaderos es de 22°C a 28°C durante el día y de 16°C a 18°C durante la noche.

- Humedad

- La humedad relativa del aire influye en los niveles de transpiración de las plantas y en la disponibilidad de agua en el sustrato.
- Un control adecuado de la humedad ayuda a prevenir enfermedades fúngicas y bacterianas, así como la pérdida excesiva de agua por las plantas.
- La humedad también puede afectar la calidad del producto final y la eficiencia del proceso de polinización.
- En el caso de la humedad del suelo el rango estándar es entre 60% y 80%, mientras que la humedad relativa o ambiental ronda entre el 50% y 70%.

- Dióxido de carbono

- El dióxido de carbono es esencial para la fotosíntesis de las plantas, ya que se utiliza como fuente de carbono para la producción de carbohidratos y otros compuestos orgánicos.
- Un suministro adecuado de CO₂ puede mejorar significativamente la tasa de crecimiento y producción de los cultivos en un invernadero.
- Monitorear y controlar los niveles de CO₂ ayuda a optimizar la eficiencia de la fotosíntesis y maximizar la producción de biomasa.
- Los niveles estándar de CO₂ en los invernaderos rondan entre las 800ppm y 1200ppm (partes por millón), ya que este rango maximiza la fotosíntesis y acelera el crecimiento de las plantas.

En resumen, un monitoreo constante de la temperatura, la humedad, el dióxido de carbono y la luz es indispensable para crear condiciones óptimas. Esto no solo favorece el crecimiento saludable de los cultivos, sino que también incrementa la producción y rentabilidad, minimizando riesgos de enfermedades y estrés vegetal. (Martínez, Farías, & Ruiz, 2020)

1.3 Objetivos del Proyecto

1.3.1 Objetivo General

Implementar un sistema tecnológico en Arduino y de bajo costo para la monitorización y el control automatizado de variables medioambientales esenciales, como la temperatura, la humedad del suelo y del aire, y la concentración de dióxido de carbono (CO₂), con el fin de controlar las condiciones de cultivo en invernaderos.

1.3.2 Objetivos específicos

- Diseñar e implementar un sistema de monitoreo inalámbrico que permita capturar y transmitir lecturas de temperatura, humedad (ambiental y del suelo) y concentración de dióxido de carbono (CO₂) en tiempo real, utilizando sensores adecuados.
- Desarrollar un sistema para la recepción de datos en PHP, que procese las lecturas transmitidas por el sistema inalámbrico y las prepare para ser almacenadas en una base de datos.
- Diseñar una interfaz informativa de las condiciones del invernadero.
- Crear una base de datos que almacene las lecturas medioambientales capturadas, permitiendo un acceso organizado a los datos históricos para su posterior análisis.

CAPÍTULO II

Metodología

Al comenzar el desarrollo del proyecto, se harán una serie de pasos, empezando con las necesidades requeridas las cuales serán derivadas de otros estudios que tuvieron como objetivo la biorremediación de los suelos, por lo tanto, las variables medioambientales que se heredarán son la temperatura ambiental, humedad de suelo y humedad ambiental, incorporándose a estas luego de lo investigado en el estado del arte para un mejor monitoreo la concentración de dióxido de carbono y temperatura del suelo, requiriendo su visualización, gestión y almacenamiento.

Para lograr la toma de requerimientos, se efectuarán reuniones vía online, en donde, se deben comentar los objetivos que se buscan alcanzar. Al estar trabajando en la creación de un sistema electrónico se debe tener en consideración que para poner a prueba los requerimientos, se debe hacer de forma práctica, por lo tanto, se elaborarán diagramas de conexión que permitirán visualizar cómo funcionará el hardware para solucionar los requerimientos principales del proyecto.

Con el objetivo de cumplir con los requerimientos, se tomará de forma prioritaria las funcionalidades principales del sistema, luego se tomarán en cuenta las funcionalidades secundarias.

Hay varias metodologías de software que pueden utilizarse en el desarrollo de un proyecto, en este específicamente se utilizará la llamada metodología basada en prototipos, la cual será definida y explicada en el siguiente punto.

2.1 Metodología de Software

La metodología de software utilizada en el desarrollo del proyecto será la basada en prototipos, esta fue elegida por la cantidad de ventajas que entrega, unas de ellas es que permite reducir riesgos, entre otras palabras permite detectar y corregir problemas antes de llegar a fases avanzadas del proyecto, además se puede hacer una validación rápida de ideas, ya que, las ideas iniciales pueden parecer prometedoras, pero hasta que se prueban con un prototipo funcional, es difícil asegurarse de su viabilidad. Otras de las ventajas que tiene es la mejor comprensión de los requerimientos y debido a su rápido accionar es posible tener un feedback más temprano del cliente.

El paradigma de hacer prototipos comienza con la comunicación. Usted se reúne con otros participantes para definir los objetivos generales del software, identifica cualesquiera requerimientos que conozca y detecta las áreas en las que es imprescindible una mayor definición. Se planea rápidamente una iteración para hacer el prototipo, y se lleva a cabo el modelado (en forma de un “diseño rápido”). Éste se centra en la representación de aquellos aspectos del software que serán visibles para los usuarios finales (por ejemplo, disposición de la interfaz humana o formatos de la pantalla de salida). El diseño rápido lleva a la construcción de un prototipo. Éste se entrega y es evaluado por los participantes, que dan retroalimentación para mejorar los requerimientos. La iteración ocurre a medida que el prototipo es afinado para satisfacer las necesidades de distintos participantes, y al mismo tiempo le permite a usted entender mejor lo que se necesita hacer. (Pressman, 2010, p.37)

Para poder utilizar la metodología basada en prototipo cada iteración es implementada en alguna función fundamental en el desarrollo, considerando esto, las iteraciones tendrán como base 5 etapas, siendo:

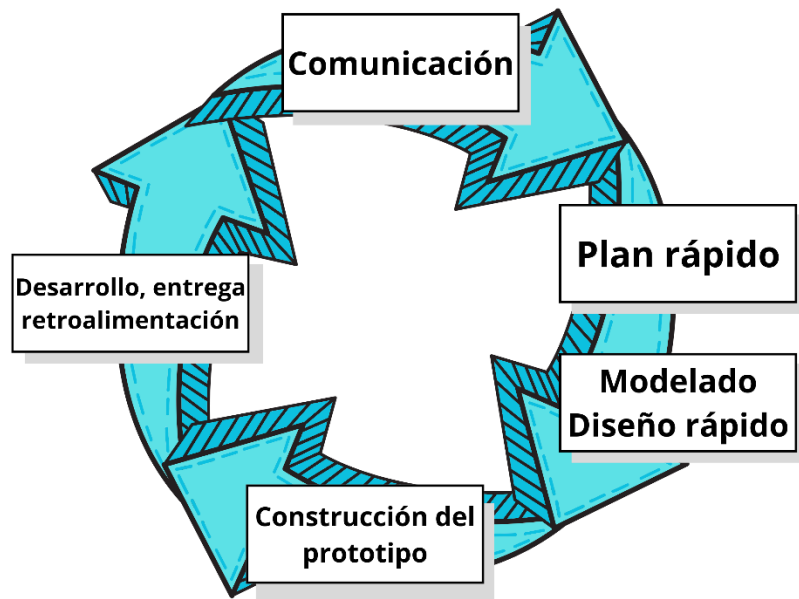


Ilustración 1 Metodología basada en prototipos

Comunicación: En este caso la comunicación se dará mediante conferencias online, donde siempre se mantendrá al tanto del objetivo principal que se quiere lograr en el proyecto, sumando a ello las características y funciones que deben de estar presentes en el desarrollo, además el objetivo de esta etapa es tener siempre informado en cuanto a la información que se entregue de la retroalimentación de cada prototipo.

Plan rápido: En esta etapa se buscará formular una idea que sea principalmente funcional y que sea capaz de responder a la necesidad que se requiere, además se enfocará en los recursos que serán necesarios para el desarrollo de dicho prototipo, ya sea, software o hardware.

Modelado rápido: Considerando que algunas de las funciones principales del proyecto ya se tenían planificadas desde antes de empezar con la metodología, ya se contaba con algunos diagramas de conexión enfocados a la electrónica que se tenía pensado utilizar, los nuevos modelos

que se crearán durante la metodología serán relacionados a la base de datos y los objetivos que puedan surgir a través de la retroalimentación.

Construcción del prototipo: En la etapa de construcción se programará el código necesario para las funciones y se creará el hardware necesario para el prototipo, además se harán pruebas que ayudarán a encontrar errores de funcionamiento que deban ser corregidos.

Desarrollo, entrega retroalimentación: En esta etapa el prototipo creado se mostrará por medio de conferencias online junto a sus pruebas para que sean evaluadas por las personas involucradas en el proyecto, para de este modo poder obtener una retroalimentación sobre si se ha podido cubrir con lo solicitado o se deben hacer cambios para obtener un mejor resultado que satisfaga el requerimiento solicitado.

CAPÍTULO III

Tecnologías de Desarrollo

3.1 Arduino IDE

Acrónimo de “Arduino Integrated Development Environment” traducido al español como entorno de desarrollo integrado de Arduino, es una herramienta de software que permite a los usuarios escribir, compilar y cargar código en placas de Arduino. Este entorno incluye un editor de texto para escribir el código, un área de mensajes para mostrar información y una consola de texto para la depuración.

3.2 Autodesk Fusion 360

Es una plataforma de software de modelado 3D que integra herramientas de CAD (diseño asistido por computadora), CAM (fabricación asistida por computadora), CAE (análisis asistido por computadora) y diseño de circuitos impresos (PCB). Esta herramienta está basada en la nube, lo que permite a los usuarios colaborar en tiempo real y acceder a sus proyectos desde cualquier lugar.

3.3 Fritzing

Es un software de diseño electrónico gratuito que permite a los usuarios crear circuitos virtuales, documentar proyectos y exportar archivos para la fabricación de placas de circuito impreso (PCB). Está diseñado para facilitar el proceso de diseño electrónico, siendo especialmente útil para diseñadores, artistas, educadores y entusiastas de la electrónica.

3.4 PrusaSlicer

Es un software de código abierto diseñado para la preparación de modelos 3D para impresión en 3D. Este programa permite a los usuarios "cortar" modelos 3D, generando el código G necesario para que las impresoras 3D puedan reproducir el objeto.

3.5 PHP

Es un lenguaje de programación interpretado, por lo tanto, este no es compilado y es utilizado en el desarrollo web para dar dinamismo a las páginas. Este lenguaje es uno de los pioneros en lograr una integración casi total con HTML. Su característica es que el código tiene etiquetas de apertura y de cierre.

3.6 MySQL

Sistema de gestión de bases de datos con código abierto, esta tecnología permite a todos los usuarios acceso a su código fuente. Una de sus características es la utilización de bases de datos relacionales, es decir, usa tablas múltiples que se conectan entre sí para almacenar y organizar información, además utilizando su lenguaje de programación de consulta estructurado SQL permitirá gestionar, administrar y recuperar información en la base de datos.

3.7 JavaScript

Es un lenguaje de programación interpretado, por lo tanto, este no requiere de ser compilado y está diseñado para funcionar en un navegador. Es utilizado para implementar funciones al usuario y uno de sus principales objetivos es la integración de funciones interactivas a las páginas, al igual que

permite el añadido de efectos visuales que dan más dinamismo a las páginas web. Añade tipos estáticos y objetos basados en clases.

3.8 Bootstrap 5.3

Es el framework más popular para el desarrollo web, con enfoques de diseño responsivo. Actualmente, Bootstrap 5.3 ofrece mejoras significativas y mayor estabilidad en comparación con versiones anteriores, brindando así una ventaja para los desarrolladores que buscan crear sitios modernos y eficientes.

3.9 HTML5

Acrónimo de “Hyper Text Markup Language” traducido al español como lenguaje de hipertexto marcado, su función es utilizar el lenguaje de etiquetas para construir páginas web. Cuando nos referimos a HTML5 se habla de la quinta versión de HTML publicada, la cual es la más estable para los distintos navegadores y plataformas actuales. HTML5 es una unión de HTML, CSS y JS con el fin de lograr una diversidad y un mayor alcance en el desarrollo de las páginas web.

CAPÍTULO IV

Desarrollo Propuesta

4.1 Requerimientos

4.1.1 Funcionales

Tabla 1 Requerimientos Funcionales

Nro.	Descripción
RF-01	El sistema debe poder tomar lecturas de temperatura, humedad y concentración de dióxido de carbono.
RF-02	Los sensores deben enviar los datos de forma inalámbrica a una unidad de recopilación.
RF-03	Los datos obtenidos se deben guardar en una base de datos.
RF-04	El sistema recolecta los datos obtenidos y los ingresa a la base de datos cada 5 minutos.

4.1.2 No funcionales

Tabla 2 Requerimientos no Funcionales

Nro.	Descripción
RNF-01	Los sensores deben ser alimentados por medio de baterías.
RNF-02	Los registros deben tener hora y fecha de ingreso.
RNF-03	La electrónica creada debe tener una forma de contenerla.
RNF-04	Los datos deben ser fácilmente legibles.
RNF-05	La interfaz de usuario debe ser de fácil navegación.

4.2 Propuestas de Solución

Para dar solución se tiene como propuesta implementar un sistema de adquisición de datos por medio de sensores los cuales trabajarán en conjunto para medir temperatura, humedad y concentración de CO₂. Esto con el fin de crear un registro que permitirá dar seguimiento a las variables ambientales, poder controlar los parámetros de dichas variables y adicionalmente almacenar los registros en una base de datos para un futuro uso de ellos.

Para llevar a cabo esta propuesta se requiere trabajar con componentes electrónicos los cuales serían:

- Placa NodeMCU ESP32
- Sensor de temperatura y humedad DHT22
- Sensor capacitivo de humedad SEN0193
- Sensor para CO2 MQ-135
- Sensor de temperatura sumergible DS18B20
- Microcomputadora Raspberry PI 2

Dado que en el mercado existe una gran variedad de dispositivos electrónicos que nos permiten tomar lectura de estas variables, sin embargo, hay que tener en cuenta cuales son las cualidades de cada uno de ellos para saber qué artículos serán mejores para el trabajo que uno busca realizar.

Haciendo referencia a los proyectos anteriormente mencionados, en estos normalmente se hace uso del sensor DHT11 (ilustración 2), un sensor de temperatura y humedad ambiental. Se trata de una opción básica y económica, adecuada para proyectos de baja exigencia en precisión (Picuino, n.d.). Según su ficha técnica, presenta las siguientes características:

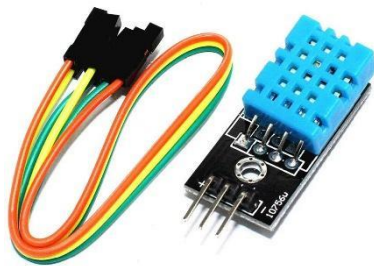


Ilustración 2 Sensor DHT11

- Tensión de alimentación de 3 a 5 voltios
- Corriente máxima de Alimentación 2.5 mA

- Rango de humedad relativa 20% a 80% con 5% de exactitud
- Rango de temperatura de 0 a 50°C con $\pm 2^{\circ}\text{C}$ de exactitud
- Velocidad de 1 medida por segundo
- Tamaño 15.5mm x 12mm x 5.5mm

Si bien las características de este sensor son bastante buenas, sin embargo, para este proyecto utilizaré el sensor DHT22 (ilustración 3), puesto que, por su nombre se puede ver que es una versión mejorada del sensor antes mencionado, para verificar lo que estoy proponiendo se mostrará una tabla (Tabla 3) comparando las características de los dos sensores para de esta forma evidenciar de mejor manera las características de cada uno.



Ilustración 3 Sensor DHT22

Tabla 3 Cuadro comparativo sensores DHT11 y DHT22

Características	Sensor DHT11	Sensor DHT22
Alimentación	3 a 5 voltios	3.3 a 6 voltios
Rango de humedad	20 – 80%	0 – 100%
Rango de Temperatura	0 a 50°C	-40 a 80°C
Exactitud humedad	5%	$\pm 2\%$
Exactitud Temperatura	$\pm 2^{\circ}\text{C}$	$\pm 0.5^{\circ}\text{C}$
Tamaño	15.5mm x 12mm x 5.5mm	14mm x 18mm x 5.5mm

Según Liu (2013), el sensor DHT22 ofrece un rango de medición de temperatura entre -40°C y 80°C , y una humedad relativa entre 0% y 100%,

con una precisión superior a la del modelo DHT11. De acuerdo con la tabla anteriormente presentada, esto confirma la elección del DHT22 como el sensor más adecuado, ya que proporciona un rango de trabajo más amplio tanto en temperatura como en humedad. Esta mejora en la exactitud y cobertura lo convierte en una opción preferente para proyectos donde es crucial obtener datos precisos y confiables, como en el monitoreo ambiental en invernaderos.

También es sumamente importante obtener los datos provenientes del suelo donde se encuentran los cultivos y para eso hare uso de un sensor capacitivo de humedad SEN0193 (ilustración 4), si bien como todo componente electrónico también existen otras opciones como es el caso del sensor SEN13322 utilizado en el proyecto “Monitoreo y control de variables ambientales mediante una red inalámbrica para agricultura de precisión en invernaderos”. Pensando en las características de cada uno la diferencia entre ellos radica principalmente en el tipo de funcionamiento junto a la precisión de la lectura, para entender mejor las diferencias entre ellos se presentará una tabla (Tabla 4) comparando las características de los dos sensores y así entender porque la decisión de optar por el sensor capacitivo.

Tabla 4 Cuadro comparativo sensores capacitivo y SEN13322

Características	Sensor Capacitivo	Sensor SEN13322
Funcionamiento	Capacitancia entre electrodos	Resistencia variable
Voltaje	3.3V – 5V	3.3V – 5V
Corriente de operación	5 mA	5 mA
Salida	Señal analógica (0-5V)	Señal analógica (0-5V)
Rango de medición	0-100%	0-100%
Precisión	+ -2%	+ -5%

Temperatura de Funcionamiento	-40°C a 85°C	-40°C a 85°C
Dimensiones	60 mm x 20mm x 5mm	50mm x 20mm x 5mm

Como se logra divisar en la tabla una de las principales diferencias es que el sensor capacitivo como indica su nombre trabaja con la capacitancia entre electrodos, en otras palabras, es la capacidad de dos placas metálicas para almacenar carga eléctrica cuando se aplica un voltaje. Cuando se aplica voltaje, una placa se carga positivamente y la otra negativamente, creando un campo eléctrico entre ellas que almacena energía. La capacitancia depende del área de las placas, la distancia entre ellas y el material que las separa. Lo que quiere decir que el sensor mide la capacidad del suelo para retener agua, ya que a medida que el suelo se humedece más, aumenta su capacidad para almacenar carga eléctrica, lo que se traduce en una mayor capacitancia medida por el sensor. Mientras el sensor SEN13322 mide la humedad del suelo mediante resistencia variable. Tiene dos electrodos que, al ser insertados en el suelo, detectan cambios en la conductividad eléctrica. Cuando el suelo está húmedo, la resistencia entre los electrodos disminuye debido a la mayor conductividad del agua, resultando en una mayor señal analógica de salida.

Otra diferencia que se puede apreciar es con la precisión, mientras el sensor SEN13322 tiene una precisión de más o menos 5% el sensor capacitivo supera esa precisión obteniendo una de más o menos 2% lo que lo hace un mejor candidato para el proyecto. Además otra de las razones por la cual se propone usar el sensor capacitivo es porque cuenta con un recubrimiento especial el cual ayuda a proteger los electrodos y alargar su vida útil en ambientes húmedos (DFRobot, n.d.), en cambio el sensor SEN13322 que se propuso en el proyecto antes mencionado no cuenta con

este recubrimiento lo que hace que al momento de someterlo a un uso prolongado, la zona donde se encuentra en contacto con la humedad tenderá a corroerse y posteriormente al fallo del sensor y toma de datos errónea.

Este tipo de sensores resistivos, como el SEN13322, tienden a sufrir corrosión con el tiempo al estar expuestos directamente a la humedad del suelo, lo que afecta su durabilidad y precisión (MCI Electronics, n.d.; SparkFun, n.d.).



Ilustración 4 Sensor SEN0193

Uno de los sensores que no es utilizado en los proyectos mencionados es el MQ-135 (ilustración 5), este sensor permite detectar gases como amoníaco, alcohol, benceno y CO_2 , lo que lo hace útil para aplicaciones de monitoreo ambiental. Esto se confirma en su hoja técnica, donde se detalla su sensibilidad a estos gases (alldatasheet.com, n.d.). El motivo por el cual quiero implementar en el proyecto este sensor es para lograr una monitorización óptima para el crecimiento de los cultivos en un invernadero, ya que, al igual que se mencionó antes los niveles óptimos de CO_2 en un invernadero son esenciales para mejorar significativamente la tasa de crecimiento y producción dentro de él. Sin embargo, no tan solo se trata de

una buena producción, más bien se espera que cuando llegue el momento de reemplazar algún sensor o bien se tenga que intervenir por algún motivo dentro del invernadero los niveles de CO₂ no sean nocivos para la persona que se encuentre dentro. Según Ventilación (2023), los niveles recomendados de CO₂ en ambientes interiores deben situarse entre las 400 ppm y las 800 ppm (partes por millón), ya que este rango se considera seguro para la salud y adecuado para la ventilación en espacios cerrados.



Ilustración 5 Sensor MQ-135

Para el caso del sensor DS18B20 (ilustración 6), este, al ser un sensor de temperatura sumergible, puede ser utilizado principalmente en dos áreas: tomar la lectura de la temperatura del agua utilizada para el riego de los cultivos en el invernadero, o bien identificar la temperatura de la tierra de cultivo. Al igual que la temperatura ambiental, conocer los niveles de temperatura del suelo es fundamental para asegurar una buena producción y un desarrollo adecuado de las plantas. Según User (2016), la temperatura ideal varía dependiendo del tipo de cultivo y de la etapa de desarrollo en que se encuentre, pero en general, la mayoría de las semillas germinan entre los 18 °C y 25 °C. Asimismo, si la temperatura del suelo desciende por debajo de los 7 °C, las raíces crecen más lentamente y disminuye su capacidad de absorber agua y nutrientes.



Ilustración 6 Sensor DS18B20

En cuanto al prototipo inalámbrico se usará la placa NodeMCU ESP32 (ilustración 7) por las siguientes razones, como bien se menciona en los proyectos “Red De Sensores Inalámbricos Para El Monitoreo De Variables Agroecológicas En Cultivos Bajo Invernadero” y “Monitoreo y control de variables físicas en un invernadero a través del Internet de las Cosas”, estos usaron su antecesor que sería la NodeMCU ESP8266 la cual no se desmerece que sea una buena placa de trabajo, sin embargo, esta se encuentra limitada en temas de conectividad, ya que, la ESP8266 solo dispone de la conectividad por medio de wifi, en cambio la ESP32 dispone tanto de conectividad wifi como conectividad bluetooth lo que permite generar instancias donde el proyecto permita entregar opciones para posteriores mejoras.

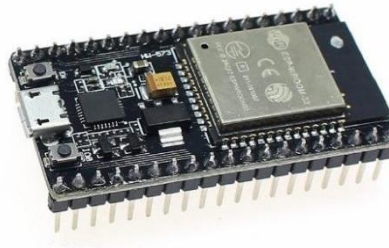


Ilustración 7 NodeMCU ESP32

La microcomputadora Raspberry Pi 2 (ilustración 8) es una excelente opción para implementar en proyectos que requieren la recopilación de datos de sensores y la visualización en tiempo real. Con su capacidad de procesamiento y su capacidad de conectividad, la Raspberry Pi 2 puede recibir datos de diferentes sensores y mostrarlos en una pantalla en tiempo real.

Además, puede realizar otras funciones importantes, como la creación de registros de los datos obtenidos. Esto significa que puedes almacenar los datos recopilados en una base de datos o en un archivo para su posterior análisis o referencia.

La elección de la Raspberry Pi 2 se basa en sus características y funcionalidades (Tabla 5). Aunque no requiere una gran capacidad de funcionamiento, la Raspberry Pi 2 ofrece suficiente potencia para manejar los datos de los sensores y realizar las tareas necesarias para visualizar y registrar los datos. Esta información concuerda con lo señalado por Manuti (2019), quien destaca la versatilidad y potencia de la Raspberry Pi 2 en aplicaciones de monitoreo.

Es importante tener en cuenta que existen numerosos recursos en línea, como tutoriales y comunidades de usuarios, que pueden proporcionar más información y orientación sobre cómo utilizar la Raspberry Pi 2 en diferentes proyectos.



Ilustración 8 Raspberry pi 2

Tabla 5 Características Raspberry pi 2

Procesador	Broadcom BCM2836 de 900 MHz ARM quad core Cortex-A7 de cuatro núcleos
GPU	VideoCore IV de doble núcleo con soporte de Open GL ES 2.0, hardware acelerado OpenVG hasta 1080p30 H.264
RAM	1GB SDRAM LPDDR2
Almacenamiento	tarjeta MicroSD para el Sistema Operativo
Video	• HDMI 1.3 y 1.4 tamaño estándar a 1080p con soporte CEC para control desde el mando del televisor • Salida de vídeo compuesto (PAL/NTSC) tipo jack de 3,5 mm 4 polos

Audio	• Audio digital por salida HDMI • Salida de audio estéreo compartida con la salida de vídeo compuesto
Red	Ethernet RJ45 10/100 BaseT
USB	4 x USB 2.0
Otros	• Conector MPI CSI-2 de 15 vías para cámara de vídeo HD Raspberry Pi • Conector de interfaz serie de display de 15 vías
GPIO	40 x pin conector macho
Sistemas Operativos Disponibles	• Snappy Ubuntu Core versión alfa • Raspbian
Alimentación	+5V a 2A a través de conector hembra microUSB
Tamaño	86 x 56 x 20 mm

Considerando que se estará trabajando todo el tiempo con humedad, la cual es el mayor enemigo para las piezas electrónicas, tal como se menciona en el proyecto "Red De Sensores Inalámbricos Para El Monitoreo De Variables Agroecológicas En Cultivos Bajo Invernadero", donde una de las conclusiones mencionaba que uno de los sensores utilizados se tendía a sulfatar con mucha rapidez, lo que hacía que comenzara a fallar y posteriormente dejará de funcionar. Por esta razón, se propone incorporar una nueva idea para solventar este problema, que es hacer uso de la impresión 3D.

La impresión 3D ofrece la posibilidad de crear contenedores herméticos que resguarden toda la parte electrónica sensible, evitando así la entrada de humedad que pueda dañar los componentes. Esta solución puede ser muy efectiva para proteger los sensores utilizados en el monitoreo de variables medioambientales en cultivos bajo invernadero.

Además, teniendo en cuenta que se tiene pensado crear un sistema inalámbrico junto con la idea de fabricar cajas con impresión 3D permitirá la creación de diferentes módulos de sensores. Esto significa que no será necesario utilizar cables que puedan pasarse a llevar y terminar dañando alguna pieza por un descuido.

En resumen, el uso de la impresión 3D para crear contenedores herméticos y módulos de sensores individuales puede ser una solución eficiente para proteger las piezas electrónicas de la humedad y evitar posibles daños. Esta innovadora técnica ofrece la oportunidad de mejorar la durabilidad y confiabilidad de los sensores utilizados en la monitorización de variables medioambientales en cultivos bajo invernadero.

4.3 Diagramas del Proyecto

4.3.1 Base de datos

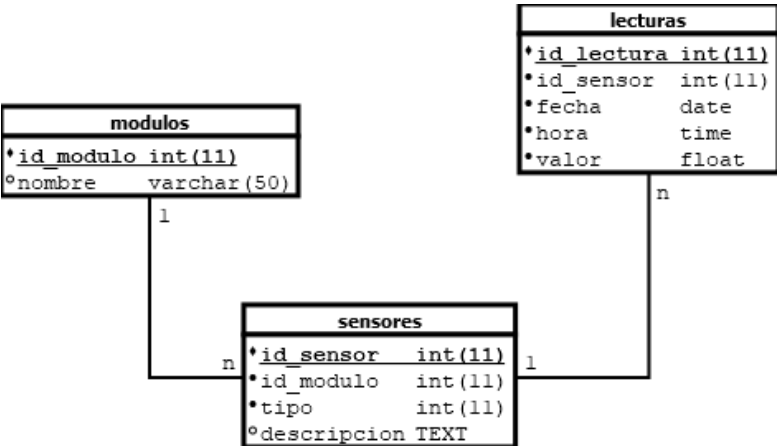


Ilustración 9 Diagrama Base de datos

Tabla 6 Detalle para tabla de lecturas

Modelo Lecturas – Cada dato obtenido de los sensores será almacenado según corresponda					
COLUMNA	OBSERVACIONES	RESTRICCIONES	TIPO	LÍM. CARÁCTER.	VALORES O FORMATOS
id_lectura	Identificador único para el ingreso de lecturas, autoincrementa	Clave Primaria	Int	11	N/A
id_sensor	Identificador del sensor asociado al registro previo	Clave Foránea (Sensores)	Int	11	N/A
fecha	Fecha actual obtenida desde el sistema	Obligatorio	Date	N/A	%d %m %Y Hora local de Chile (Ej: 18 01 2024)
hora	Hora actual obtenida desde el sistema	Obligatorio	Time	N/A	%H %M %S Hora local de Chile (Ej: 18 22 15)
valor	Lecturas enviadas desde módulo de sensores	Obligatorio	Float	2 decimales	Los datos se guardarán de forma estándar, interpretándose de forma distinta según el sensor

Tabla 7 Detalle para tabla sensores

Modelo Sensores – Lista de los sensores que se encuentran vinculados al sistema					
COLUMNA	OBSERVACIONES	RESTRICCIONES	TIPO	LÍM. CARÁCTER.	VALORES O FORMATOS

id_sensor	Identificador único para cada sensor del sistema	Clave Primaria	Int	11	N/A
id_modulo	Identificador del módulo asociado al registro previo	Clave Foránea (Módulos)	Int	11	N/A
tipo	Indicador que ayuda a identificar a que sensor corresponde	Obligatorio	Int	11	1 sen. Hum, 2 sen. Temp Amb, 3 Sen. Hum Amb, 4 sen. Temp, 5 sen.CO2
descripción	Descripción sensor ej. Nombre junto a unidad de medida	Opcional	Text	N/A	N/A

Tabla 8 Detalle para tabla módulos

Modelo Módulo – Lista de los sensores que se encuentran vinculados al sistema					
COLUMNA	OBSERVACIONES	RESTRICCIONES	TIPO	LÍM. CARÁCTER.	VALORES O FORMATOS
id_modulo	Identificador único para cada módulo del sistema	Clave Primaria	Int	11	N/A
nombre	nombre que recibirá cada módulo, ej. Modulo_prueba	Obligatorio	Varchar	50	N/A

4.3.2 Caso de uso

Un punto de vista por medio del diagrama de casos de uso que ayuda a identificar de qué manera interactúa el usuario con la página web.

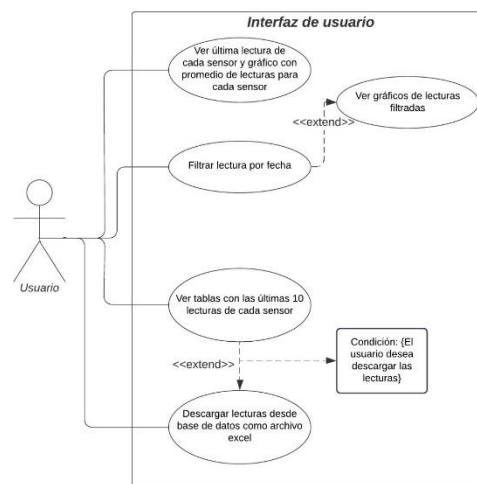


Ilustración 10 Diagrama caso de uso usuario general

4.3.3 Diagramas de conexión

En estos diagramas se muestra de qué forma deben ir conectados los componentes electrónicos para cada uno de los sensores involucrados en la obtención de datos.

- Diagramas prototipo inicial

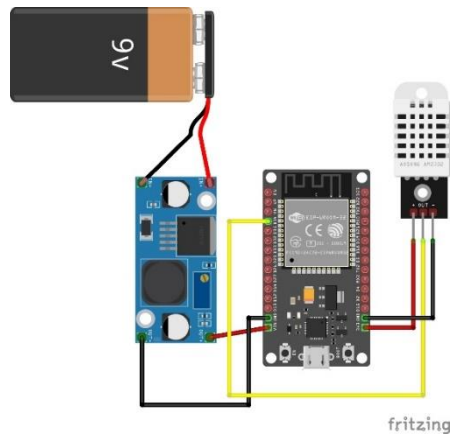


Ilustración 11 Conexión DHT22

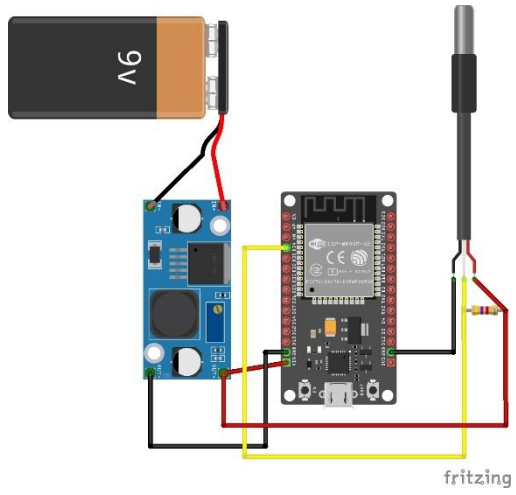


Ilustración 12 Conexión DS18B20

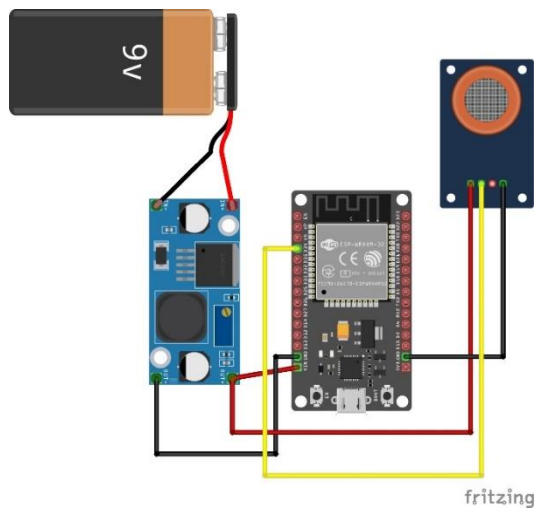


Ilustración 13 Conexión MQ-135

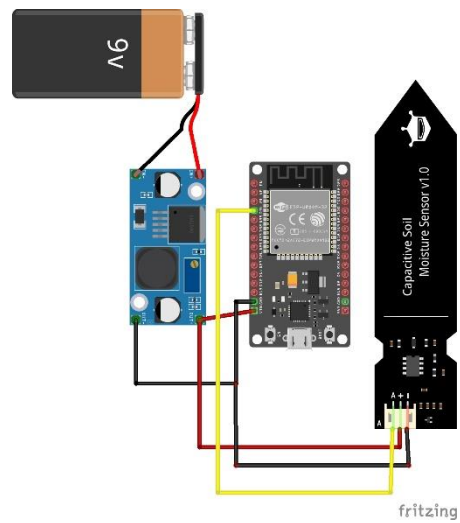


Ilustración 14 Conexión sensor capacitivo

- Diagrama módulo compacto

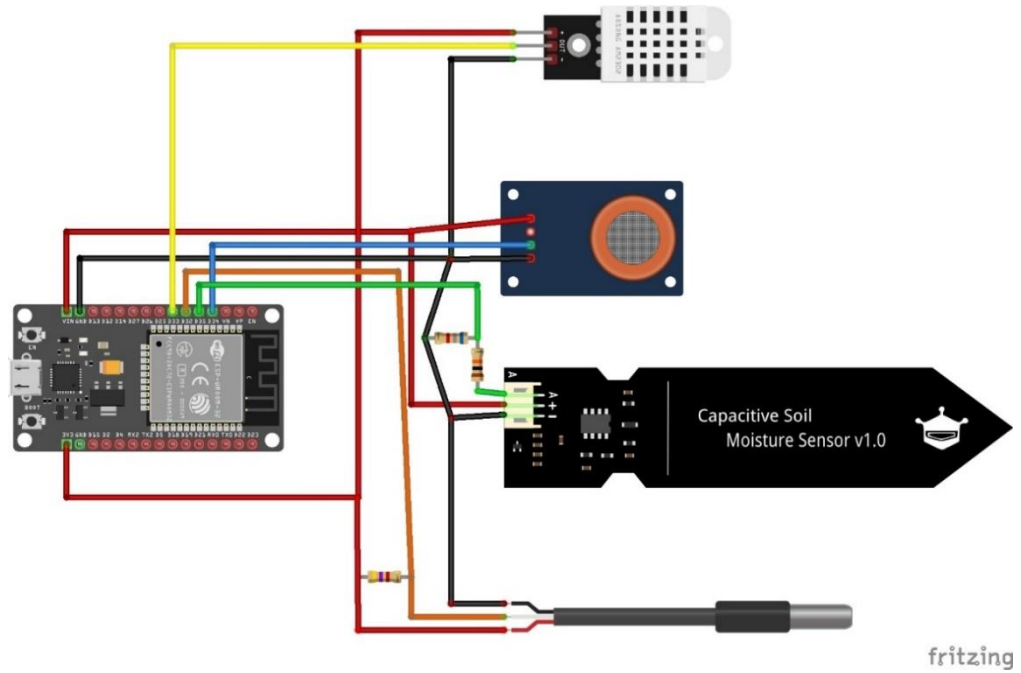


Ilustración 15 Conexión sistema inalámbrico módulo de cuatro sensores

CAPÍTULO V

Desarrollo

5.1 Tecnologías Implementadas en el Desarrollo

Tabla 9 Softwares principales en el desarrollo

Nombre	Descripción
Bootstrap 5.3	Biblioteca multiplataforma de código abierto para diseño de sitios y aplicaciones web.
Visual Studio Code	Es un editor de código fuente desarrollado por Microsoft, el desarrollo inicial de la página web se realizará mediante este editor.
MySQL	Sistema de gestión de base de datos.
PHPMyAdmin	Es una herramienta escrita en PHP teniendo como objetivo manejar la administración de MySQL
Arduino IDE	Herramienta de software que permite a los usuarios escribir, compilar y cargar código en placas de Arduino.

5.2 Creación Prototipo Inicial Sistema Inalámbrico

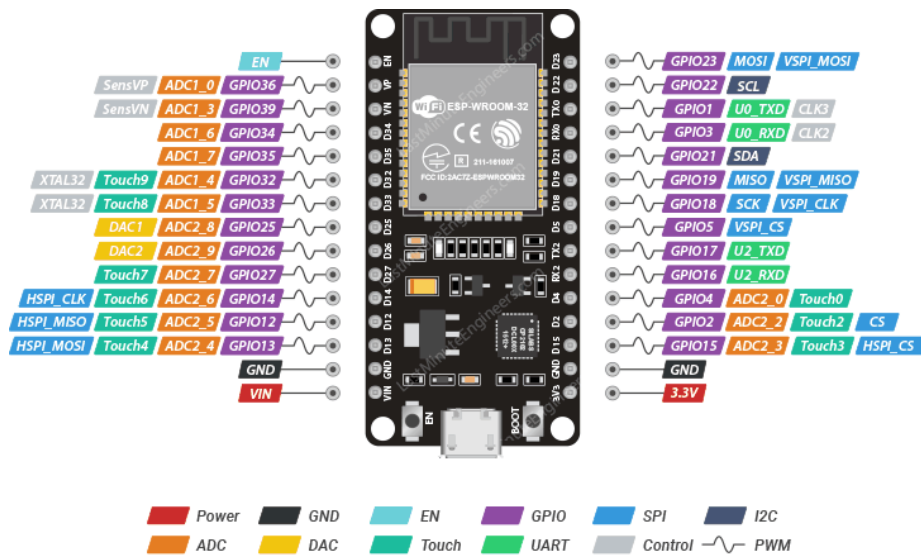
Para la creación del sistema inalámbrico se implementarán las placas ESP32 las cuales cuentan con conexión wifi y bluetooth, lo que nos permitirán enviar los datos de forma inalámbrica a nuestra Raspberry que será la encargada de almacenar los datos obtenidos. El sistema estará compuesto de diferentes módulos de los cuales se componen de una placa ESP32 junto a los sensores con los que trabajará, además de su propia fuente de alimentación.

5.2.1 Armado de hardware prototipo inicial

Al preparar el ensamblaje de los componentes electrónicos, es esencial tener especial cuidado al trabajar con la placa ESP32, ya que esta cuenta con diversas modalidades de operación que afectan el uso y configuración de sus pines. La funcionalidad de los pines varía según el modo seleccionado, lo que hace indispensable consultar el diagrama de pines (Pinout) específico de la placa, así como cualquier información adicional relevante para su correcto funcionamiento.

Es importante destacar que algunas versiones de la ESP32 tienen pines reservados para funciones específicas que no pueden ser utilizados para otros propósitos. Además, en el mercado existen múltiples modelos y marcas de placas ESP32, cada una con su propio diseño y diagrama de pines. Por esta razón, es recomendable identificar el modelo exacto de la placa que se está utilizando y verificar cuidadosamente su documentación técnica antes de proceder con el montaje. Esto garantizará un ensamblaje seguro y funcional, evitando errores o daños en los componentes. También es fundamental considerar la protección contra descargas electrostáticas (ESD) durante el ensamblaje, ya que estas pueden dañar los componentes sensibles del microcontrolador (Texas Instruments, 2019).

Para este proyecto en concreto se usará la popular placa de desarrollo ESP32 devkit v1 de 30 pines, cuyo diagrama de pines fue consultado en la documentación proporcionada por Last Minute Engineers (LME Editorial Staff, 2023).



ESP32 Dev. Board Pinout

Ilustración 16 Pinout placa ESP32 devkit v1

En concreto esta placa originalmente solo opera con valores digitales, pero cuenta con una unidad que puede convertir señales analógicas a digitales, según el Pinout de la placa estas se encuentran con las siglas ADC que están divididas en dos grupos, los ADC1 que sería conformados por los pines GPIO36, GPIO39, GPIO34, GPIO35, GPIO32 y GPIO33 de los cuales de acuerdo a la información proporcionada por la página los cuatro primeros pines solo se podrían configurar para la entrada de datos y no para alguna acción de salida. El segundo grupo ADC2 está conformado por los pines GPIO25, GPIO26, GPIO27, GPIO14, GPIO12, GPIO13, GPIO4, GPIO2 y GPIO15, aunque lamentablemente este segundo grupo no se podrá utilizar debido a que se va a hacer uso de la conexión wifi y estos pueden ser inestables cuando se usan con el módulo

wifi activo, ya que comparten recursos con la comunicación inalámbrica de la placa. Por otra parte, el ADC del ESP32 también presenta ciertas limitaciones técnicas y puede requerir calibración adicional para obtener lecturas precisas, especialmente en entornos ruidosos o con señales analógicas fluctuantes (Teja, 2024).

Ya teniendo esta información clara se puede dar comienzo al armado del prototipo inicial de este sistema. Se tiene pensado implementar cuatro módulos los cuales corresponderían a los cuatro sensores que se utilizarán, por lo que cada uno de ellos tendrá su propia placa ESP32 y su fuente de alimentación para lograr una mejor maniobrabilidad del hardware.

Como se puede apreciar en la sección de diagramas de conexión específicamente las que son del prototipo inicial podemos ver que las conexiones son bastantes sencillas el sensor que puede ser alimentado por medio de 3.3V podemos conectarlo directamente a la placa mientras que los demás que necesitan un suministro de 5V irán conectados directamente al mismo sistema que alimentará la placa, el cual está compuesto por un Stepdown y una batería de 9V.

El Stepdown es un reductor de voltaje el cual es necesario configurar antes de usarlo para ello es necesario conectar la batería de 9V, ya que esta suministra la energía requerida para el proceso. Posteriormente, se debe ajustar manualmente para obtener los 5V necesarios para alimentar todo el sistema. Para ello, se utiliza un multímetro que permite monitorear el voltaje de salida. Con la ayuda de un destornillador plano, se localiza un pequeño bloque azul que corresponde a un potenciómetro, encargado de regular el voltaje de salida. Girando el potenciómetro en sentido horario, se incrementa el voltaje, mientras que, al girarlo en sentido antihorario, se reduce. Este proceso debe realizarse

hasta que el multímetro indique el voltaje deseado, como se muestra en la siguiente imagen.

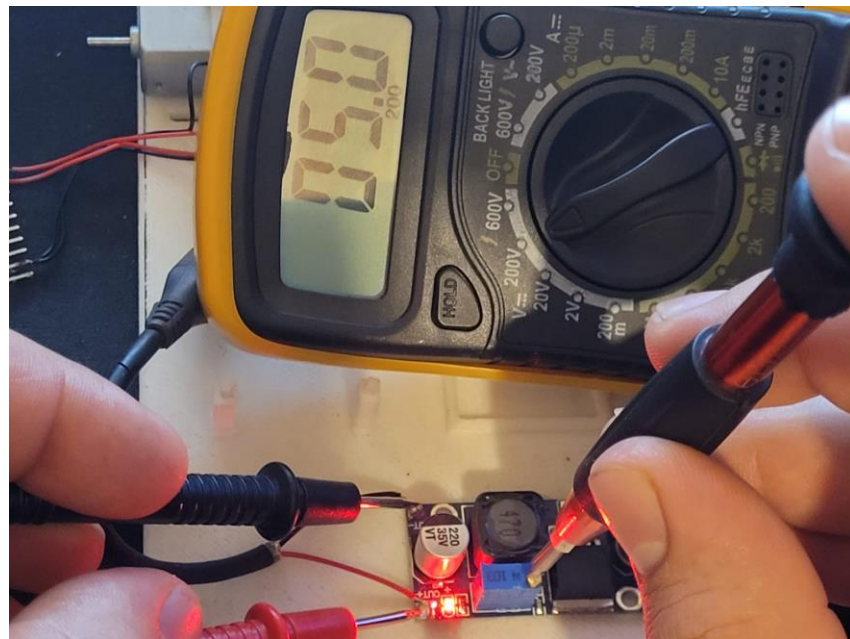


Ilustración 17 Configuración Stepdown

5.2.2 Creación códigos primer prototipo

Para comenzar con la creación de los códigos que irán dentro de las placas ESP32 primeros debemos configurar nuestro entorno de desarrollo que es el Arduino IDE, preparar el entorno de Arduino IDE para trabajar con la ESP32 es necesario porque, de forma predeterminada, Arduino IDE no incluye soporte para esta placa. Para programar la ESP32, se deben instalar los controladores y las herramientas necesarias para compilar y cargar el código. Esto permite que el IDE reconozca la placa, configure los pines correctamente y aproveche las capacidades específicas de la ESP32, como el Wi-Fi y Bluetooth. Sin este paso, no

sería posible desarrollar ni cargar programas en la ESP32 desde Arduino IDE.

Para poder instalar el paquete de soporte necesario primero se debe abrir Arduino IDE, luego en la esquina superior izquierda en la opción de archivo presionamos y se desplegarán más opciones de las cuales nos iremos a la que dice preferencias, una vez seleccionada esa opción aparecerá un cuadro como el que se muestra en la siguiente imagen.

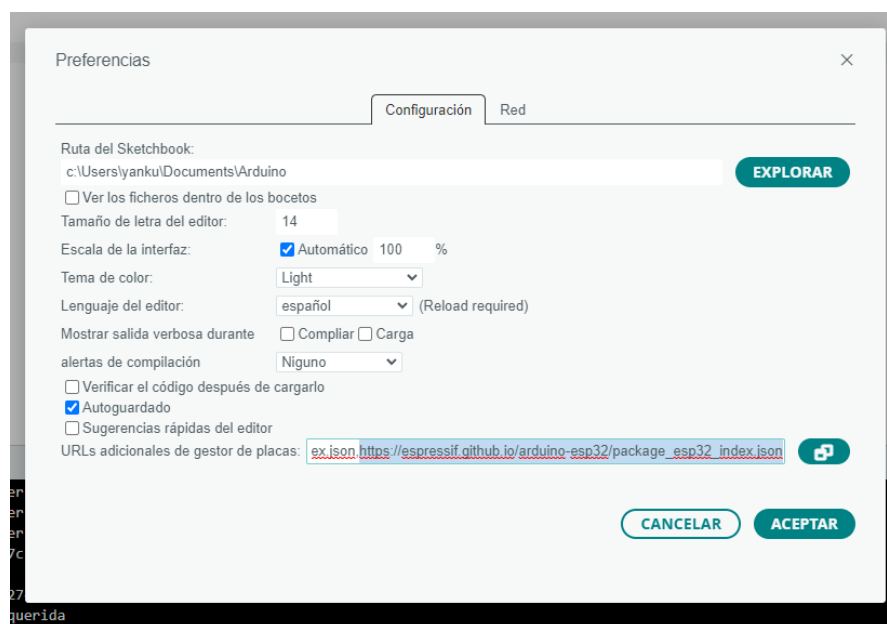


Ilustración 18 Cuadro de preferencias para la instalación de package ESP32

En este cuadro nos tenemos que dirigir donde dice “URLs adicionales de gestor de placas”, en esta zona se deberá colocar la siguiente URL.

https://espressif.github.io/arduino-esp32/package_esp32_index.json

Ilustración 19 URL package ESP32

Esta URL apunta a un archivo JSON alojado por Espressif, el fabricante de la ESP32. Este archivo es un índice que contiene la

información necesaria para que Arduino IDE pueda descargar e instalar los controladores y las herramientas específicas para programar placas basadas en ESP32. Para poder instalar el paquete solo se debe de colocar la URL en el cuadro de texto como se muestra en la ilustración 19 y en caso de tener una o más URLs dentro del mismo cuadro de texto para incorporar una nueva solo hace falta separarlas por medio de una coma, una vez terminado solo se presiona aceptar y se instalarán automáticamente los paquetes de soporte de la ESP32.

Una vez ya obtenidos los paquetes para la ESP32 ahora hay que instalar las librerías para poder utilizar los sensores y las funciones de WiFi de la placa, para esto nos dirigimos al gestor de bibliotecas de Arduino IDE que se puede encontrar en las opciones al costado izquierdo o bien en la opción de herramientas donde se desplegaran varias opciones y elegimos la de gestor de bibliotecas.

En este gestor tendremos un buscador que nos permitirá filtrar nuestra búsqueda para poder encontrar más rápido las librerías necesarias de las cuales necesitaremos para el sensor DHT22, sensor DS18B20 que junto a la librería que se debe instalar también requiere de una dependencia llamada OneWire que el mismo gestor sugerirá su instalación en caso de no tenerla, en cuanto a las librerías de la ESP32 necesitaremos las de HTTPClient y WiFi, a continuación se encontraran imágenes de las librerías que se debiesen instalar ya que al buscar salen varias opciones pero no todas son compatibles.



Ilustración 20 Librerías sensor DHT22

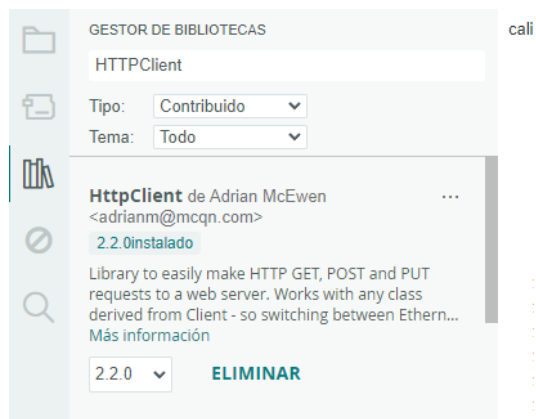


Ilustración 21 Librería función HTTPClient ESP32

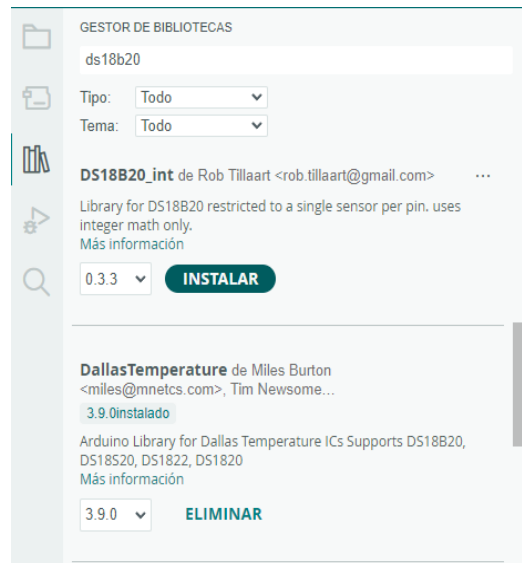


Ilustración 22 Librería sensor DS18B20



Ilustración 23 Librería función WiFi ESP32

Preparando los códigos para los sensores hay aspectos o ciertas zonas del código que son compartidas por cada uno de ellos, por lo que no es necesario explicarlas tan a detalle por que como se muestra en la ilustración principalmente lo que comparten entre ellos son las librerías que nos ayudaran a conectarnos a una red wifi y poder enviar los datos a través de internet, cambiando netamente las variables de los datos del sensor que en ese caso cada sensor tiene sus propios datos.

```
Connection_Sen_Humedad.ino
1  #include <WiFi.h>
2  #include <HttpClient.h>
3
4  const char* ssid = "";
5  const char* password = "";
6  const char* url = "http://192.168.1.8/Prueba_Tesis/ingresoValores.php";
7  const int sensorPin = 34;
8  const int idSensor = 1; //Correspondiente al número otorgado al registrar en la Base de datos
9  const int tipoSensor = 1; //1 = Sen. Humedad Suelo | 2 = Sen. Temperatura y Humedad Ambiental | 3 = Sen. Temperatura | 4 = Sen. CO2
```

Ilustración 24 Vista valores principales código sensores

Para entender un poco lo que sucede en esa parte del código se están definiendo las variables ssid, password y url que serán de tipo char* y las variables sensorPin, idSensor y tipoSensor que van a ser de tipo int. En la variable ssid ira el nombre de la red wifi a la cual queramos conectar la placa y en la variable password ira la contraseña de la red wifi, estas variables junto a la de url que esta almacenará la dirección del archivo que nos va a ayudar a ingresar los datos a la base de datos, son las únicas que no cambiarán su contenido en cada uno de los sensores, integrando a este grupo la variable de sensorPin que para este caso se quedará fijo en el pin 34 de la placa.

Mientras que para las otras variables dependerá de que sensor se esté trabajando, tipoSensor puede tomar valores del 1 al 4 siendo sensor de humedad suelo, sensor de temperatura y humedad, sensor de temperatura suelo y sensor de concentración de dióxido de carbono correspondientemente, para la variable idSensor el valor de este dependerá del que otorgue la base de datos al momento de registrar el sensor.

```
14 void setup() {  
15     Serial.begin(115200);  
16     WiFi.begin(ssid, password);  
17  
18     while (WiFi.status() != WL_CONNECTED) {  
19         delay(1000);  
20         Serial.println("Conectando a la red WiFi...");  
21     }  
22     Serial.println("conectado a la red WiFi");  
23 }
```

Ilustración 25 Vista configuración función void setup()

Más abajo en el código tenemos la función void setup() que en esta parte se hacen todas las configuraciones que solo se ejecutarán una vez en todo el código, para este caso se iniciará la comunicación serial de la placa con el computador y se configurará la velocidad de comunicación entre ellos a 115200 Baudios, luego implementaremos la función WiFi.begin() que nos permitirá conectar la placa a una red wifi pero para ello debemos entregarle antes el nombre y la contraseña de la red que queremos conectar de los cuales ya fueron definidas más arriba en las variables ssid y password, para luego hacer una pequeña consulta con tal de saber si la placa pudo conectarse o no a la red wifi.

Hasta este punto dependiendo del sensor que se trabaje hay ciertas cosas que se tiene que incorporar como es el caso del código para el

sensor de humedad de suelo que se incorporan dos variables que serían las de la siguiente ilustración.

```
const int valorSinHumedad = ;  
const int valorConHumedad = ;
```

Ilustración 26 Variables para calibración sensor capacitivo

En las cuales se deben anotar dos valores que se dan al calibrar el sensor, uno de ellos se obtiene mientras el sensor está seco y debe ser el valor más alto que pueda alcanzar y mientras el sensor está en contacto con el agua se debe identificar el valor más bajo que pueda alcanzar. Al trabajar con el sensor de temperatura de suelo primero debemos incorporar dos librerías nuevas que son necesarias para poder trabajar este sensor y son las siguientes.

```
#include <OneWire.h>  
#include <DallasTemperature.h>
```

Ilustración 27 Librerías sensor DS18B20

Ya luego de incorporar esas nuevas librerías se tiene que instanciar las clases OneWire y DallasTemperature, por lo que se debe colocar las siguientes líneas de código (ilustración 28) antes de la función void setup() y dentro de la función tenemos que integrar el siguiente código “sensorDS18B20.begin();”.

```
//Instanciar clases OneWire y DallasTemperature  
OneWire oneWireObjeto(sensorPin);  
DallasTemperature sensorDS18B20(&oneWireObjeto);
```

Ilustración 28 Instanciar clases OneWire y DallasTemperature

Para trabajar el sensor de humedad y temperatura ambiental debemos incorporar la librería DHTesp.h la cual también podemos renombrar de la siguiente manera “DHTesp dht;” esto se hace para que

cada vez que necesitemos acudir a la librería no sea necesario colocar el nombre completo, luego dentro de la función void setup() tenemos que incorporar lo siguiente “dht.setup(sensorPin, DHTesp::DHT22);” que básicamente nos permitirá configurar por el pin que recibiremos la información y cuál va a ser el sensor que usaremos específicamente.

Pasando a la última parte del código tenemos la función void loop() donde debemos colocar el código necesario para obtener los datos ya que este se estará ejecutando repetidamente en esta función la única zona que quedará igual en todos los sensores es la siguiente.

```
29  if (WiFi.status() == WL_CONNECTED){
30      HTTPClient http;
31
32      http.begin(url);
33      http.addHeader("Content-Type", "application/x-www-form-urlencoded");
34
35      http.POST("id_sensor="+String(idSensor)+"&tipo_sensor="+String(tipoSensor)+"&hum_suelo="+String(hum_suelo));
36
37      http.end();
38  }
39  delay(300000);
40 }
```

Ilustración 29 Código para envío de datos por método POST

La razón por la que se queda igual para todos los sensores, es porque comenzamos la comunicación entre la ESP32 y la página encargada de insertar los datos ya que al momento de implementar la línea 32 de la ilustración estamos entregándole la dirección en donde se encuentra nuestro archivo para luego con la línea 33 realizar una solicitud HTTP indicando que los datos enviados en el cuerpo de la solicitud están codificados como application/x-www-form-urlencoded, en palabras más simples, los datos serán enviados en un formato clave-valor, con cada par separado por un signo igual (=) y los pares separados por un signo de ampersand (&). Por ejemplo, name=John&age=30. Esta cabecera le dice al servidor cómo interpretar los datos recibidos y por último específicamente línea 35 enviamos una solicitud tipo POST al servidor entregándole los datos que necesitamos sean enviados que en este caso

serán el idSensor, tipoSensor y los datos obtenidos por cada uno de los sensores.

Para poder completar el código nos falta solo una parte que es la obtención de los datos, esta obtención se trabaja de forma distinta para cada sensor comenzando con el sensor capacitivo de humedad de suelo.

```
void loop() {  
  int sensorValue = analogRead(sensorPin);  
  int hum_suelo = map(sensorValue, valorSinHumedad, valorConHumedad, 0, 100);  
}
```

Ilustración 30 código para obtener lecturas sensor capacitivo

Como es un sensor analógico usaremos la función `analogRead()` entregándole el pin por el cual se va a obtener los datos, luego usaremos la función `map()` que nos ayudará a interpretar los valores arrojados en un rango de 0 a 100, para ello le debemos entregar primero el valor que recoge el sensor luego colocar los valores ya predefinidos anteriormente del sensor seco y el otro de contacto con el agua.

Para el sensor MQ-135 se hace un proceso similar solo que para él solo se implementa la función de `analogRead()` y nada más ya que no hace falta hacer un reajuste de los datos proporcionados.

En el sensor de temperatura de suelo se simplifican más las cosas ya que solo se implementa “`sensorDS18B20.requestTemperature()`” para poder obtener los datos del sensor, sin embargo, si queremos imprimir los valores en el puerto serial o enviar como es en este caso debemos escribir lo siguiente “`sensorDS18B20.getTempCByIndex(0)`”, el cero que se implementa en esa función es para indicar que estamos solicitando las lecturas del primer sensor en la lista, ya que, estos tipos de sensores pueden estar varios trabajando simultáneamente y se pueden diferenciar de esa manera.

En cuanto al sensor DHT22 como este recopila dos tipos de información que son la temperatura y humedad ambiental el siguiente código `TempAndHumidity sensorValue = dht.getTempAndHumidity()` si desglosamos un poco `dht.getTempAndHumidity()` es una función que llama al sensor DHT22 para leer tanto la temperatura como la humedad actual, mientras que el retorno de los datos es de tipo `TempAndHumidity` y es por eso que a la variable se le otorga ese tipo de dato, una vez teniendo este la forma que se tiene para acceder a los datos almacenados en la variable se debe hacer de esta manera `sensorValue.temperature` y `sensorValue.humidity`, con estos valores podemos configurar la cantidad de decimales que queremos obtener y solo hace falta colocar una coma y el número correspondiente a la cantidad de decimales que queremos por ejemplo `sensorValue.temperature,2`.

Al momento de hacer unas pruebas para ver cómo estaban trabajando los datos de los sensores me pude percatar que al momento de trabajar con el sensor capacitivo los datos que arrojaba por el monitor serial eran extremadamente altos llegando a 4095 que era el valor más alto que se podía alcanzar en los 12 bits que proporciona la ESP32. En cuanto a los demás sensores no se encontraron problemas al momento de tomar las lecturas.

5.3 Primera Iteración Y Retroalimentación

En la primera versión del prototipo se identificaron diversos aspectos a mejorar que servirían de base para el desarrollo de la siguiente iteración. Entre los principales puntos señalados estuvieron: la necesidad de centralizar todos los sensores en un solo módulo para mejorar el mantenimiento y reducir la redundancia de hardware; incluir un interruptor de encendido/apagado que permitiera controlar manualmente la toma de datos; disminuir el intervalo de recolección de datos de 30 minutos a 5 minutos para capturar cambios ambientales con mayor precisión; agregar fecha y hora a cada lectura para

mejorar la trazabilidad; revisar y corregir las inconsistencias detectadas en las lecturas del sensor capacitivo de humedad; y evaluar el rendimiento energético del sistema al alimentarlo desde una sola fuente para todo el módulo. Estos puntos sirvieron como lineamientos para orientar las mejoras implementadas en la segunda versión del sistema.

En la retroalimentación recibida sobre el primer prototipo del proyecto, se analizaron diversos aspectos que requieren ajustes y mejoras para optimizar el sistema. El diseño inicial, que utilizaba cuatro sensores (capacitivo, DHT22, DS18B20 y MQ-135), cada uno conectado a su propia ESP32 y funcionando de manera unitaria mediante Wi-Fi, cumplía con la función básica de captura de datos. Sin embargo, se señaló que esta implementación pierde practicidad debido a que cada sensor operaba de forma independiente, lo que incrementa la complejidad en la configuración y el mantenimiento en escenarios reales. Como solución, se recomendó integrar todos los sensores en un único módulo compacto, centralizando la toma de lecturas para facilitar su manejo y reducir redundancias en el hardware.

Además, se identificaron problemas relacionados con la alimentación del sistema. Cada sensor junto con su ESP32 contaba con su propia fuente de poder, una batería de 9V. Uno de los principales problemas observados fue que, al agotarse la energía de la batería, esta debía ser reemplazada, y al reconectar el suministro de energía, comenzaba a tomar datos erróneos porque no se encontraba en la zona de lectura adecuada en ese momento. Para evitar que estos valores incorrectos se registraran en la base de datos, se recomendó implementar un switch que permitiera desconectar y reconectar manualmente el suministro eléctrico, asegurando que las lecturas fueran precisas desde el inicio. Adicionalmente, se sugirió realizar pruebas para evaluar la duración de la batería de 9V, considerando que en la próxima

iteración ya no estará energizando únicamente un sensor, sino todo un módulo integrado, lo que podría impactar significativamente en su autonomía.

Otro aspecto crítico señalado fue el tiempo de recolección de datos, establecido inicialmente en 30 minutos, lo cual fue considerado excesivo para un sistema que necesita responder a cambios rápidos en las condiciones ambientales. Por ejemplo, un aumento repentino en la temperatura podría pasar desapercibido dentro de ese intervalo, retrasando acciones pertinentes. Se recomendó reducir este tiempo de recolección a 5 minutos, debido a que es un periodo más corto, permite una mayor precisión en el monitoreo y una reacción oportuna.

En cuanto al manejo de los datos recolectados. Se indicó que cada registro debería incluir tanto la fecha como la hora exacta de toma de los datos, mejorando la trazabilidad y facilitando el análisis histórico. Por último, se abordó un problema técnico relacionado con el sensor capacitivo, el cual presentó fallas en sus lecturas, luego de tener 2 días seguidos probando el funcionamiento de los sensores este sensor presentó lecturas elevadas y que se mantenían en el tiempo por lo que se recomendó realizar una revisión detallada de este componente para identificar y corregir las inconsistencias, asegurando que los datos generados por el sistema sean confiables. Estas observaciones marcan un camino claro para las siguientes iteraciones del desarrollo, enfocándose en mejorar la integración, la eficiencia energética y la funcionalidad general del sistema.

5.4 Mejoras Del Prototipo

5.4.1 Hardware

Tomando en consideración lo hablado en la presentación del primer prototipo, comenzaré solucionando el dilema de los sensores específicamente el armado del hardware, lo que se recomendó fue implementar un solo módulo con los cuatro sensores juntos por lo cual se

creó otro diagrama de conexión (ilustración 15) en el cual nos podemos dar cuenta que ahora utilizaremos más los recursos de la placa ESP32 porque para esta ocasión la placa se energizará directamente por su puerto microUSB y así poder utilizar cada uno de sus pines a disposición, como podemos darnos cuenta, ahora los sensores que necesitan de 5V para poder funcionar van directamente conectados al pin VIN de la placa el cual nos permitía tanto alimentar la placa como otorgar 5V en caso de conectarse por su puerto de alimentación original y para los sensores que necesitan 3.3V seguirán conectados al pin correspondiente de la placa. Siguiendo los cambios propuestos para el hardware se implementó un switch para poder cortar o dar el suministro energético de forma manual. Por el problema en las lecturas del sensor capacitivo, resulta que era a causa de que los pines de la placa ESP32 trabajan a un voltaje de 0 a 3.3V y este sensor por medio de su cable de datos entrega de 0 a 5V por lo que al momento de tomar las lecturas la placa tenía una saturación de valores, entonces para solucionar este problema podemos integrar un divisor de voltaje que nos ayude a interpretar los datos entregados por el sensor al rango seguro de la placa que es 0 a 3.3V, esto se logra implementando dos resistencias una de 10KOhms y otra de 6.8KOhms las cuales se deben conectar de la siguiente manera, uno de los extremos de la resistencia de 10KOhms, se debe conectar con el cable de datos proveniente del sensor y el otro extremo va conectada a uno de los extremos de la otra resistencia de 6.8KOhms. el nodo que se genera al conectar estas dos resistencias debe de ir conectado al pin 35 de la placa, mientras que el extremo faltante de la segunda resistencia se debe conectar a la línea de GND que está usando el sensor

5.4.2 Creación nuevo código

Ya que los sensores se juntaron todos dentro de una misma placa toca crear un nuevo código para poder controlar los sensores, sin embargo, como ya disponíamos de los otros códigos solo faltaba juntarlos y hacer que entre ellos funcionasen. Con esto quiero decir que las funciones que se implementaron anteriormente no sufrieron grandes cambios más bien se reacomodaron.

```
1  #include <DHTesp.h>
2  #include <WiFi.h>
3  #include <HTTPClient.h>
4  #include <OneWire.h>
5  #include <DallasTemperature.h>
6  #include <esp_sleep.h>
7  #define factor_convertor 1000000 // Conversión de segundos a microsegundos
8  #define time_to_sleep 300 // Tiempo de Deep Sleep en segundos (5 minutos)
9
10 const char* ssid = "";
11 const char* password = "";
12
13 const char* url = "http://192.168.1.19/ingreso.php";
14
15 //1 = Sen. Humedad Suelo | 2 = Sen. DHT22 Humedad Ambiental | 3 = Sen. DHT22 Temperatura Ambiental | 4 = Sen. Temperatura | 5 = Sen. CO2
16 const int idSensor1 = 1;
17 const int idSensor2 = 2;
18 const int idSensor3 = 3;
19 const int idSensor4 = 4;
20 const int idSensor5 = 5;
21
22 const int sensorPin1 = 33;
23 const int sensorPin2 = 35;
24 const int sensorPin3 = 34;
25 const int sensorPin4 = 32;
26
27 const int valorSinHumedad = 426;
28 const int valorConHumedad = 187;
29
30
31 DHTesp dht;
32
33 //Instanciar clases OneWire y DallasTemperature
34 OneWire oneWireObjeto(sensorPin4);
35 DallasTemperature sensorDS18B20(oneWireObjeto);
36
```

Ilustración 31 Mejora de código para trabajar 4 sensores juntos

Como se logra ver en la ilustración 31 en este nuevo código implementaremos todas las librerías que se necesitaban para cada sensor incorporando una nueva que es `esp_sleep.h` la cual nos va a habilitar una funcionalidad en nuestra placa más adelante, también se incorporan la línea 7 y 8 las cuales nos ayudan a hacer una conversión de segundos a microsegundos que se usaran para poder configurar la nueva funcionalidad de la placa. Más abajo tanto la variable que almacena el nombre y la variable que almacena la contraseña de la red wifi que

queremos conectar quedaran igual a los códigos anteriores, mientras que la url esta vez cambiará porque ya comenzaremos a implementar el uso de la Raspberry pi.

Dado que estaremos trabajando con cuatro sensores juntos se colocó un id para cada caso, con una excepción que en realidad serán cinco ids distintas las cuales fueron otorgadas por la base de datos al momento de registrar los sensores en la tabla sensores que se puede visualizar en la ilustración 9, el hecho de porque son cinco y no cuatro es porque el sensor DHT22 tiene dos lecturas en una y como en la base de datos se hará un solo registro por cada lectura para poder acceder de mejor manera a los datos históricos se tomó la decisión de separar estas

lectura. Ahora para fines de utilizar todos los sensores se configuran otros tres pines que en concreto se estarán usando los pines 32, 33, 34 y 35.

Ilustración 32 Implementación toma de lecturas para todos los sensores

En la función `setup()` repetiremos lo mismo que en los códigos

```
37 void setup() {
38     Serial.begin(115200);
39     WiFi.begin(ssid, password);
40     dht.setup(sensorPin1, DHTesp::DHT22);
41     sensorDS18B20.begin();
42     esp_sleep_enable_timer_wakeup(time_to_sleep * factor_conversor);
43
44     while (WiFi.status() != WL_CONNECTED) {
45         delay(1000);
46         Serial.println("Conectando a la red WiFi...");
47     }
48     Serial.println("conectado a la red WiFi");
49
50     delay(5000);
51     //toma de muestra y envio a raspberry DHT22
52     TempAndHumidity sensorValueDht = dht.getTempAndHumidity();
53     envioDatos(String(idSensor2),String(sensorValueDht.humidity,0));
54     delay(500);
55     envioDatos(String(idSensor3),String(sensorValueDht.temperature,2));
56
57     //toma de muestra y envio a raspberry capacitivo
58     int sensorValueCap = analogRead(sensorPin2);
59     int hum_suelo = map(sensorValueCap, valorSinHumedad, valorConHumedad, 0, 100);
60     envioDatos(String(idSensor1),String(hum_suelo));
61
62     //toma de muestra y envio a raspberry DS18B20
63     sensorDS18B20.requestTemperatures();
64     envioDatos(String(idSensor4),String(sensorDS18B20.getTempCByIndex(0)));
65
66     //toma de muestra y envio a raspberry MQ-135
67     int sensorValueCO2 = analogRead(sensorPin3);
68     envioDatos(String(idSensor5),String(sensorValueCO2));
69
70     //Se inicia modo deep sleep
71     delay(3000);
72     esp_deep_sleep_start();
73 }
74
```

anteriores incorporando esta vez la nueva modalidad de la placa que será un modo de sueño profundo el cual nos permite que la placa entre en un estado de suspensión donde consume menos energía en un tiempo determinado que se debe configurar previamente y luego se reinicia automáticamente. Cómo se puede apreciar en la ilustración 32 en la línea 42 se está configurando el tiempo que estará en este modo de sueño

profundo el cual será de 5 minutos como fue propuesto en la retroalimentación. Más abajo se puede apreciar que implementamos las lecturas de los sensores uno por uno y luego envían esos datos a una función llamada `envioDatos()` en forma de cadena, lo que hace esta nueva función es enviar los datos por el método POST como se hacía anteriormente solo que esta vez para no tener una redundancia de código se implementa esta función aparte, solo entregándole los datos que se necesitan enviar, en la siguiente ilustración se muestra cómo quedaría estructurada la nueva función.

Ilustración 33 Nueva estructuración para enviar datos por método POST

```
void envioDatos(const String& idSensor,const String& valor){  
    if (WiFi.status() == WL_CONNECTED){  
        HTTPClient http;  
  
        http.begin(url);  
        http.addHeader("Content-Type", "application/x-www-form-urlencoded");  
  
        String postData = "id_sensor="+ idSensor + "&valor="+ valor;  
  
        http.POST(postData);  
  
        http.end();  
    }  
}
```

5.4.3 Código recolector de datos

Luego de terminar las mejoras y cambios solicitados entorno a los sensores procedo a la creación del código que recibirá los datos de la placa y los insertará, este está hecho en php utilizando consultas sql para hacer la inserción.

```
1  <?php
2  // Configuración de La base de datos
3  $servername = "localhost";
4  $username = "root";
5  $password = "123456";
6  $dbname = "invernadero";
7
8  date_default_timezone_set('America/Santiago');
9
10 $id_sensor      = $_POST['id_sensor'];
11 $valor          = $_POST['valor'];
12 $fecha_actual   = date("d-m-Y");
13 $hora_actual    = date("H:i:s");
14
15
16 // Crear conexión
17 $conn = new mysqli($servername, $username, $password, $dbname);
18
19 // Verificar conexión
20 if ($conn->connect_error) {
21     die("Conexión fallida: " . $conn->connect_error);
22 }
23
24 $sql = "INSERT INTO lecturas (id_sensor, fecha, hora, valor) VALUES
25 ('$id_sensor', '$fecha_actual', '$hora_actual', '$valor')";
26
27 if($conn->query($sql) == true){
28     echo "Registro de datos Exitoso";
29 }else{
30     die("Fallo en ingreso del registro" . $conn->error);
31 }
32
33 mysqli_close($conn);
34
35
36 ?>
```

Ilustración 34 Código receptor y de inserción de datos

Como también se puede apreciar en la retroalimentación fue solicitado incorporar la hora y la fecha en la cual se tomaba la lectura y en el código fue incorporado junto a los otros datos provenientes de la placa, para la obtención de estos datos se implementó la función que se encuentra en la línea 8 del código, lo que hace esta función es establecer la zona horaria predeterminada a "America/Santiago". Esto asegura que

todas las funciones relacionadas con la fecha y la hora utilicen la zona horaria de Santiago, Chile.

5.4.4 Configuración Raspberry pi

Primero partiremos por instalar el sistema operativo de Raspberry el cual lo podremos encontrar en su página oficial www.raspberrypi.com en el apartado software, luego al estar ya en esa página bajaremos hasta donde diga “Manually install an operating system image” y presionaremos el botón que esta debajo.

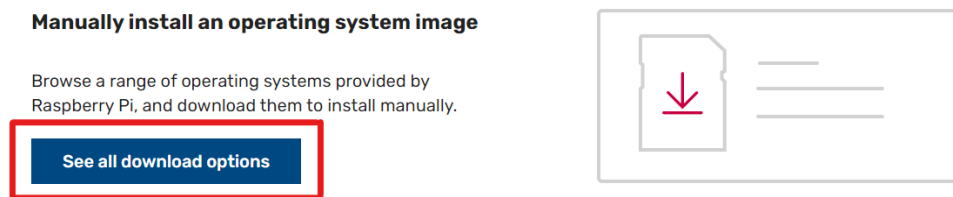
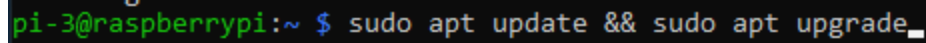


Ilustración 35 Referencia de donde sacar S.O Raspberry

Luego ya dentro de esta nueva página a la cual nos redirigió vamos a bajar y descargaremos la primera opción que nos aparecerá ya que esa es compatible para todas las Raspberry pi, además recomiendo que será la que dice “with desktop” ya que esta viene solo el sistema operativo junto a una interfaz de usuario como lo es Windows. Una vez ya obtenida la imagen solo falta obtener una aplicación para poder montarla en una tarjeta microSD.

Después de tener ya todo iniciado de forma correcta en la Raspberry podremos iniciar con la instalación de un servidor LAMP el cual es un paquete de software que incluye Apache (servidor Web), MySQL/MariaDB (base de datos) y PHP (lenguaje de programación). Este tipo de servidor permite tener una red local en un dispositivo, permitiendo ejecutar aplicaciones web sin necesidad de un servidor en línea.

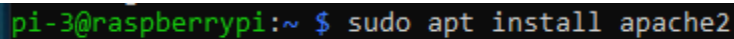
Comenzaremos abriendo una terminal de código en la Raspberry y una vez dentro digitaremos el siguiente comando.



```
pi-3@raspberrypi:~ $ sudo apt update && sudo apt upgrade_
```

Ilustración 36 Comando para actualizar paquetes de Raspberry

Este comando nos va a permitir actualizar todos los paquetes de nuestra Raspberry a su última versión, hay que tener en cuenta que esto tardará unos minutos y también necesitaremos que la Raspberry esté conectada a internet, en algún momento va a preguntar que si queremos continuar solo colocamos que sí y comenzara a trabajar sola en la actualización. Una vez ya terminada la actualización tendremos que proceder con la instalación de apache que se logra con el siguiente comando.



```
pi-3@raspberrypi:~ $ sudo apt install apache2
```

Ilustración 37 Comando para instalar apache

Al momento de correr este comando comenzaremos la instalación, pero luego de unos segundos al igual que en la actualización nos preguntará que si queremos continuar y le daremos que sí, el proceso de instalación la verdad no es tan tardado por lo que será un poco más rápido el avance. Luego de terminar la instalación podemos verificar si el apache se instaló correctamente, utilizando el comando “hostname -I” (ilustración 38) podremos obtener la dirección IP de la Raspberry que en mi caso será la 192.168.1.19 ya teniendo esta IP en mano podremos ingresar al servidor que va a estar alojado en la Raspberry por cualquier dispositivo ya sea un computador o bien un dispositivo móvil. Después de verificar

que el apache esté funcionando correctamente procederemos a la instalación de PHP.

```
pi-3@raspberrypi:~ $ hostname -I  
192.168.1.19 2803:c600:5125:f57a:4965:bd28:9b4a:4e82  
pi-3@raspberrypi:~ $ sudo apt install php
```

Ilustración 38 Comandos para obtener IP e instalar PHP

Al momento de implementar el comando para instalar PHP, automáticamente se instalará la última versión disponible que en estos momentos se encuentra en la 8.2.26, al igual que las otras veces preguntará si se quiere seguir con la instalación y solo se marca que sí.

```
pi-3@raspberrypi:~ $ sudo apt install mariadb-server php-mysql
```

Ilustración 39 Comando para instalar MySQL

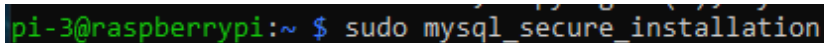
Una vez finalizada la instalación de PHP pasaremos a la instalación de MySQL, con el comando de la ilustración 39 se instalará el servidor de mariadb y phpmysql para poder hacer el enlace entre ellas y como es costumbre luego de unos segundos nos preguntará que si deseamos continuar. Después de que se finalice la instalación tenemos que reiniciar el servicio de apache con el fin de que se hagan los enlaces correspondientes y no ocurran fallas, para hacer el reinicio del apache se debe correr el siguiente comando.

```
pi-3@raspberrypi:~ $ sudo service apache2 restart
```

Ilustración 40 Comando para reiniciar servicios de apache

Ahora lo que toca hacer es la configuración inicial de MySQL para la cual debemos ingresar por medio del comando mostrado en la ilustración 41, al momento de correr el comando nos pedirá ingresar la contraseña perteneciente al usuario root, como la instalación es reciente

no se ha configurado ninguna contraseña por lo que solo debemos presionar la tecla enter, ya que, por defecto en nada.



```
pi-3@raspberrypi:~ $ sudo mysql_secure_installation
```

Ilustración 41 Comando para configuración inicial MySQL

Una vez ya aceptado nos preguntará si queremos crear una nueva contraseña para el usuario root, como recomendación y por seguridad yo diría que si se creara una nueva contraseña para evitar que cualquier persona entre. Luego de eso comenzara a preguntar sobre unas configuraciones básicas para MySQL como por ejemplo remover los usuarios anónimos que en mi caso coloque que sí, quitar el acceso remoto al usuario root el cual coloque que no, otras cosas que aparecen también son que, si queremos borrar la base de datos de prueba, que en lo personal si quise borrarla y por último pregunta si queremos recargar los privilegios, a lo cual tenemos que colocar sí.

Ya casi llegando al final de la configuración de la Raspberry tenemos que instalar una última cosa que es el phpmyadmin que se hace de la misma forma que las otras instalaciones y al igual que las otras instalaciones también nos va a pedir confirmación para continuar. En algún momento de la instalación aparecerá un recuadro que nos preguntara sobre qué servicio web vamos a querer usar con phpmyadmin, en el aparecen dos opciones apache2 y lighttpd como nosotros hemos instalado apache anteriormente, seleccionaremos la opción de apache con la tecla espacio y luego presionamos la tecla enter, para saber si se seleccionó correctamente se colocará un asterisco en la opción elegida.

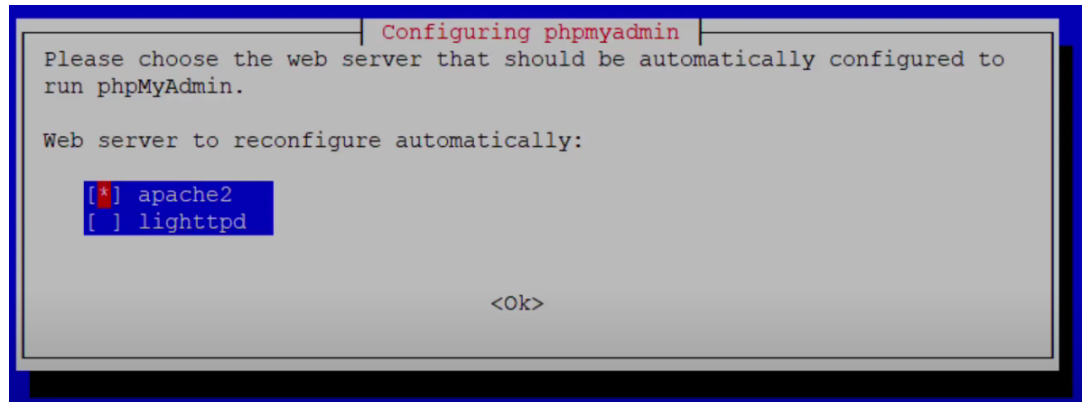


Ilustración 42 Cuadro selección servidor web

Después de hacer esa elección nos aparecerá otro recuadro en el cual nos preguntará que si queremos hacer la configuración de la base de datos de dbconfig-common, es muy importante que lo demos sí, porque simplifica y automatiza la configuración de bases de datos en sistemas Debian y derivados. Una vez se haya seleccionado que sí, nos pedirá una contraseña para phpmyadmin, la cual es muy distinta a la que se configuró en la base de datos anteriormente, luego que la ingresemos nos pedirá confirmarla y una vez confirmada la contraseña para phpmyadmin se continuará la instalación y una vez finalizada debemos proceder a enlazar PHP con MySQL utilizando el comando “sudo phpenmod mysqli” y una vez ejecutado ese comando debemos reiniciar los servicios de apache con el comando de la ilustración 40.

5.4.5 Creación modelo 3D

Como fue comentado en la propuesta de solución se buscaba implementar la impresión 3D como método de resguardar la parte de electrónica delicada que podría con el tiempo sufrir daños con la humedad, además esto nos ayudaría a tener el módulo de los sensores más ordenado y correr menos riesgos debido a que los cables no quedarían expuestos a que se manipulen de forma errónea. El modelado 3D es una

parte fundamental en cuanto a diseño y practicidad se refiere, puesto que, para este caso se debe pensar en tener un fácil acceso a las zonas importantes del prototipo, que serían, la placa junto a las conexiones de los sensores y el acceso a la fuente de alimentación destinada al sistema.

Lo bueno de implementar el modelado es que permite crear elementos según los requerimientos que uno necesite, si bien hay muchas páginas web dedicadas a lo que es el mundo de la impresión 3D no todos los diseños dentro de ellas satisfacen las necesidades de nuestro propio proyecto, puesto que, para nuestra propuesta posiblemente necesitemos que sea de un tamaño mayor o que necesitemos almacenar más cosas o no trabaje con los mismos componentes. Son muchos los factores que entran en juego, por lo que el modelado para este prototipo fue creado desde cero por medio del software de diseño Autodesk Fusion 360, permitiendo así trabajar con mediciones totalmente personalizadas para satisfacer las necesidades que se tenían para el hardware de este proyecto.

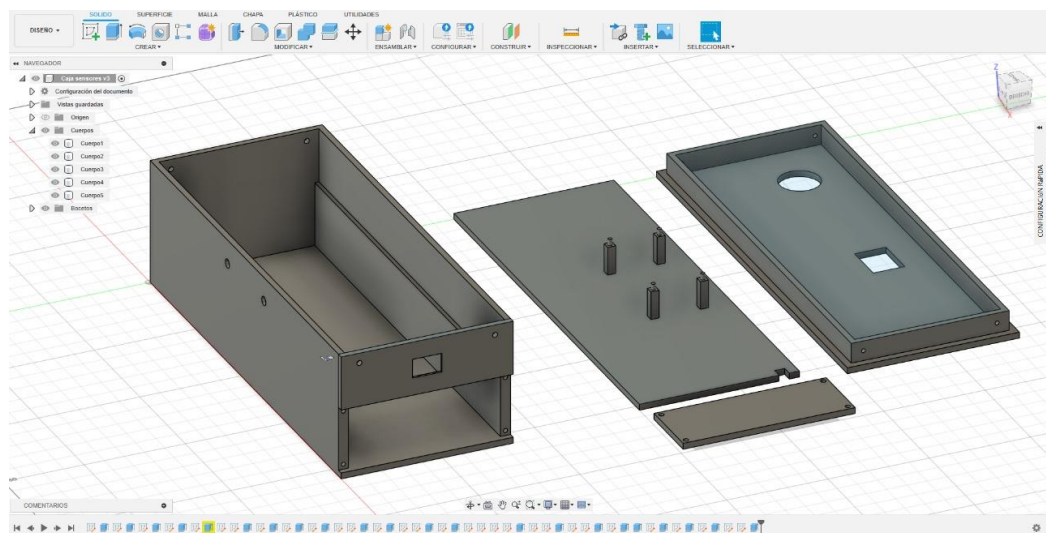


Ilustración 43 Modelado 3D caja módulo compacto

Además, se optó por utilizar PLA como material de impresión, ya que es biodegradable, fácil de trabajar y tiene un bajo impacto ambiental, ideal para entornos agrícolas donde se busca sostenibilidad (Food Pac Service, 2024).

Como se puede apreciar en la ilustración 43 se ha diseñado de una forma específica que nos permitirá tener acceso a las dos zonas claves mencionadas anteriormente. Explicando un poco de cómo va este diseño, tenemos el primer elemento que es la caja principal que está diseñada con un espacio dedicado para almacenar la alimentación, luego de eso tendría una especie de segundo piso el cual se dedicaría a contener toda la electrónica y conexiones, para finalmente tener una tapa superior que puede ser extraída para acceder de forma fácil y rápida en caso de hacer un mantenimiento, la forma de sellar esta tapa es por medio de tornillos que se encuentran ubicados por el alrededor de ella.

Una vez ya listo el modelado 3D debemos exportarlo en el formato STL para que así podamos pasar al siguiente software que es PrusaSlicer el cual nos permitirá crear el archivo necesario para colocarlo en la impresora 3D e imprimir la pieza.



Ilustración 44 Cuadro informativo del proceso de impresión

Esta parte del proceso es la más demorosa, puesto que, los tiempos de impresión dependerán de muchos factores como por ejemplo el tamaño o que tan detallada necesitemos la impresión, así como se aprecia en la ilustración 44 el tiempo que tardará en imprimir solo la caja principal será de 11 horas con 18 minutos aproximadamente y eso es solo una de las piezas, ya que, si calculamos el tiempo que llevará fabricar la pieza completa sería de 19 horas con 36 minutos aproximadamente.

5.4.6 Pruebas Prototipo mejorado

Dentro de las pruebas para saber si los datos se estaban enviando de forma correcta, también se puso a prueba la autonomía que podría tener la batería de 9V con las mejoras implementadas, los resultados fueron que la autonomía cayó abruptamente, quedándose sin energía luego de aproximadamente 3 horas, por lo que se tomó la decisión de cambiar la fuente de alimentación quedando conectado a una powerbank, la diferencia fue claramente notoria, ya que, con esta segunda fuente de alimentación logro estar sin problemas hasta 3 días y medio aproximadamente, considerando que la powerbank no estaba cargada a su máxima capacidad.

Una vez encontrada una fuente de poder que se pudiese tener varios días alimentando el sistema, comenzaron las tomas de lecturas para ver cómo funcionaba el sistema mejorado, como se esperaba las lecturas en la base de datos se ingresaron de forma correcta junto a la fecha y hora en la cual se registraron, teniendo además el identificador a cuál sensor pertenecían.

☐	 Editar	 Copiar	 Borrar	2023	3	2024-12-16 06:59:07	11.8
☐	 Editar	 Copiar	 Borrar	2024	1	2024-12-16 06:59:07	73
☐	 Editar	 Copiar	 Borrar	2025	4	2024-12-16 06:59:08	15
☐	 Editar	 Copiar	 Borrar	2026	5	2024-12-16 06:59:08	1073
☐	 Editar	 Copiar	 Borrar	2027	2	2024-12-16 07:04:14	76
☐	 Editar	 Copiar	 Borrar	2028	3	2024-12-16 07:04:15	12
☐	 Editar	 Copiar	 Borrar	2029	1	2024-12-16 07:04:15	74
☐	 Editar	 Copiar	 Borrar	2030	4	2024-12-16 07:04:15	15
☐	 Editar	 Copiar	 Borrar	2031	5	2024-12-16 07:04:15	1088
☐	 Editar	 Copiar	 Borrar	2032	2	2024-12-16 07:09:21	75
☐	 Editar	 Copiar	 Borrar	2033	3	2024-12-16 07:09:22	12.2
☐	 Editar	 Copiar	 Borrar	2034	1	2024-12-16 07:09:22	72
☐	 Editar	 Copiar	 Borrar	2035	4	2024-12-16 07:09:22	14.94
☐	 Editar	 Copiar	 Borrar	2036	5	2024-12-16 07:09:22	1087
☐	 Editar	 Copiar	 Borrar	2037	2	2024-12-16 07:14:29	75
☐	 Editar	 Copiar	 Borrar	2038	3	2024-12-16 07:14:29	12.2
☐	 Editar	 Copiar	 Borrar	2039	1	2024-12-16 07:14:29	68
☐	 Editar	 Copiar	 Borrar	2040	4	2024-12-16 07:14:30	14.94
☐	 Editar	 Copiar	 Borrar	2041	5	2024-12-16 07:14:30	1120
☐	 Editar	 Copiar	 Borrar	2042	2	2024-12-16 07:19:36	74
☐	 Editar	 Copiar	 Borrar	2043	3	2024-12-16 07:19:36	12.5
☐	 Editar	 Copiar	 Borrar	2044	1	2024-12-16 07:19:36	71
☐	 Editar	 Copiar	 Borrar	2045	4	2024-12-16 07:19:37	14.87
☐	 Editar	 Copiar	 Borrar	2046	5	2024-12-16 07:19:37	1133
☐	 Editar	 Copiar	 Borrar	2047	2	2024-12-16 07:24:43	74

Ilustración 45 Tabla de lecturas

En la ilustración anterior podemos ver un fragmento de las lecturas tomadas en el proceso de pruebas, como se puede ver y recordando un poco el id 1 pertenece al sensor capacitivo de humedad de suelo, el 2 pertenece al sensor DHT22 en su lectura de humedad ambiental, el 3 al mismo sensor DHT22 en su lectura de temperatura ambiental, el 4 correspondiente al sensor DS18B20 temperatura de suelo y el id 5 que corresponde al sensor MQ-135 concentración de dióxido de carbono. Las lecturas que podemos ver son en las primeras horas de la mañana, si nos enfocamos en las lecturas del sensor con id 3 que es la temperatura ambiental va incrementando poco a poco, pero si las tomas de lecturas

hubiesen seguido con un intervalo de 30 minutos nos toparíamos con cambios muy abruptos en la temperatura, ya que, tomando la referencia que aparece en la ilustración entre la lectura que se efectuó a las 06:59 de la mañana a la que se efectuó a las 07:19 han pasado solo 20 minutos y ya hay un incremento de 0.7° C en la temperatura, si bien no tiene el mismo impacto a distintas horas del día, al momento de tener altas temperaturas el haber seguido con los 30 minutos iniciales de intervalo entre lecturas no se podría haber tenido un monitoreo de variables confiable por lo que el cambio a 5 minutos fue una gran decisión, por el hecho que nos permite tener un tiempo más rápido en caso de detectar un alza repentina.

5.5 Segunda Iteración Y Retroalimentación

Durante esta segunda iteración se identificaron mejoras orientadas principalmente a la interfaz de usuario. Se propuso incluir vistas diferenciadas para mostrar datos históricos, promedios, últimas lecturas y exportación de registros en Excel. También se recomendó incorporar un sistema de semáforo para facilitar la interpretación de los datos, especialmente para usuarios sin conocimientos técnicos. Estas observaciones guiaron las mejoras implementadas y servirán como base para posibles ajustes finales.

En esta segunda iteración al ver que las lecturas ya se podían apreciar más fiables se enfocó principalmente en la interfaz de usuario en la cual se recomendó incorporar tres distintas vistas, donde se mostraran gráficos con los datos históricos, las últimas lecturas, los promedios de lecturas para cada sensor y una forma de poder obtener un archivo Excel con los datos de cada sensor.

Otra cosa que se destacó bastante para implementar en la interfaz de usuario, fue colocar un sistema de semáforo en las últimas lecturas con el fin

de tener una interpretación de los datos más sencilla, ayudando así a los usuarios que no tengan un conocimiento más técnico sobre el tema.

Con el fin de poder poner a prueba como va quedando la interfaz de usuario el sistema completo se dejó trabajando continuamente durante 2 semanas para poder tener una cantidad de datos más robustas y así poder apreciar de mejor manera como estarán funcionando las diferentes funciones que se implementarán.

5.6 Interfaz De Usuario

La interfaz de usuario ha sido diseñada para proporcionar una visualización clara y accesible de los datos obtenidos de sensores, almacenados en una base de datos central. Este componente del sistema facilita el acceso a la información en tiempo real, permitiendo a los usuarios interpretar rápidamente las lecturas capturadas por los sensores y tomar decisiones informadas.

En la página del índice o la de inicio se implementó una serie de cartas donde se mostrarán los datos de las últimas lecturas registradas en la base de datos además de un gráfico que permitirá visualizar los promedios de las lecturas de cada uno de los sensores. Para las cartas que mostrarán las últimas lecturas también se implementa un sistema de semáforo que ayuda a entender mejor los datos que aparecen, identificando rangos estándares para saber si se encuentran dentro del rango idóneo para el invernadero, la forma en que se consigue esto es la siguiente.

```

if($sensor['min'] < $row['valor'] and $row['valor'] < $sensor['max']){
    echo"<div class='col-md-3'>
        <div class='p-3 bg-white shadow-sm d-flex justify-content-around align-items-center rounded'>
            <div>
                <h3 class='fs-2'>". $row['valor'] . $sensor['unit'] . "</h3>
                <p class='fs-5'>". $sensor['name'] . "</p>
            </div>
            <i class='bx bx-check fs-1 primary-text border rounded-full p-3' style='background-color: green;'></i>
        </div>
    </div>";
}

```

Ilustración 46 Carta de últimas lecturas

El sistema de semáforo se trabaja a través de comprobaciones implementando un if, comparando la lectura junto a rangos preestablecidos óptimos dentro de un invernadero, visualizándose de la siguiente manera.

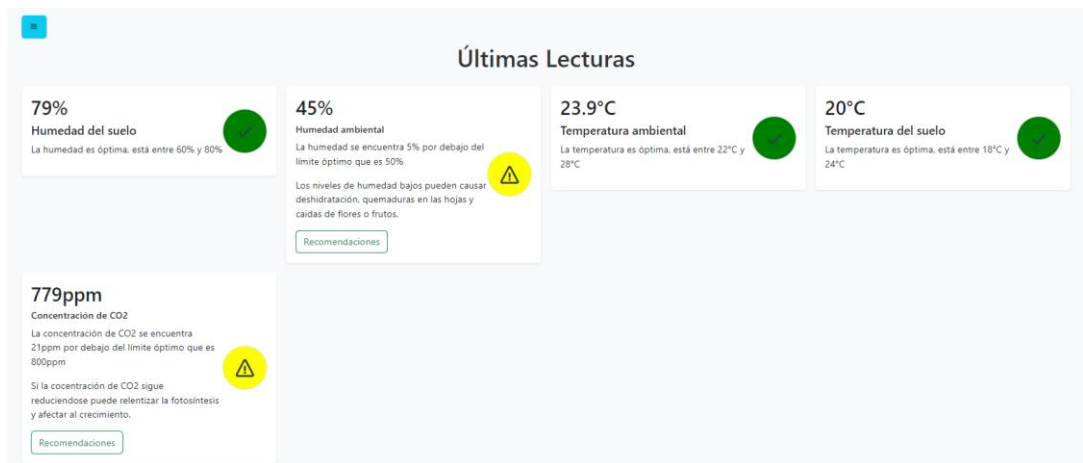


Ilustración 47 Cartas de últimas lecturas junto a sistema semáforo

Para los promedios de cada sensor se mostrarán por medio de un gráfico de barras utilizando las librerías de Google Charts y en cada una de las barras se colocará el valor arrojado por la consulta, esto se hizo con el fin de que el usuario pudiese leer el gráfico sin mayor dificultad, ya que, se espera que en el inicio sea un vistazo rápido a los datos almacenados, la forma en que se consiguió esto es de la siguiente.

```
function drawChart() {
    var data = google.visualization.arrayToDataTable([
        ['Sensor', 'Promedio', { role: 'annotation' }]],
        <?php
        include_once '../RB_Invernadero/config_db.php';

        $sql = "SELECT
            AVG(CASE WHEN id_sensor = 1 THEN valor END) AS humedad_suelo,
            AVG(CASE WHEN id_sensor = 2 THEN valor END) AS humedad_ambiental,
            AVG(CASE WHEN id_sensor = 3 THEN valor END) AS temperatura_ambiental,
            AVG(CASE WHEN id_sensor = 4 THEN valor END) AS temperatura_suelo,
            AVG(CASE WHEN id_sensor = 5 THEN valor END) AS concentracion_co2
        FROM
            lecturas";
        $result = $conn->query($sql);
        if ($result->num_rows > 0) {
            $row = $result->fetch_assoc();
            echo "[Humedad suelo', ".round($row['humedad_suelo']).", ".round($row['humedad_suelo'])."%',\n";
            echo "[Humedad ambiental', ".round($row['humedad_ambiental']).", ".round($row['humedad_ambiental'])."%',\n";
            echo "[Temperatura ambiental', ".round($row['temperatura_ambiental']).", ".round($row['temperatura_ambiental'])."°C',\n";
            echo "[Temperatura suelo', ".round($row['temperatura_suelo']).", ".round($row['temperatura_suelo'])."°C',\n";
            echo "[Concentración CO2', ".round($row['concentracion_co2']).", ".round($row['concentracion_co2'])."ppm']";
        }
    ]);
}
```

Ilustración 48 Estructuración gráfico de promedios

Tras obtener los datos de la base de datos, luego de ejecutar la consulta SQL que calcula los promedios de los diferentes sensores, es el momento de dibujar el gráfico de barras utilizando esos datos e implementando el formato requerido por Google Charts. Una vez implementada las configuraciones del gráfico donde se verá la orientación de las barras, los colores de estas, la posición de la leyenda o bien sacarla directamente y el estilo de las anotaciones entre otras cosas, podremos obtener el cómo se verá el gráfico, para mi caso con las configuraciones que implementé obtuve la siguiente visualización.

Promedio lecturas de sensores

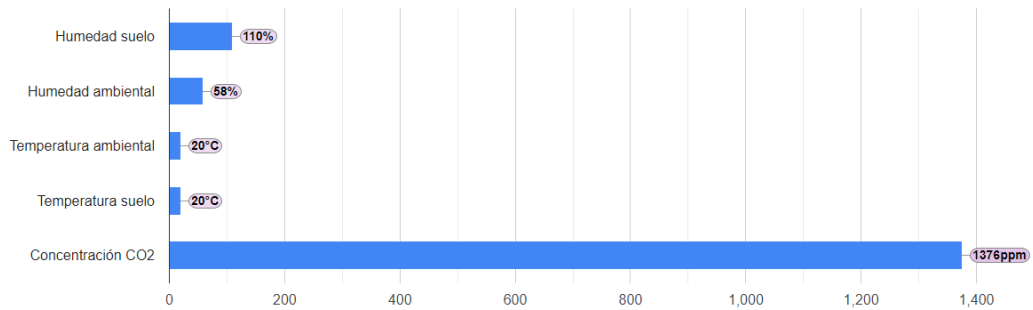


Ilustración 49 Gráfico de promedios

La página de gráficos está pensada para mostrar datos históricos y ver cómo se comportan las lecturas, por lo que se implementaron gráficos de líneas los cuales facilitan la legibilidad de las fluctuaciones en las lecturas. Estos, al igual que el gráfico anterior también se obtuvo de Google Charts solo que este se trabajó de forma distinta, ya que se necesitaba implementar un gráfico por cada sensor, además de implementar un filtro que ayude a buscar valores de diferentes fechas dependiendo lo que solicite el usuario por medio de un formulario implementando rangos de fechas. Para los gráficos que se visualizan al momento de ingresar a la página estos están programados para obtener los datos de acuerdo a la fecha actual, en otras palabras, al momento de hacer la consulta se obtiene la fecha desde el sistema y genera la consulta para obtener solo los datos que se han registrado ese mismo día, quedando de la siguiente forma.

```

$sql = "SELECT hora, valor FROM lecturas
WHERE fecha = '{$fecha_actual}'
and id_sensor = {$sensor['id']}
ORDER BY id_lectura DESC LIMIT 15";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
    echo"
    <html>
    <head>
    <script type='text/javascript' src='https://www.gstatic.com/charts/loader.js'></script>
    <script type='text/javascript'>
        google.charts.load('current', {'packages':['corechart']});
        google.charts.setOnLoadCallback(drawChart);

        function drawChart() {
            var data_". $sensor['id']. " = google.visualization.arrayToDataTable([
                ['Intervalos', 'Lecturas'],\n";
            while($row = $result->fetch_assoc()) {
                echo"['".(String)$row['hora']. "',". $row['valor']. "'],\n";
            }
            echo "]);

```

Ilustración 50 Código gráficos principales

En el caso de aplicar el filtro este enviará las fechas a la página principal de los gráficos donde serán recibidas por método POST y entregadas por medio de una función que conecta con la consulta SQL para obtener los datos y dibujar el gráfico, quedando de la siguiente forma el código de la consulta.

```

$sql = "SELECT hora, valor FROM lecturas
WHERE fecha BETWEEN '{$startDate}'
AND '{$endDate}' AND id_sensor = {$sensor['id']}";

```

Ilustración 51 Consulta SQL para filtro de datos

Como indiqué la página principal donde se muestran los gráficos sería la que recibe las fechas, sin embargo, en esta misma página se deben mostrar dos tipos de datos distintos, con esto me refiero a los que se

encuentran sin filtrar y los que se filtraron y eso se logra con la siguiente manera.

```
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $start_date = $_POST['start_date'];
    $end_date = $_POST['end_date'];

    function filtrarGrafico($startDate,$endDate){
        include '../RB_Invernadero/views/sensor_grafic.php';
    }

    filtrarGrafico($start_date,$end_date);
}else{
    include "../RB_Invernadero/views/sensor_grafic.php";
}
?>
```

Ilustración 52 Código gráficos con y sin filtro

Para el usuario la vista de todo esto será más reducida incluso se implementa un botón para poder descartar el filtro y volver a los gráficos de un inicio, quedando de esta manera.

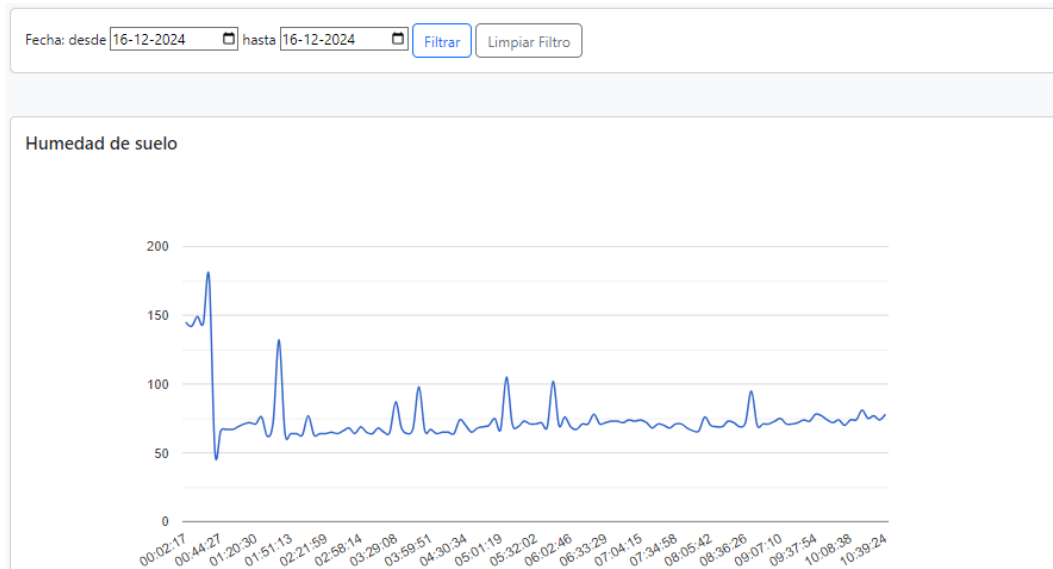


Ilustración 53 Vista usuario página de gráficos históricos

Por último, la página de las tablas otorga una vista más organizada y de fácil visualización, ya que muestra los datos de una manera conocida para el usuario, mostrando las últimas diez lecturas registradas por el sensor, sin embargo, otro propósito que tiene esta página es de otorgar la posibilidad de adquirir un archivo Excel con los datos de los sensores, la forma de implementar esto es adquiriendo los datos dependiendo de qué sensor se solicite, en otras palabras las lecturas contenidas en el archivo Excel solo serán las de un sensor que será elegido por el usuario. Para la creación de las tablas se implementa el siguiente código.

```
$sql = "SELECT fecha, hora, valor FROM lecturas
WHERE id_sensor = {$sensor['id']}
ORDER BY id_lectura DESC LIMIT 10";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    echo "...
    while($row = $result->fetch_assoc()){
        echo "
        <tbody>
        <tr>
        <td>".$row['fecha']."</td>
        <td>".$row['hora']."</td>
        <td>".$row['valor']."</td>
        </tr>
        </tbody>";
    }
}
```

Ilustración 54 Código de obtención de datos y creación tabla

Mientras que la implementación para la creación del archivo Excel se llevaría a cabo por medio de un formulario que entregará unos datos a la página donde se encuentra el código que genera el archivo viéndose de la siguiente manera.

```

<form action='/var/www/html/views/download_excel.php' method='post'>
  <input type='hidden' id='nombre' name='nombre' value='".$sensor['name']."'>
  <input type='hidden' id='id' name='id' value='".$sensor['id']."'>
  <input type='hidden' id='unit' name='unit' value='".$sensor['unit']."'>
  <button type='submit' class='btn btn-outline-success'>
    <i class='fas fa-file-excel'></i>
    Descargar Excel
  </button>
</form>

```

Ilustración 55 Formulario creación archivo Excel

Como se ve en la ilustración 55 se enviarán como un campo oculto los datos de los sensores como el id, el nombre y la unidad con la que trabaja, este botón estará disponible en cada una de las tablas que se genere, ya que enviará los datos respecto a la tabla en donde se encuentre.

Humedad de suelo (Sensor ID: 1)		
Descargar Excel		
Fecha	Hora	Valor (%)
2024-12-16	10:39:24	78
2024-12-16	10:34:17	74
2024-12-16	10:29:10	77
2024-12-16	10:24:02	75

Ilustración 56 Tabla datos sensores con descarga

Para la implementación de la descarga de un Excel con los datos debemos instalar Composer que nos ayudará a generar el archivo con ayuda de las bibliotecas de PHP para esto debemos primero instalar Composer en la Raspberry y para esto debemos implementar el siguiente comando que nos descargará el instalador “`wget -O composer-setup.php https://getcomposer.org/installer`”, una vez descargado el instalador tenemos que instalarlo y para eso implementamos este comando “`sudo php composer-setup.php --install-dir=/usr/local/bin --filename=composer`”, este comando instalará Composer en la dirección “/usr/local/bin” y además lo colocará disponible como comando de sistema. Para crear definitivamente el archivo Excel, solo falta instalar PhpSpreadsheet dentro del directorio donde se

encuentra nuestra página y lo podemos hacer con este comando “cd /var/www/html/” una vez ya dentro del directorio, ejecutaremos el siguiente comando “composer require phpooffice/phpspreadsheet” una vez ya terminada la instalación podemos trabajar tranquilamente en el código. Cabe recalcar que Composer en estos momentos necesita de la versión 8.1 hacia arriba de PHP por lo que se recomienda revisar la versión que se tiene instalada de PHP antes de comenzar con la instalación de esta herramienta y en caso de no tener la versión necesaria para instalar Composer solo ejecutaremos nuevamente el comando para instalar PHP de la ilustración 38.

Una vez en el código, para crear el archivo primero debemos incluir el archivo “autoload.php” generado por Composer, que carga automáticamente las clases necesarias para el proyecto, luego debemos importar las clases Spreadsheet y Xlsx de la biblioteca PhpSpreadsheet, junto al archivo que contiene la conexión a la base de datos y la recuperación de las variables enviadas por el método POST, quedando de la siguiente manera.

```
<?php
require '/var/www/html/vendor/autoload.php';

use PhpOffice\PhpSpreadsheet\Spreadsheet;
use PhpOffice\PhpSpreadsheet\Writer\Xlsx;

include_once '/var/www/html/config_db.php';

// Verificar que la solicitud sea POST
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $idSensor      = $_POST['id'];
    $nombreSensor  = $_POST['nombre'];
    $unidad        = $_POST['unit'];
}
```

Ilustración 57 Iniciación código descarga

Una vez terminado ese proceso comienza la configuración para lo que va a ir escrito en el archivo creando una nueva hoja de cálculo y escribiendo los encabezados de las columnas.

```
// Crear una nueva hoja de cálculo
$spreadsheet = new Spreadsheet(); Undefined type
$sheet = $spreadsheet->getActiveSheet();

// Escribir encabezados de las columnas
$sheet->setCellValue('A1', 'ID Lectura');
$sheet->setCellValue('B1', 'ID Sensor');
$sheet->setCellValue('C1', 'Fecha');
$sheet->setCellValue('D1', 'Hora');
$sheet->setCellValue('E1', 'Valor ('. $unidad. ')');
```

Ilustración 58 Preparación columnas de archivo Excel

En el siguiente paso solo resta crear y hacer la consulta a la base de datos, para traer todas las lecturas registradas del sensor seleccionado, implementando un incremento en una variable numérica para controlar la fila en la que se insertarán los valores.

```
// Consultar la base de datos para obtener las lecturas de los sensores
$sql = "SELECT id_lectura, id_sensor, fecha, hora, valor FROM lecturas WHERE id_sensor = {$idSensor}";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    $rowNumber = 2; // Empezar en la segunda fila para dejar la primera para los encabezados
    while ($row = $result->fetch_assoc()) {
        $sheet->setCellValue('A' . $rowNumber, $row['id_lectura']);
        $sheet->setCellValue('B' . $rowNumber, $row['id_sensor']);
        $sheet->setCellValue('C' . $rowNumber, $row['fecha']);
        $sheet->setCellValue('D' . $rowNumber, $row['hora']);
        $sheet->setCellValue('E' . $rowNumber, $row['valor']);
        $rowNumber++;
    }
}
```

Ilustración 59 Código contenido del archivo Excel

Ya finalizando con el proceso, debemos indicar que queremos crear un archivo Xlsx a partir de la hoja de cálculo previamente configurada, para luego hacer una limpieza del buffer de salida y evitar cualquier tipo de error que no nos permita acceder al archivo. Para facilitar el manejo del nombre

del archivo se crea una variable para modificar esa línea más adelante. Ya al final del código se implementan los encabezados HTTP para indicar que el contenido es un archivo Excel, que debe ser descargado como un archivo adjunto. Este tendrá un nombre específico y no será almacenado en caché. Finalmente, se guarda en la salida.

```
// Crear un archivo Excel y guardarlo
$writer = new Xlsx($spreadsheet);

// Limpiar el buffer de salida
if (ob_get_contents()) ob_end_clean();

$filename = "lecturas_".$nombreSensor.".xlsx";
// Enviar archivo para descarga
header('Content-Type: application/vnd.openxmlformats-officedocument.spreadsheetml.sheet');
header('Content-Disposition: attachment;filename="'.$filename.'"');
header('Cache-Control: max-age=0');
$writer->save('php://output');
exit();
```

Ilustración 60 Parte final creación archivo Excel

CAPÍTULO VI

Prototipo Final

6.1 Funcionalidades

El proyecto al comenzar tenía como objetivo desarrollar un sistema innovador que capture datos de diversos sensores, almacenarlos, monitorizarlos y controlarlos a través de una página web intuitiva y fácil de usar. Utilizando la metodología de prototipado, hemos podido iterar y refinar el diseño y la funcionalidad del sistema. A continuación, se detallan las funcionalidades finales que el sistema ofrece, destacando su capacidad para proporcionar datos precisos, mejorar la experiencia del usuario y garantizar la confiabilidad del monitoreo de los sensores.

1. Captura de datos de sensores: el módulo de sensores es capaz de tomar lecturas a intervalos de cinco minutos con el fin de tener una mejor precisión al momento de tomar una decisión.
2. Almacenamiento de datos: los datos capturados deben almacenarse en una base de datos.
3. Visualización de datos en tiempo real: mostrar los datos de los sensores en la página web en tiempo real usando gráficos y/o tablas.
4. Historial y Análisis de datos: permitir que el usuario tenga acceso a los datos históricos para realizar análisis.
5. Interfaz de usuario intuitiva: Una página web fácil de usar que permita a los usuarios navegar y entender los datos sin dificultad.
6. Exportación de datos: Permitir a los usuarios exportar los datos en formato Excel.
7. Compatibilidad multidispositivo: que la página web se vea y funcione bien en dispositivos móviles, tabletas y computadoras de escritorio.

Varias de estas funcionalidades se pueden visualizar fácilmente en la página web con tan solo navegar por ella. A continuación, se mostrarán capturas de pantalla mostrando la página web tanto en pantalla de celular como en pantalla de computador para tener una mejor idea de cómo se visualiza la página.

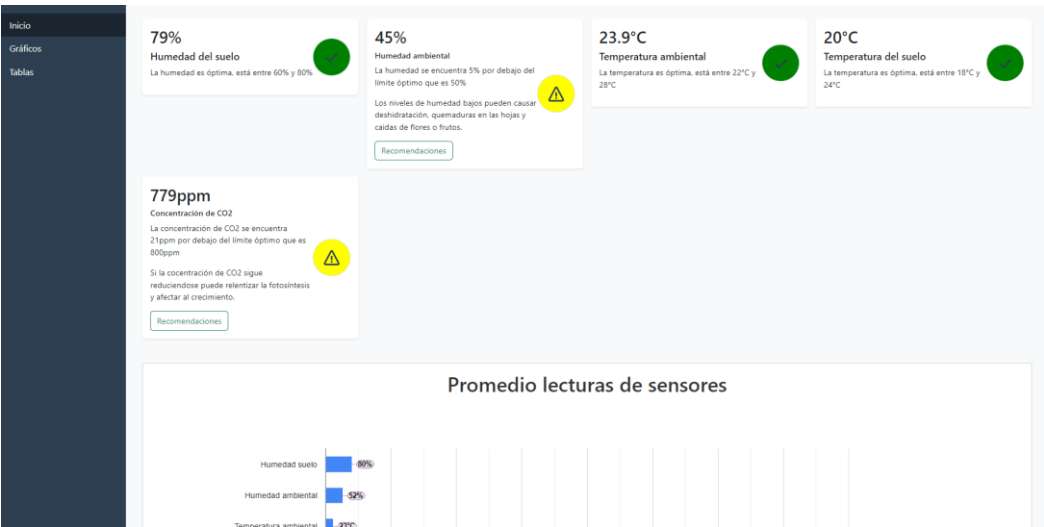


Ilustración 61 Interfaz visual de inicio



Ilustración 62 Interfaz visual gráficos históricos

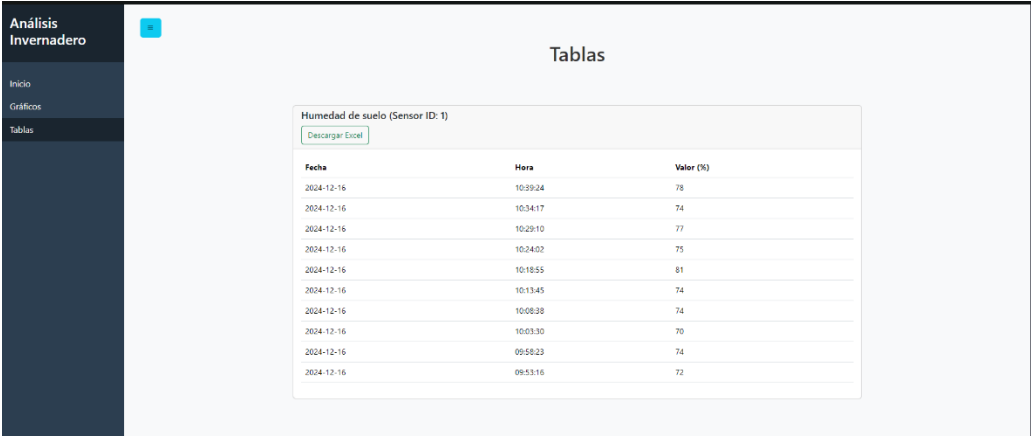


Ilustración 63 Interfaz visual tablas de los últimos 10 datos

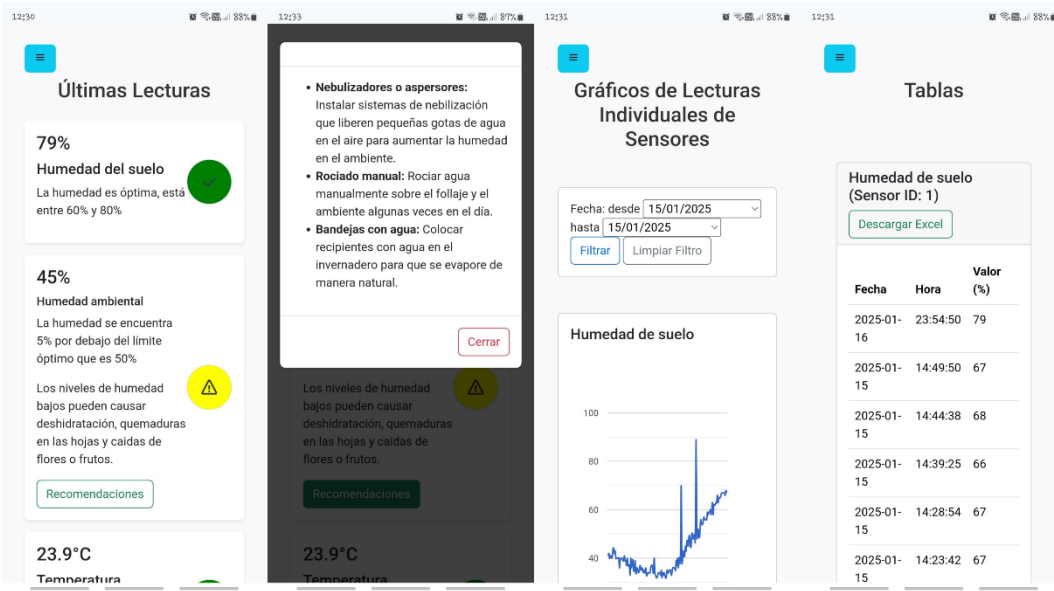


Ilustración 64 Interfaz visual en dispositivo móvil

CAPÍTULO VII

Conclusión

El desarrollo del proyecto permitió implementar una solución tecnológica eficiente para el monitoreo y control de variables medioambientales esenciales en invernaderos, destacando la integración de tecnologías accesibles y de bajo costo como microcontroladores, sensores especializados y una Raspberry Pi. Los resultados obtenidos durante las pruebas validaron la eficacia del sistema propuesto, evidenciando su capacidad para registrar datos precisos y ofrecer retroalimentación en tiempo real, lo que optimiza el manejo de recursos y las condiciones de cultivo.

La metodología basada en prototipos fue clave para la iteración constante y la mejora del diseño, permitiendo adaptar el sistema a los requerimientos detectados en cada etapa. Asimismo, el uso de impresión 3D para proteger los componentes electrónicos demostró ser una medida efectiva para aumentar la durabilidad del hardware en condiciones adversas.

El alcance del proyecto, si bien principalmente se tenía pensado para invernaderos de baja envergadura, al ir desarrollándolo se dio cabida a que este mismo sistema es lo suficientemente accesible para poder implementarlo incluso en invernaderos domésticos donde claramente se necesitan parámetros distintos a los trabajados en el proyecto, pero una vez conociéndolos se puede ajustar el sistema para su uso.

Finalmente, este proyecto no solo cumple con los objetivos planteados, sino que también abre oportunidades para futuras mejoras, como la incorporación de más parámetros de monitoreo, funcionalidades avanzadas de predicción y análisis, una mayor escalabilidad para diferentes aplicaciones agrícolas. Incluso, en el ámbito de la electrónica, se puede pensar en un sistema totalmente sostenible en términos energéticos, por medio de paneles solares los cuales

permitirán que los sensores nunca queden sin energía, ya que es el factor que aún necesita refinamiento. Sin embargo, el éxito de este proyecto radica en la combinación de innovación tecnológica y enfoque práctico, contribuyendo al desarrollo sostenible en la agricultura de precisión.

REFERENCIAS

- Aguirre, E. A., Pira, D. D. B., Rodríguez, J. S. S., Mesa, H. C. B., & Castro, D. A. V. (2019). Red de sensores inalámbricos para el monitoreo de variables agroecológicas en cultivos bajo invernadero. *Publicaciones e Investigación*, 13(1), 53-60.
- Arancio Fernández, M. (2020). *Monitoreo y control de variables físicas en un invernadero a través del Internet de las Cosas* [Tesis de grado, Universidad de Tarapacá].
- alldatasheet.com. (n.d.). MQ135: Gas sensor datasheet. <https://pdf1.alldatasheet.com/datasheet-pdf/view/1307647/WINSEN/MQ135.html>
- *Capacitive Soil Moisture Sensor SKU_SEN0193-DFRobot.* (n.d.). https://wiki.dfrobot.com/Capacitive_Soil_Moisture_Sensor_SKU_SEN0193?form=MGOAV3
- Duque, S. I. O. (2017). Monitoreo y control de variables ambientales mediante una red inalámbrica para agricultura de precisión en invernaderos. *Revista vector*, 12, 51-60.
- Food Pac Service. (2024, 23 agosto). ¿Qué es el PLA? y para qué se utiliza | Food PAC Service. *Food Pac Service*. <https://foodpacservice.com/centro-de-conocimiento/que-es-el-pla-y-para-que-se-utiliza/>
- Liu, T. (2013). Digital-output relative humidity & temperature sensor/module DHT22 (DHT22 also named as AM2302). *Aosong Electronics Co., Ltd.*
- LME Editorial Staff. (2023, 23 noviembre). *ESP32 Pinout reference*. Last Minute Engineers. <https://lastminuteengineers.com/esp32-pinout-reference/>
- Martínez, J. P., Farías, K., & Ruiz, R. (2020). *Informativo n.º 36: Factores limitantes para la producción hortícola*. Instituto de Investigaciones Agropecuarias (INIA). <https://biblioteca.inia.cl/server/api/core/bitstreams/438cedaa-8b6c-480f-9b3b-b09fd6ad0462/content>
- Manuti. (2019, 19 diciembre). *Raspberry Pi 2 model B*. Raspberry Para Torpes. <https://raspberryparatorpes.net/hardware/raspberry-pi-2-model-b/>
- *Módulo Detector de humedad del suelo – MCI Electronics.* (n.d.). <https://mcielectronics.cl/shop/product/modulo-detector-de-humedad-del-suelo-30176/>

- Pressman, R. S. (2010). *Ingeniería de software: Un enfoque práctico* (7.^a ed.). McGraw-Hill.
- *Sensor de temperatura y humedad DHT11 - Shield Picuino UNO - Picuino.* (n.d.). <https://www.picuino.com/es/uno-doc/dht11.html>
- *SparkFun Soil Moisture Sensor.* (n.d.). SEN-13322 - SparkFun Electronics. <https://www.sparkfun.com/products/13322?form=MG0AV3>
- Texas Instruments. (2019). *SimpleLink™ Microcontroller System-Level ESD Protection* (Application Report No. SPRAD89). Recuperado de <https://www.ti.com/lit/an/sprad89/sprad89.pdf>
- Teja, R. (2024, 21 septiembre). *ESP32 ADC Tutorial | How to use ADC in ESP32?* ElectronicsHub. <https://www.electronicshub.org/esp32-adc-tutorial/>
- User, S. (2016, 7 septiembre). *Clima de un Invernadero. ¿Cómo conseguir la Temperatura Ideal?* <https://www.novagric.com/es/blog/articulos/clima-invernadero-como-conseguir-temperatura-ideal>
- Ventilación, S. (2023, 26 abril). *Niveles recomendados de CO2, temperatura y humedad en el hogar.* El Blog De La Ventilación Inteligente. <https://www.siberzone.es/blog-sistemas-ventilacion/niveles-recomendados-co2-temperatura/>