



Universidad de Panamá
Centro Regional Universitario de Coclé
Facultad de Informática, Electrónica y Comunicaciones
Licenciatura en Ingeniería Informática
Teleinformática II & Electrónica Digital II



PROYECTO SEMESTRAL

SISTEMA DE MONITOREO Y CONTROL DE RIEGO AUTOMÁTICO PARA UNA CASA DE CULTIVO

Estudiante: Marichell Araúz
Doris Arcia
Caroline Bethancourth
Kevin García
Jorge Mendoza
Cesar Quijada
Isabel Rodríguez
Javier Rodríguez
Joseline Sánchez
Osvaldo Torrero
Carlos Ureña

Calificación: _____
Profesor: Héctor R. Rosales M.

Fecha: 2 de octubre de 2023
Entrega: Día del Examen Semestral

OBJETIVO

- Implementar un sistema de monitoreo y control automático de riego para una Casa de Cultivo.
- Aplicar diversas tecnologías relacionadas a la Teleinformática y a la Electrónica Digital, con el propósito de ofrecer posibles soluciones a problemas reales en nuestro entorno.

PLANTEAMIENTO

Según el perfil de egresado de un estudiante de Ingeniería en Informática, este debe ser un profesional que posea un conocimiento interdisciplinario, científico y tecnológico; conocimientos que son útiles para aplicar la informática en diversos campos como la industria, la teleinformática, la educación, la salud, lo social, producción agrícola, entre otros. En el campo agrícola, las Casas de Cultivo vienen representando una gran alternativa para producir alimentos dentro de un ambiente controlado. La implementación de Casas de Cultivo con sistemas de monitoreo y control automático de riego, representan una alternativa tecnológica para que las familiares familias puedan cultivar algunos de los alimentos que se consumen en el hogar.

En este proyecto se plantea el diseño y la implementación de una pequeña casa de cultivo en la residencia de unos de los estudiantes del curso. Se propone que el sistema pueda obtener y registrar datos de diversas variables como temperatura y humedad ambiente, radiación solar interna y externa, humedad de suelo y medir el volumen de agua que se consume en el riego. Los datos deben registrarse

durante cinco días y almacenarlos en una memoria SD, y ser publicados en servidor con acceso a internet. También, se propone la toma de imágenes fotográficas de la evolución del cultivo. El grupo debe utilizar como referencia los laboratorios parciales realizados en los cursos de Teleinformática I y II, y Electrónica Digital I y II. Al finalizar el proyecto, se debe tener una data disponible en internet de las diferentes variables registradas por el sistema.

CONTENIDO

INTRODUCCIÓN	- 6 -
DESCRIPCIÓN GENERAL DEL PROYECTO.....	- 7 -
Plataforma utilizada para ver los datos.....	- 8 -
DESCRIPCIÓN DEL SOFTWARE Y HARDWARE UTILIZADOS.....	- 10 -
SOFTWARE THONNY MICROPYTHON.....	- 10 -
SOFTWARE IDE ARDUINO.....	- 10 -
HARDWARE	- 11 -
➤ Tarjeta ESP32.....	- 11 -
➤ ESP32 CAM	- 12 -
➤ Módulo Relé	- 12 -
➤ Caudalímetro	- 13 -
➤ Electroválvula solenoide.....	- 14 -
➤ Sensor UV	- 14 -
➤ Sensor de humedad del suelo	- 15 -
➤ Sensor DTH 22	- 15 -
➤ Redes de malla	- 16 -
➤ Memoria microSD 16 GB.....	- 17 -
➤ Extensión (corriente).....	- 17 -
➤ PCB.....	- 18 -
➤ Cables UTP.....	- 19 -
➤ Cables de Red	- 19 -
➤ Cable micro USB	- 20 -
➤ Adaptador de 5V	- 20 -
➤ Tubos de PVC.....	- 21 -
➤ Cargador de 12 V y 2 A	- 21 -
DESCRIPCIÓN GENERAL DEL DIAGRAMA ESQUEMATICO	- 23 -
DESCRIPCIÓN DE CADA PARTE DEL DIAGRAMA.....	- 23 -
DESCRIPCIÓN DEL CÓDIGO DE LA ESP32.....	- 24 -
Documentación del Código en MicroPython para proyecto con ESP32	- 24 -
Descripción General	- 24 -
Configuración de Conexión Wi-Fi	- 24 -
Tokens de los Bots de Telegram.....	- 25 -
TI-II&ED-II-2023-PROYECTO SEMESTRAL	

Control de Electroválvula	- 25 -
Función para Enviar Mensajes a Telegram.....	- 25 -
Descripción Detallada de envío a Telegram	- 26 -
Sensores de Humedad en el Suelo	- 27 -
Sensor de Flujo de Agua	- 27 -
Sensores UV Interior y Exterior	- 27 -
Sensores DHT22 (Temperatura y Humedad).....	- 28 -
Bucle Principal	- 28 -
DESCRIPCIÓN DEL CÓDIGO DE LA ESP32 CAM	- 28 -
Código de la Esp32-cam	- 28 -
Descripción de cada parte del código de la ESP32 CAM	- 36 -
Librerías:.....	- 36 -
Credenciales del WIFI y los token e ID de Telegram	- 37 -
configInitCamera	- 37 -
handleNewMessages	- 38 -
sendPhotoTelegram	- 38 -
setup.....	- 40 -
loop:.....	- 40 -
initMicroSDCard	- 41 -
tomarecorrente:.....	- 41 -
takeSavePhoto:.....	- 42 -
DESCRIPCIÓN DE LAS GRÁFICAS POR DÍA	- 43 -
Gráficas día 1	- 43 -
➤ Gráfica de temperatura con sensores DTH22	- 43 -
➤ Gráfica de humedad con sensores DTH22	- 43 -
➤ Gráfica con sensores de humedad de suelo.....	- 44 -
Gráficas día 2	- 44 -
➤ Gráfica de temperatura con sensores DTH22	- 44 -
➤ Gráfica de humedad con sensores DTH22	- 45 -
➤ Gráfica con sensores de humedad del suelo.....	- 45 -
➤ Gráfica de temperatura con DTH22	- 46 -
Gráficas día 3	- 46 -

➤ Gráfica de humedad con DTH22	- 46 -
➤ Gráfica con sensores de humedad del suelo.....	- 47 -
Gráficas día 4	- 47 -
➤ Grafica con sensores UV	- 47 -
➤ Gráfica con sensores de humedad de suelo.....	- 48 -
Gráficas día 5	- 48 -
➤ Gráfica de temperatura con DTH22	- 48 -
➤ Gráfica de humedad con DTH22	- 49 -
➤ Gráfica con sensores UV	- 49 -
➤ Gráfica con sensores de humedad de suelo.....	- 50 -
OBSERVACIONES	- 50 -
IMÁGENES TOMADAS CON LA ESP32 CAM	- 51 -
APRENDIZAJES SIGNIFICATIVOS	- 53 -
APORTES INDIVIDUALES	- 56 -
LINK DEL VIDEO	- 58 -
CONCLUSIÓN	- 59 -
BIBLIOGRAFÍA	- 61 -
EVALUACIÓN	- 62 -

INTRODUCCIÓN

El proyecto se centra en el desarrollo de un sistema de monitoreo y control automático de riego para una casa de cultivo ubicada en El Encanto de Penonomé, provincia de Coclé. La implementación se llevó a cabo en un espacio amplio y plano, donde se preparó la tierra y se construyó una estructura de 2 metros cuadrados y 2 metros de altura para el cultivo de tomates y ajíes. La estructura fue cubierta por mallas para cultivo.

Se creará un sistema de monitoreo y riego automático donde se utilizarán diversos componentes de hardware y software.

En hardware, se implementarán sensores DTH22 de temperatura y humedad, sensores de humedad de suelo, sensores de radiación UV, una electroválvula solenoide, un caudalímetro, una ESP32 para conectar todos los sensores y controlar el riego, y una ESP32 CAM para tomar fotografías periódicas de los plántones.

El software se usa programas como Thonny con MicroPython para programar la ESP32 y el IDE de Arduino para la ESP32 CAM.

La finalidad del sistema consiste en monitorizar variables ambientales críticas para el óptimo desarrollo de los cultivos, como temperatura, humedad, radiación solar UV y humedad del suelo. Asimismo, el sistema permitirá automatizar el riego de los plántones activando una electroválvula cuando los niveles de humedad del suelo descienden por debajo de ciertos umbrales considerados ideales.

Los datos se registran durante cinco días y se almacenan en una memoria SD, y se publican en un servidor con acceso a través de una página web.

DESCRIPCIÓN GENERAL DEL PROYECTO

El proyecto consiste en desarrollar un sistema de monitoreo y control automático de riego para una Casa de Cultivo en este caso se realizó en casa de un integrante del grupo ubicada en el Encanto de Penonomé, provincia de Coclé, un lugar amplio y plano en donde los estudiantes realizaron la preparación de la tierra, se hizo la estructura que tiene una medida de 2 m² y altura de 2 metros y se procedió a realizar la siembra de tomates y ajís que fueron los plantones que se utilizaron. Esta pequeña casa de cultivo fue cubierta por todos sus lados con malla para cultivo.

Se utilizaron sensores DTH 22 de húmeda/temperatura, sensores de humedad de suelo, sensores de UV, una electroválvula solenoide, un caudalímetro, una ESP32 y una ESP32 CAM que es para las fotos; todo esto a excepción de la ESP32 CAM fue programado mediante el IDE de Thonny de MicroPython.

La ESP32 CAM fue programada con el IDE de Arduino, para monitorear por un periodo de 5 días calendario y estos datos se pueden ver en tiempo real.

Existen una gran variedad de sistemas autónomos de riego para casas de cultivos y huertos caseros pero estos sistemas son muy costosos y muchas veces las personas no lo pueden implementar, mientras que un sistema de riego basado en IoT permite obtener resultados de huertos con bastante precisión y aun costo flexible para la población ya que implementar un sistema de esto es de mucha importancia ya que se reduciría el consumo de agua porque se regaría solamente cuando los plantones lo necesita esto significaría un alto ahorro económico y beneficio para el ambiente.

Durante los días de monitoreo en la casa de cultivo se presentaron ciertos inconvenientes como lo fueron los 3 primeros días los sensores UV no enviaron datos ya que en una investigación que se dio los pines que utilizamos ya que hay una condición que si se está utilizando la señal WIFI los puertos ADC 2 no funcionan y por esta razón los sensores UV no enviaron datos y se tomó la decisión de desconectar un sensor del suelo quedando solamente 4 esto para poder conectar los otros sensores para liberar un puerto ADC 1.

Los DHT22 no enviaron datos durante parte del día 4 ya que al cambiarlos de pines 36 y 39 que en la ESP32 son VP Y VN desactivaban los pines GPIO derecha de la ESP32.

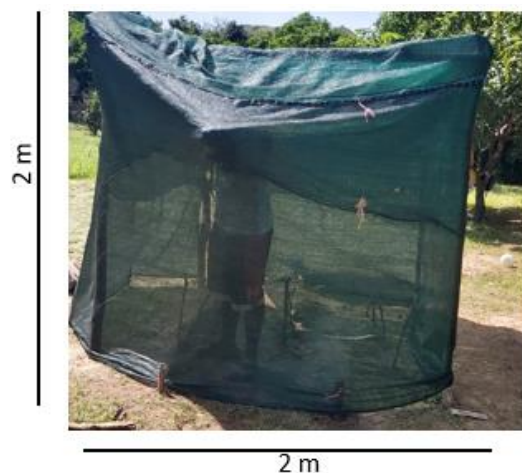


Ilustración 1 Medidas de la casa de cultivo



Ilustración 2 Imagen de los plantones dos días después de sembrados

Plataforma utilizada para ver los datos



Telegram

Ilustración 3 Telegram plataforma utilizada para ver los datos de la casa de cultivo

Telegram es una aplicación de mensajería instantánea que permite la comunicación entre usuarios de manera rápida y segura. En el contexto de una casa de cultivo, se puede utilizar Telegram para monitorear y controlar diversos aspectos, como temperatura, humedad y sistemas de riego, a través de Bots personalizados.

Los Bots en Telegram actúan como interfaces automáticas que pueden proporcionar información, recibir comandos y ejecutar acciones predefinidas. En una casa de cultivo, esto facilita el seguimiento y la gestión remota de condiciones ambientales y sistemas de riego, permitiendo a los usuarios recibir actualizaciones y realizar ajustes a través de la aplicación.

Los Bots permiten centralizar y simplificar el monitoreo de Datos ambientales, Sensores UV, Caudalímetro, Humedad del suelo y Cultivo32cam.

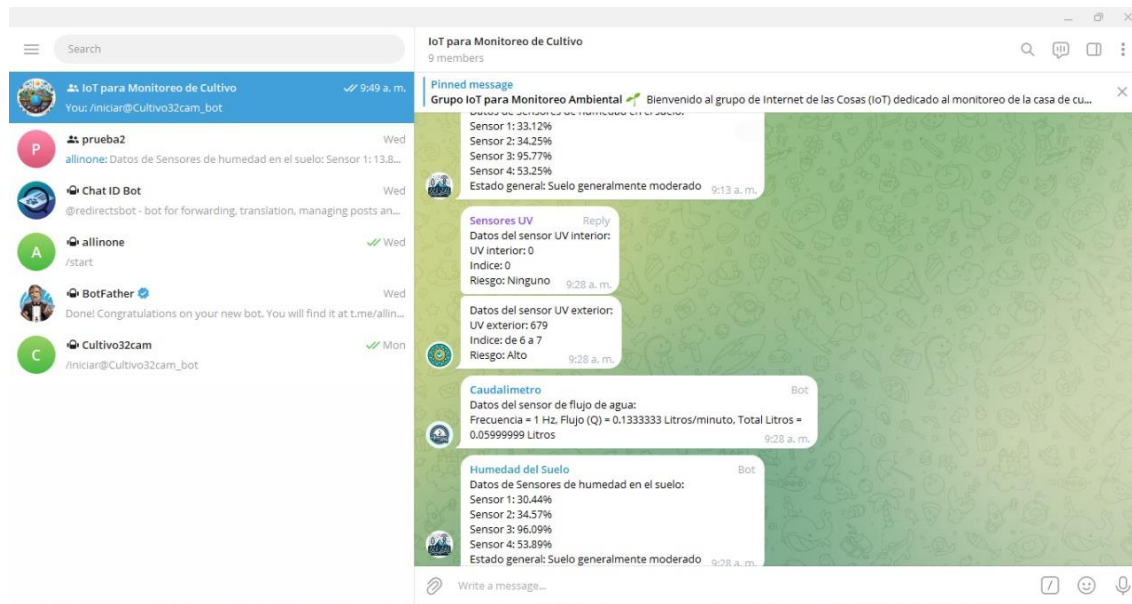


Ilustración 4 Captura donde se observan los datos que envían los bot en el Telegram y en una esquina se observan los bot que están en este grupo

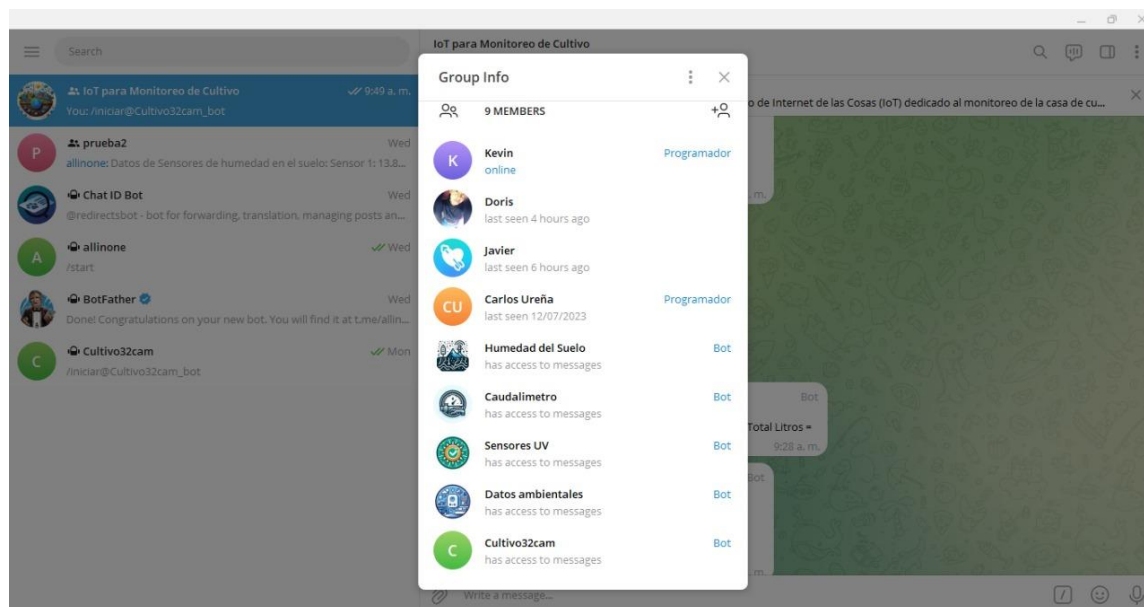


Ilustración 5 Captura donde se observa los miembros del grupo de visualización de datos de la casa de cultivo.

DESCRIPCIÓN DEL SOFTWARE Y HARDWARE UTILIZADOS

SOFTWARE THONNY MICROPYTHON

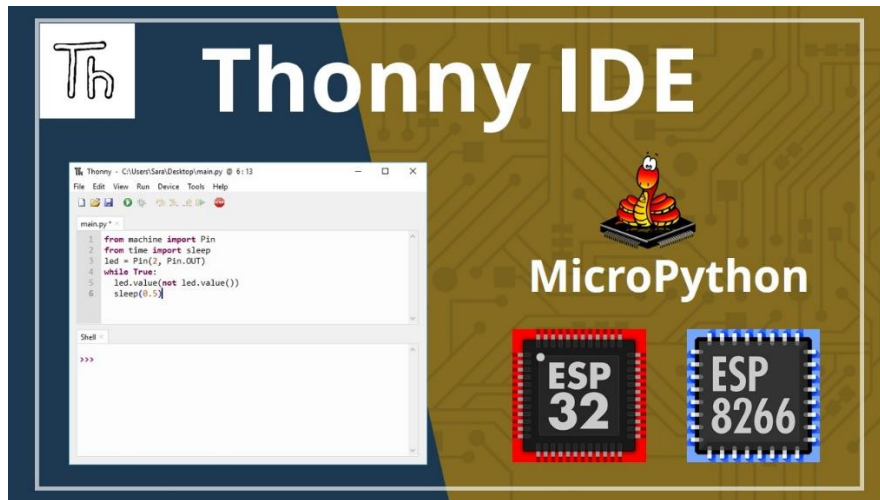


Ilustración 6 Software de thoonny utilizado para la programación de este sistema de monitoreo

Thonny es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) diseñado específicamente para programar en Python. Es una herramienta amigable y fácil de usar, ideal para principiantes en la programación. Thonny incluye un editor de código, un administrador de paquetes integrado para instalar y gestionar bibliotecas de Python, así como un depurador para ayudarte a identificar y corregir errores en tu código. Además, cuenta con una interfaz sencilla que facilita la escritura y ejecución de programas Python, convirtiéndolo en una excelente opción para aquellos que están dando sus primeros pasos en el mundo de la programación.

SOFTWARE IDE ARDUINO



Ilustración 7 IDE de Arduino utilizado para la programación de la ESP32 CAM

Definición: IDE – entorno de desarrollo integrado, llamado IDE (sigla en inglés de integrated development environment), es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios.

Uso en el proyecto: Arduino IDE se utilizó para toda la programación de la ESP32 CAM

Características: Es un editor de código con resaltado de sintaxis, bibliotecas predefinidas y un monitor serie para la depuración, el Arduino IDE facilita el desarrollo de proyectos y la programación de microcontroladores Arduino.

HARDWARE

➤ Tarjeta ESP32



Ilustración 8 Tarjeta ESP32 utilizada para el sistema de monitoreo

Definición: El módulo ESP32 es una solución de Wi-Fi/Bluetooth todo en uno, integrada y certificada que proporciona no solo la radio inalámbrica, sino también un procesador integrado con interfaces para conectarse con varios periféricos.

Uso en el proyecto: Se utilizó para manejar las conexiones de los sensores a través de una programación en Thonny IDE con Python.

Características

- Voltaje de Alimentación (USB): 5V DC
- Voltaje de Entradas/Salidas: 3.3V DC
- Placa: ESP32 DEVKIT V1 (Espressif)
- SoM: ESP-WROOM-32 (Espressif)
- SoC: ESP32 (ESP32-D0WDQ6)
- CPU: Dual-Core Tensilica Xtensa LX6 (32 bit)
- Frecuencia de Reloj: hasta 240Mhz
- Desempeño: Hasta 600 DMIPS
- Procesador secundario: Permite hacer operaciones básicas en modo de ultra bajo consumo
- Wifi: 802.11 b/g/n/e/i (802.11n @ 2.4 GHz hasta 150 Mbit/s)
- Bluetooth: v4.2 BR/EDR and Bluetooth Low Energy (BLE)

➤ ESP32 CAM

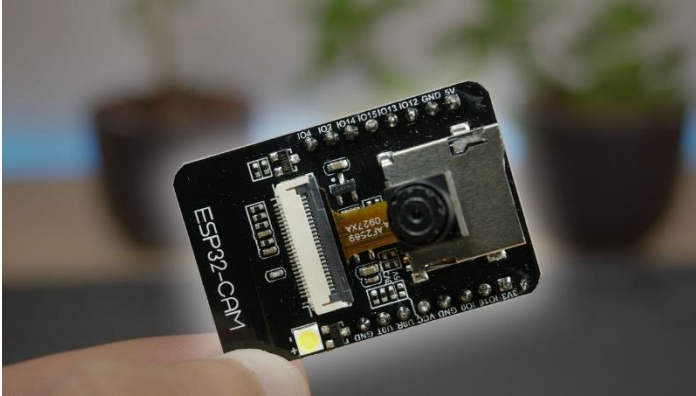


Ilustración 9 ESP32 CAM utilizada para la toma de fotos de los plantones

Definición: es un módulo de cámara muy pequeño con el chip ESP32-S, cuenta con una tarjeta con ranura de microSD que puede resultar útil para almacenar imágenes tomadas con la cámara.

Uso en el proyecto: se utilizó para la toma de imágenes de los plantones en la casa de cultivo.

Características

- El módulo Soc BT WI-FI 802.11b/g/n más pequeño.
- CPU de 32 bits de bajo consumo, también puede servir como procesador de aplicaciones
- Velocidad de reloj de hasta 160 MHz, potencia informática resumida de hasta 600 DMIPS.
- SRAM incorporada de 520 KB, 4 MPSRAM externa.
- Soporta UART/SPI/I2C/PWM/ADC/DAC
- Admite cámaras 0V2640 Y 0V7670, lámpara de flash incorporada
- Admite cargas de imágenes WIFI
- Admite múltiples modos de suspensión
- admite tecnología Smart Config/AirKiss

➤ Módulo Relé



Ilustración 10 módulo Relé

Definición: está constituido internamente por una bobina de cobre, la cual genera un campo magnético y acciona un interruptor, al momento que ocurre esto, se permite el paso de voltajes altos

Utilidad en el proyecto: se utilizó para el encendido de la electroválvula.

Características:

- Voltaje de fuente de alimentación: 5VDC
- Corriente de más de 100mA
- Recogida de bajo nivel, liberación de alto nivel
- Relés de alta calidad

➤ Caudalímetro



Ilustración 11 Caudalímetro sensor de flujo de agua

Definición: Se puede medir el paso de agua y gracias a este se puede saber la cantidad de fluido, la cual será entregada a la casa de cultivo, de esta manera se obtiene el consumo de agua, dentro del captador de flujo tiene implementado una rueda constituida por un imán y un sensor hall que detecta el cruce del dispositivo magnético

Uso en el proyecto: medir la cantidad de agua que se utilizara en la casa de cultivo

Características

- Rango de voltaje de funcionamiento: CC 5-18 V
- Diámetro de la rosca externa: G1/2"
- Rango de flujo: 1-30 l/min
- Permite la compresión: presión de agua 1.75 Mpa por debajo
- Forma de onda de salida: onda cuadrada, pulso de salida signal

➤ Electroválvula solenoide

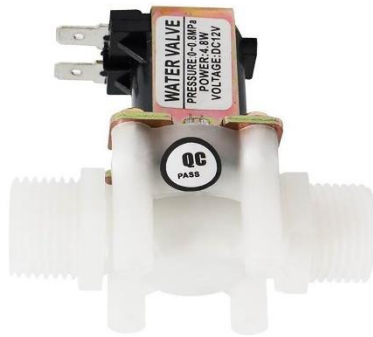


Ilustración 12 Electroválvula solenoide

Es un elemento electromecánico, tiene como función autorizar o inmovilizar el paso del líquido a través de un conducto, su manejo es por medio de un electroimán el cual genera un campo electromagnético cuando recibe electricidad.

Uso en el proyecto: permitirá el paso del agua en cuanto se le de alimentación

Características

- Conectar a rosca macho G1/2"
- Voltaje de trabajo: DC12V
- Presión de agua: 0.02-0.8 Mpa
- Amplia aplicación
- Modo de funcionamiento: normalmente cerrado, abrirá cuando la alimentación este encendida

➤ Sensor UV

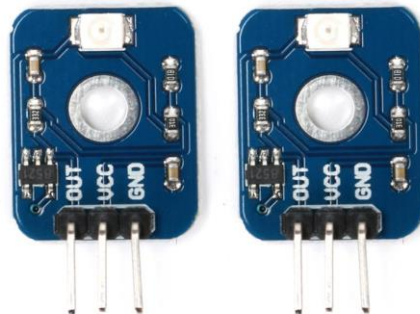


Ilustración 13 Sensor UV

Definición: adecuado para medir la cantidad total de intensidad ultravioleta de alta fiabilidad y precisión.

Uso: se utilizó para medir los rayos UV dentro de la casa de cultivo

Características

- Voltaje de trabajo: 3,3-5 V
- Voltaje de salida: DC 0-1 V
- Longitud de onda de respuesta: 200nm-370nm
- Tiempo de respuesta: menos de 0,5 segundos

➤ Sensor de humedad del suelo



Ilustración 14 Sensor de humedad del suelo

Definición: esta hecho de un material resistente, mide los niveles de humedad del suelo mediante detención capacitiva.

Uso: medir la humedad del suelo dentro de la casa de cultivo

Características

- Voltaje de funcionamiento: 3.3 – 5.5 VDC
- Voltaje de salida: 0 – 3.0 VDC
- Aplicaciones para plantas de jardín, detención de humedad, agricultura inteligente

➤ Sensor DTH 22



Ilustración 15 Sensor DTH 22 de temperatura/humedad

Definición: es un sensor digital de temperatura y humedad relativa de buen rendimiento y bajo costo. Integra un sensor capacitivo de humedad y un termistor para medir el aire circundante.

Uso en el proyecto: Los sensores nos sirvieron para saber la temperatura y humedad dentro y fuera de la casa cultivo.

Características

- Voltaje de Operación: 3V - 6V DC
- Rango de medición de temperatura: -40°C a 80 °C
- Precisión de medición de temperatura: $<\pm 0.5$ °C
- Resolución Temperatura: 0.1°C
- Rango de medición de humedad: De 0 a 100% RH
- Precisión de medición de humedad: 2% RH
- Resolución Humedad: 0.1% RH
- Tiempo de sensado: 2s
- Interface digital: Single-bus (bidireccional)
- Modelo: AM2302
- Dimensiones: 20*15*8 mm
- Peso: 3 gr.
- Carcasa de plástico blanco

➤ Redes de malla



Ilustración 16 Redes de malla

Definición: están hechas de polietileno estabilizado UV de alta densidad, ligero duradero y transpirable.

Usos: se utilizó para proteger los plantones de la luz solar directa sin necesidad de quitarlo

➤ Memoria microSD 16 GB



Ilustración 17 Memoria microSD 16 GB

Definición: es un formato para tarjetas de memoria flash para el almacenamiento de archivos digitales en dispositivos electrónicos

Uso en el proyecto: La tarjeta microSD guarda la información que detectan los sensores y para guardar las imágenes capturadas con la ESP32 CAM.

Características

- 16GB De capacidad
- Clase 10
- permite escribir grandes cantidades de datos en la tarjeta

➤ Extensión (corriente)



Ilustración 18 Extensión

Definición: Una extensión eléctrica es una longitud de cable con una clavija en un extremo y un contacto o toma en el otro y fueron creadas para proveer energía temporal en espacios donde no tenemos un tomacorriente cercano.

Uso en el proyecto: Se utilizo para dar energía directa.

Características

- Longitudes largas
- Enchufes
- Protección contra sobretensiones
- Protección infantil

➤ PCB

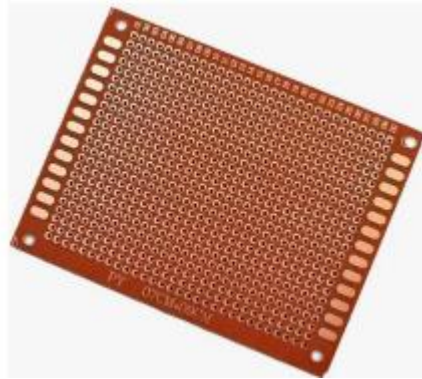


Ilustración 19 Tarjeta PCB

Definición: El significado de los PCB es Printed Circuit Board, es decir, placa de circuito impreso. Básicamente, es un soporte físico dónde instalamos posteriormente componentes electrónicos y eléctricos que se interconectan entre ellos para el desarrollo de un circuito electrónico

Uso en el proyecto: Esta placa se utilizó para hacer el circuito donde manejamos la ESP-32, el módulo de almacenamiento y el módulo Alimentación 5V para Protoboard.

Características

- constante dieléctrica
- Tamaño de 9x15cm
- conductividad térmica
- coeficiente de expansión térmica
- temperatura de servicio máxima (MOT)
- aislamiento eléctrico

➤ Cables UTP



Ilustración 20 Cables UTP

Definición: El cable UTP categoría 6 es un tipo de cable de par trenzado cuya categoría es una de la clasificación de cableado UTP.

Uso en el proyecto: Estos cables se utilizaron para conectar los sensores a el cable de red que va conectado a la ESP-32

Características: Permite montar una infraestructura de telecomunicaciones genérica dentro de un edificio, creando una red de área local (LAN), trabaja a velocidades de hasta 1000Mbps dentro de un entorno Ethernet.

➤ Cables de Red



Ilustración 21 Cables de Red

Definición: El cable eléctrico resistente al fuego y a altas temperaturas AMP 600 se caracteriza por excelente resistencia al calor, buena fuerza dieléctrica, alta resistencia a la abrasión y excelente resistencia química.

Uso en el proyecto: Estos cables se utilizaron para conectar los sensores con la ESP-32 a varios metros de distancia.

Características

- Cable de conexión de un solo núcleo sólido TUOFENG 1007 22 AWG Cable de conexión: Conductor de
- cobre estañado sólido con cubierta de PVC.

- Buena resistencia a la abrasión y resistencia a sustancias como aceites, disolventes y productos químicos:
- Aislamiento de PVC que puede soportar temperaturas de hasta 80 grados C.
- Fácil manejo, pelado y terminación: diseño de bajo deshilachado.
- Aislamiento: PVC .010". Voltaje nominal: 300 voltios, Retardante de llama: VW-1.

➤ Cable micro USB



Ilustración 22 Cable micro USB

Definición: Los conectores Micro USB se utilizan para interconectar distintos dispositivos entre sí, sean del tipo que sean siempre y cuando tengan este conector. Lo más habitual es utilizar un cable USB-A a Micro USB (cada extremo de un tipo) y así se conecta un smartphone o tablet a un ordenador.

Uso en el proyecto: se utilizó para darle corriente a la ESP-32

Características

- Conector
- Versatilidad
- Transferencia de datos:
- Longitud
- Durabilidad

➤ Adaptador de 5V



Ilustración 23 Adaptador de 5V

Definición: Un adaptador de 5V se refiere a un dispositivo que convierte la corriente eléctrica de una fuente de alimentación en una salida de 5 voltios.

Uso en el proyecto: Se utilizó para darle corriente directa con la extensión.

Características

- Fuente de Poder (Salida): 5VDC / 1A (1000 mA)
- Voltaje de Entrada: 100VAC ~ 240VAC.
- Frecuencia de Entrada: 50 Hz / 60 Hz

➤ Tubos de PVC



Ilustración 24 Tubos de PVC para agua

Definición: es un material totalmente indispensable en la fontanería y plomería. Es usado comúnmente para las tuberías de grifería de baño debido a que es estéril y completamente higiénico, logrando que la potabilidad del agua sea la máxima posible al abrir la llave del grifo. Usos en el proyecto: se utilizó para la tubería de agua hacia la casa de cultivo para así mantener el riego

➤ Cargador de 12 V y 2 A



Ilustración 25 enchufe de fuente de corriente del adaptador de corriente de CA a CC de 2 A

Este cargador se utilizó para darle corriente al Relé que alimenta la electroválvula solenoide

Características

- Entrada: 100 V CA – 240 V CA, 50Hz/60Hz
- Voltaje de salida: 12 V
- Corriente de salida: 0 – 2 A
- Potencia máxima: 240 WOperation

DESCRIPCIÓN GENERAL DEL DIAGRAMA ESQUEMATICO

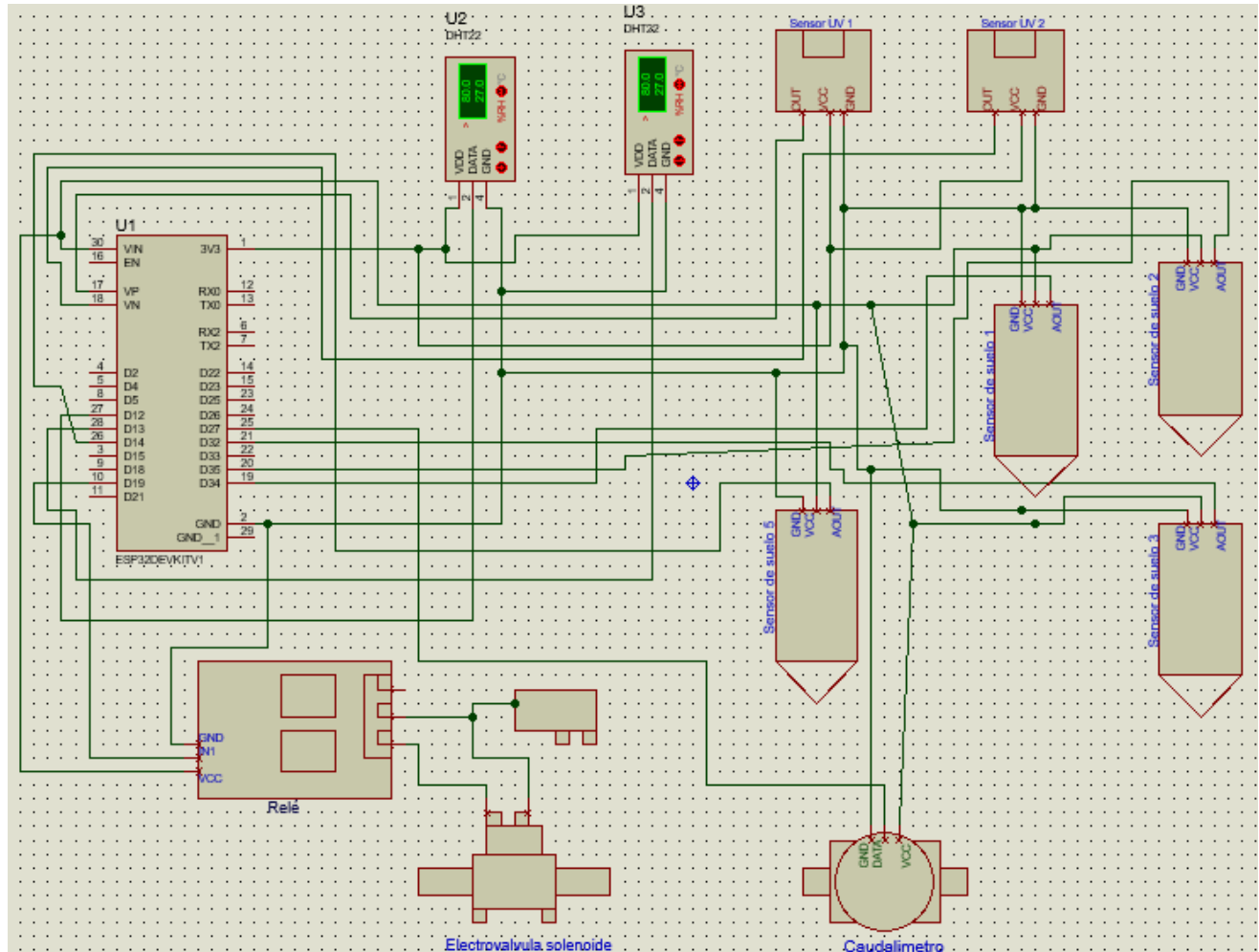


Ilustración 26 Diagrama esquemático del circuito con los componentes conectados a la ESP32

Este diagrama fue elaborado en el software de Proteus esta armado utilizando la tarjeta ESP32 esta tarjeta está conectada a su vez a una corriente directa.

DESCRIPCIÓN DE CADA PARTE DEL DIAGRAMA

- El diagrama tiene 2 sensores DHT22 (humedad / temperatura ambiental) que están conectado a los pines el primero al 12 y el segundo al 13 y ambos están conectados al voltaje de 3.3 V; conectados al GND de la tarjeta
- 2 sensores UV conectados el primero al pin 36 y el otro al pin 39 y ambos conectados al voltaje de 3.3 V; conectados al GND de la tarjeta. Nota: los pines 36 y 39 son los que en la ESP32 aparecen como VP y VN
- 4 sensores que se encargan de medir la humedad del suelo estos están conectados a los siguientes pines: sensor 1 pin 34, sensor 2 pin 35, sensor 3 pin 32, sensor 4 pin 14 estos sensores están conectados al voltaje de 5.0 V; conectados al GND de la tarjeta
- Una electroválvula solenoide que esta conecta a un voltaje de 12 V que pasa por el relé este a su vez recibe un voltaje de 5.0 V y está conectado al pin 19 de la tarjeta ESP32 al igual que el GND

- Un caudalímetro que está conectado la pin 27 y el voltaje de 5.0 V de la tarjeta al igual que el GND.

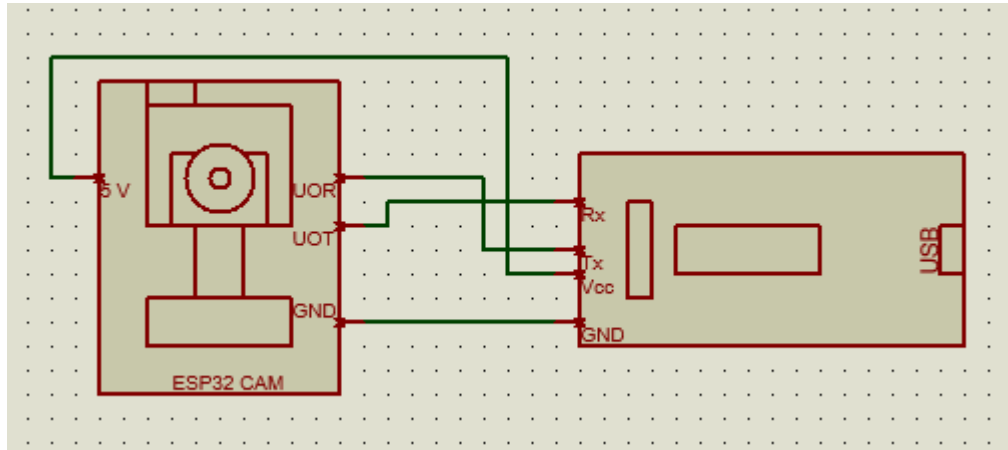


Ilustración 27 Diagrama esquemático de la ESP32 CAM

Este diagrama fue elaborado en el software de Proteus, se hizo por separado ya que su circuito es separado e incluso su programación, la tarjeta ESP32 CAM no tiene puerto para su alimentación por tanto necesita de un módulo que funcione para su conexión y alimentación los pines de esta tarjeta se conectan al módulo de la siguiente manera: GND al GND; los 5 V al Vcc; UOR al Tx; UOT al Rx. Y este módulo estará conectado a la corriente por medio de un cable USB y un adaptador.

DESCRIPCIÓN DEL CÓDIGO DE LA ESP32

Documentación del Código en MicroPython para proyecto con ESP32

Descripción General

Este código en MicroPython para ESP32 se ha desarrollado para un proyecto de IoT que supervisa diversos sensores, incluyendo sensores de humedad en el suelo, sensores UV (interior y exterior), un sensor de flujo de agua, y sensores DHT22 para medir la temperatura y humedad tanto en el interior como en el exterior. La información recolectada se envía a través de mensajes a bots de Telegram.

Configuración de Conexión Wi-Fi

El script comienza configurando la conexión Wi-Fi utilizando las credenciales proporcionadas (ssid y password). La ejecución espera a que la conexión sea exitosa antes de avanzar, asegurando una conexión estable para la transmisión de datos.

```
python
ssid = 'test'
password = 'server00001'

station = network.WLAN(network.STA_IF)
station.active(True)
```



```
station.connect(ssid, password)
while station.isconnected() == False:
    pass
print('Connection successful')
print(station.ifconfig())
```

Tokens de los Bots de Telegram

Se definen tokens para los bots de Telegram, cada uno asociado con un sensor específico. Estos tokens se utilizan más adelante para enviar mensajes a través de la API de Telegram.

```
python
TOKEN_BOT_SUELO = "6611079948:AAH81aBvL9pmFnXA8VB4spo3INBQLESIKc4"
TOKEN_BOT_UV = "6739077746:AAHxCQiN3Hy0b7vgCES5YKhO6YbgzJXnM9Q"
TOKEN_BOT_FLUJO_AGUA = "6407146772:AAFwTO-xjL225hLocDTmL28RHrkOBgqABXU"
TOKEN_BOT_DHT22 = "6941247101:AAGNo6lFBSCMwewgeHfoWMkIBlrEsf7edfY"
```

Control de Electroválvula

Se define el pin para controlar una electroválvula que regula el riego en función de la humedad del suelo. La electroválvula se abre o cierra según las condiciones del suelo.

```
python
PIN_ELECTROVALVULA = 19
electrovalvula = Pin(PIN_ELECTROVALVULA, Pin.OUT)
electrovalvula.value(0) # Inicialmente cerrada
```

Función para Enviar Mensajes a Telegram

La función `enviar_mensaje` se encarga de enviar mensajes a través de la API de Telegram. Está diseñada para ser general y reutilizable, tomando como parámetros el token del bot, el mensaje a enviar y el chat ID del grupo o usuario de destino. A continuación, se proporciona una explicación más detallada de esta función:

```
python
def enviar_mensaje(token, mensaje):
    url = "https://api.telegram.org/bot{}/sendMessage".format(token)
    payload = {"chat_id": "-1002124461151", "text": mensaje}

    try:
        response = requests.post(url, json=payload)

        # Verificar el estado de la respuesta antes de continuar
        if response.status_code == 200:
            print("Mensaje enviado exitosamente.")
        else:
            print("Error al enviar mensaje a Telegram. Código de estado:",
response.status_code)
    except Exception as e:
        print("Error al enviar mensaje a Telegram:", e)
    finally:
```

```
# Cerrar la conexión y liberar recursos
if response:
    response.close()
```

Descripción Detallada de envío a Telegram

1. Construcción de la URL y Payload:

- La URL se construye concatenando el token del bot al inicio de la URL base de la API de Telegram.
- El payload (carga útil) del mensaje se define como un diccionario que incluye el `chat_id` y el `text` del mensaje.

```
python
url = "https://api.telegram.org/bot{}/sendMessage".format(token)
payload = {"chat_id": "-1002124461151", "text": mensaje}
```

2. Envío de la Petición POST:

- Se utiliza la biblioteca `requests` para enviar una solicitud POST a la URL de la API de Telegram.
- La carga útil se envía como datos JSON en el cuerpo de la solicitud.

```
python
response = requests.post(url, json=payload)
```

3. Verificación del Estado de Respuesta:

- Se verifica si la respuesta tiene un código de estado HTTP 200, que indica que la solicitud se procesó correctamente.
- En caso de cualquier otro código de estado, se imprime un mensaje de error.

```
python
if response.status_code == 200:
    print("Mensaje enviado exitosamente.")
else:
    print("Error al enviar mensaje a Telegram. Código de estado:",
response.status_code)
```

4. Manejo de Excepciones:

- Se implementa un bloque `try-except` para manejar excepciones que puedan ocurrir durante el envío del mensaje.
- Si hay un error, se imprime el mensaje de error.

```
python
except Exception as e:
    print("Error al enviar mensaje a Telegram:", e)
```

5. Cierre de la Conexión y Liberación de Recursos:

- En el bloque `finally`, se cierra la conexión para liberar recursos, asegurándose de que se realice incluso si hay un error.

```
python
finally:
    if response:
```

```
response.close()
```

Esta función proporciona un mecanismo robusto para enviar mensajes a Telegram, con manejo de errores y liberación adecuada de recursos. Puedes usar esta función en diferentes partes del código para enviar actualizaciones y alertas a los grupos de Telegram asociados con cada sensor.

Sensores de Humedad en el Suelo

Se definen pines ADC para los sensores de humedad en el suelo y se ajustan para una sensibilidad específica. Se realiza la lectura de ADC para calcular el porcentaje de humedad en cada sensor. La función también evalúa el estado general del suelo y controla la electroválvula en consecuencia.

```
python
# Pines ADC para los sensores de suelo
AOUT_PINS = [34, 35, 39, 32]
adcs = [ADC(Pin(pin)) for pin in AOUT_PINS]
for adc in adcs:
    adc.atten(ADC.ATTN_11DB)
```

La función `medir_sensores_humedad_suelo` recorre los sensores, calcula el porcentaje de humedad y determina el estado del suelo. También controla la electroválvula según las condiciones del suelo.

```
python
def medir_sensores_humedad_suelo():
    # ...
```

Sensor de Flujo de Agua

Se configura un pin para medir la frecuencia de pulsos generados por el sensor de flujo de agua. La frecuencia se utiliza para calcular el flujo y la cantidad total de litros. La función `medir_flujo_agua` envía los datos a un bot de Telegram específico.

```
python
# Configuración de la interrupción para el flanco de subida (rising edge)
sf.irq(trigger=Pin.IRQ_RISING, handler=conteo)

def medir_flujo_agua():
    # ...
```

Sensores UV Interior y Exterior

Dos funciones (`uv_i` y `uv_e`) miden la radiación UV interior y exterior respectivamente. Los resultados se clasifican en índices de UV y niveles de riesgo, y se envían a bots de Telegram específicos.

```
python
def uv_i():
    # ...

def uv_e():
    # ...
```

Sensores DHT22 (Temperatura y Humedad)

La función `dht22` mide la temperatura y humedad interior y exterior utilizando sensores DHT22. Los datos se envían a un bot de Telegram específico.

```
python
def dht22():
    # ...
```

Bucle Principal

En el bucle principal, se llaman secuencialmente todas las funciones que obtienen datos de los sensores. Después de cada iteración, el ESP32 espera 15 minutos antes de comenzar una nueva medición.

```
python
while True:
    uv_i()
    uv_e()
    dht22()
    medir_flujo_agua()
    medir_sensores_humedad_suelo()
    sleep(60*15)
```

Este documento proporciona una visión detallada de cada sección del código, destacando la funcionalidad de cada parte del script. Asegúrate de ajustar los tiempos de espera y frecuencias de medición según las necesidades específicas de tu proyecto.

DESCRIPCIÓN DEL CÓDIGO DE LA ESP32 CAM

Código de la Esp32-cam

```
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include "esp_camera.h"
#include "FS.h"
#include "SD_MMC.h"           // SD Card ESP32
#include "driver/rtc_io.h"
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

const char* ssid = "test";
const char* password = "server00001";

// Inicializa el BOT Telegram
String BOTtoken = "6790188639:AAHy-MWFQX2T4P_4lRiYgeydfhIYXdVtznU"; // Token del
bot
```

```
String CHAT_ID = "-1002124461151";//ID del chat o grupo de telegram

bool sendPhoto = false;

WiFiClientSecure clientTCP;
UniversalTelegramBot bot(BOTtoken, clientTCP);//Inicia comunicacion con el bot

#define FLASH_LED_PIN 4
bool flashState = LOW;

//revisa si hay mensaje cada 1 segundo
int botRequestDelay = 1000;
unsigned long lastTimeBotRan;

//Datos de la pines de la camara
#define PWDN_GPIO_NUM    32
#define RESET_GPIO_NUM  -1
#define XCLK_GPIO_NUM    0
#define SIOD_GPIO_NUM    26
#define SIOC_GPIO_NUM    27

#define Y9_GPIO_NUM      35
#define Y8_GPIO_NUM      34
#define Y7_GPIO_NUM      39
#define Y6_GPIO_NUM      36
#define Y5_GPIO_NUM      21
#define Y4_GPIO_NUM      19
#define Y3_GPIO_NUM      18
#define Y2_GPIO_NUM       5
#define VSYNC_GPIO_NUM   25
#define HREF_GPIO_NUM    23
#define PCLK_GPIO_NUM    22

unsigned int pictureNumber = 0;//cantidad de fotos el numero subira a medida que se
toman las fotos

void configInitCamera(){//funcion de inicio de la camara
    camera_config_t config;
    config.ledc_channel = LEDC_CHANNEL_0;
    config.ledc_timer = LEDC_TIMER_0;
    config.pin_d0 = Y2_GPIO_NUM;
    config.pin_d1 = Y3_GPIO_NUM;
    config.pin_d2 = Y4_GPIO_NUM;
    config.pin_d3 = Y5_GPIO_NUM;
    config.pin_d4 = Y6_GPIO_NUM;
    config.pin_d5 = Y7_GPIO_NUM;
```

```
config.pin_d6 = Y8_GPIO_NUM;
config.pin_d7 = Y9_GPIO_NUM;
config.pin_xclk = XCLK_GPIO_NUM;
config.pin_pclk = PCLK_GPIO_NUM;
config.pin_vsync = VSYNC_GPIO_NUM;
config.pin_href = HREF_GPIO_NUM;
config.pin_sscb_sda = SIOD_GPIO_NUM;
config.pin_sscb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM;
config.pin_reset = RESET_GPIO_NUM;
config.xclk_freq_hz = 20000000;
config.pixel_format = PIXFORMAT_JPEG;

if(psramFound()){
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10; //0-63 numeros más bajo significa mayor calidad
    config.fb_count = 2;
} else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12; //0-63 numeros más bajo significa mayor calidad
    config.fb_count = 1;
}

// inicia la camara
esp_err_t err = esp_camera_init(&config);
if (err != ESP_OK) {
    Serial.printf("Error al iniciar la camara 0x%x", err);
    delay(1000);
    ESP.restart();
}

// Tamaño de fotograma desplegable para una mayor velocidad de fotogramas inicial
sensor_t * s = esp_camera_sensor_get();
s->set_framesize(s, FRAMESIZE_CIF); //
UXGA|SXGA|XGA|SVGA|VGA|CIF|QVGA|HQVGA|QQVGA
}

void handleNewMessages(int numNewMessages) {
    Serial.print("Handle New Messages: ");
    Serial.println(numNewMessages);

    for (int i = 0; i < numNewMessages; i++) {
        String chat_id = String(bot.messages[i].chat_id);
        if (chat_id != CHAT_ID){
            bot.sendMessage(chat_id, "Unauthorized user", "");
            continue;
        }
    }
}
```

```
}

// imprime mensaje de en el chat de telegram
String text = bot.messages[i].text;
Serial.println(text);

String from_name = bot.messages[i].from_name;
if (text == "/iniciar@Cultivo32cam_bot") { //comando de inicio del bot
    String welcome = "Bienvenido , " + from_name + "\n";
    welcome += "Para interactuar con la Esp32-cam puedes utilizar los siguientes comandos: \n";
    welcome += "/foto : Toma una foto\n";
    bot.sendMessage(CHAT_ID, welcome, "");
}

if (text == "/foto@Cultivo32cam_bot") { // comando de toma de fotos
    String wel = "Hola, " + from_name + "\n";
    wel += "En unos segundos se enviara la foto...\n";
    bot.sendMessage(CHAT_ID, wel, "");
    sendPhoto = true;
    Serial.println("Nueva solicitud de foto");
}
}
}

String sendPhotoTelegram() { // conexion al chat de telegram y envio
    const char* myDomain = "https://api.telegram.org";
    String getAll = "";
    String getBody = "";

    camera_fb_t * fb = NULL;
    fb = esp_camera_fb_get();
    if(!fb) {
        Serial.println("Camera capture failed");
        delay(1000);
        ESP.restart();
        return "Camera capture failed";
    }

    Serial.println("Connect to " + String(myDomain));

    if (clientTCP.connect(myDomain, 443)) {
        Serial.println("Connection successful");
    }
}
```

```

String head = "--electronicclinic\r\nContent-Disposition: form-data;
name=\"chat_id\"; \r\n\r\n" + CHAT_ID + "\r\n--electronicclinic\r\nContent-
Disposition: form-data; name=\"photo\"; filename=\"esp32-cam.jpg\"\r\nContent-Type:
image/jpeg\r\n\r\n";
String tail = "\r\n--electronicclinic--\r\n";

uint16_t imageLen = fb->len;
uint16_t extraLen = head.length() + tail.length();
uint16_t totalLen = imageLen + extraLen;

clientTCP.println("POST /bot"+BOTtoken+"/sendPhoto HTTP/1.1");
clientTCP.println("Host: " + String(myDomain));
clientTCP.println("Content-Length: " + String(totalLen));
clientTCP.println("Content-Type: multipart/form-data;
boundary=electronicclinic");
clientTCP.println();
clientTCP.print(head);

uint8_t *fbBuf = fb->buf;
size_t fbLen = fb->len;
for (size_t n=0;n<fbLen;n=n+1024) {
    if (n+1024<fbLen) {
        clientTCP.write(fbBuf, 1024);
        fbBuf += 1024;
    }
    else if (fbLen%1024>0) {
        size_t remainder = fbLen%1024;
        clientTCP.write(fbBuf, remainder);
    }
}

clientTCP.print(tail);

esp_camera_fb_return(fb);

int waitTime = 10000; // timeout 10 seconds
long startTimer = millis();
boolean state = false;

while ((startTimer + waitTime) > millis()){
    Serial.print(".");
    delay(100);
    while (clientTCP.available()) {
        char c = clientTCP.read();
        if (state==true) getBody += String(c);
        if (c == '\n') {

```



```
        if (getAll.length()==0) state=true;
        getAll = "";
    }
    else if (c != '\r')
        getAll += String(c);
        startTimer = millis();
    }
    if (getBody.length()>0) break;
}
clientTCP.stop();
Serial.println(getBody);

}
else {
    getBody="Conexion a api.telegram.org fallida.";
    Serial.println("conexion a api.telegram.org fallida.");
}
return getBody;
}

void setup(){
    WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
    // Inicializador del monitor serial
    Serial.begin(115200);

    // Establecer flash LED como salida
    pinMode(FLASH_LED_PIN, OUTPUT);
    digitalWrite(FLASH_LED_PIN, flashState);
    // Configurar e iniciar la cámara
    Serial.print("Iniciando camara...");
    configInitCamera();
    Serial.println("Ok!");
    // Inicializar MicroSD
    Serial.print("Initializing the MicroSD card module... ");
    initMicroSDCard();
    // Conectar al WIFI
    WiFi.mode(WIFI_STA);
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, password);
    clientTCP.setCACert(TELEGRAM_CERTIFICATE_ROOT); // Agregar certificado raíz para
    api.telegram.org
    while (WiFi.status() != WL_CONNECTED) {
        Serial.print(".");
```

```
    delay(500);
}
Serial.println();
Serial.print("ESP32-CAM IP Address: ");
Serial.println(WiFi.localIP());
}

void loop() {
    tomarecorrente();
    if (sendPhoto) {
        Serial.println("Preparing photo");
        sendPhotoTelegram();
        sendPhoto = false;
    }
    if (millis() > lastTimeBotRan + botRequestDelay) {
        int numNewMessages = bot.getUpdates(bot.last_message_received + 1);
        while (numNewMessages) {
            Serial.println("got response");
            handleNewMessages(numNewMessages);
            numNewMessages = bot.getUpdates(bot.last_message_received + 1);
        }
        lastTimeBotRan = millis();
    }
}

void initMicroSDCard(){
    // iniciar MicroSD
    Serial.println("Starting SD Card");
    if(!SD_MMC.begin()){
        Serial.println("SD Card Mount Failed");
        return;
    }
    uint8_t cardType = SD_MMC.cardType();
    if(cardType == CARD_NONE){
        Serial.println("No SD Card attached");
        return;
    }
}

void tomarecorrente() {

    // Esperar hasta que haya pasado 1 minuto
    static unsigned long lastMillis = millis();
    unsigned long currentMillis = millis();
```

```
// Comparar el tiempo transcurrido desde la última ejecución
if (currentMillis - lastMillis >= 1000 * 60*30) {
// Path where new picture will be saved in SD Card
String path = "/picture" + String pictureNumber + ".jpg";
Serial.printf("Picture file name: %s\n", path.c_str());
// Tu código que se ejecutará cada minuto
takeSavePhoto(path);
pictureNumber++;
// Actualizar el tiempo de referencia
lastMillis = currentMillis;
}
}

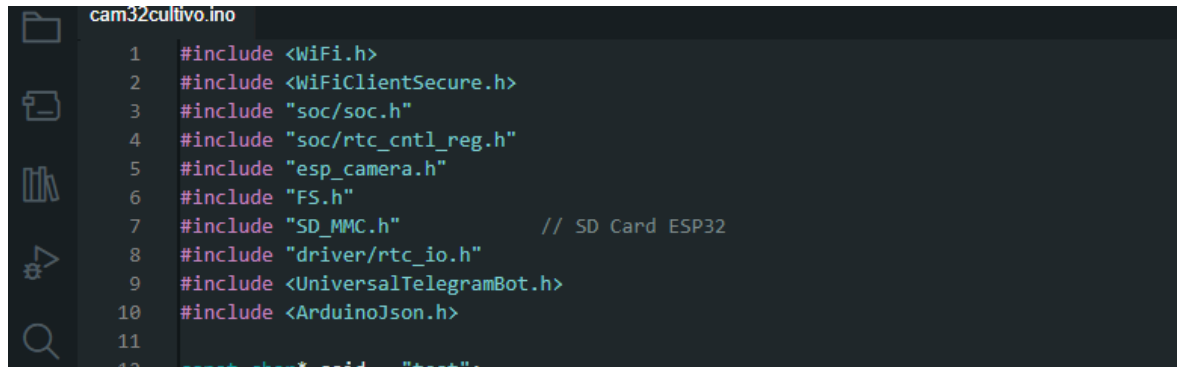
void takeSavePhoto(String path){
// Tomar foto
camera_fb_t * fb = esp_camera_fb_get();
if(!fb) {
Serial.println("Camera capture failed");
return;
}

// guardar foto en la SD
fs::FS &fs = SD_MMC;
File file = fs.open(path.c_str(), FILE_WRITE);
if(!file){
Serial.println("Failed to open file in writing mode");
}
else {
file.write(fb->buf, fb->len);
Serial.printf("Saved file to path: %s\n", path.c_str());
}
file.close();

//devolver el frame buffer al controlador para su reutilización
esp_camera_fb_return(fb);
}
```

Descripción de cada parte del código de la ESP32 CAM

Librerías:



```
cam32cultivo.ino
1  #include <WiFi.h>
2  #include <WiFiClientSecure.h>
3  #include "soc/soc.h"
4  #include "soc/rtc_cntl_reg.h"
5  #include "esp_camera.h"
6  #include "FS.h"
7  #include "SD_MMC.h"           // SD Card ESP32
8  #include "driver/rtc_io.h"
9  #include <UniversalTelegramBot.h>
10 #include <ArduinoJson.h>
11
12 const char* ssid = "test";
```

WiFi y WiFiClientSecure:

Proporcionan funcionalidades para la conexión y configuración de redes Wi-Fi.

WiFiClientSecure es una extensión segura de WiFiClient para conexiones seguras (TLS/SSL).

esp_camera:

Biblioteca específica para el manejo de la cámara ESP32-CAM.

Incluye funciones para la configuración y operación de la cámara.

FS y SD_MMC:

FS es una biblioteca para trabajar con sistemas de archivos en ESP32.

SD_MMC es una implementación específica para tarjetas SD y MMC.

rtc_io y soc:

Proporcionan funciones para el control del reloj en tiempo real y de la parte sistema en ESP32.

UniversalTelegramBot:

Facilita la comunicación con la API de Telegram para bots.

Permite enviar mensajes y archivos, y recibir actualizaciones del bot de Telegram.

ArduinoJson:

Ofrece funciones para analizar y generar datos JSON en Arduino.

Útil para trabajar con la estructura de mensajes JSON en las respuestas de Telegram.

Credenciales del WIFI y los token e ID de Telegram

```
11
12 const char* ssid = "test";
13 const char* password = "server00001";
14
15 // Inicializa el BOT Telegram
16 String BOTtoken = "6790188639:AAHy-MWFQX2T4P_4lRiYgeydfhIYXdVtznU"; // Token del bot
17
18 String CHAT_ID = "-1002124461151"; // ID del chat o grupo de telegram
19
```

- ssid y password son las credenciales de la red Wi-Fi.
- BOTtoken es el token que se obtuvo al crear el bot en BotFather.
- CHAT_ID es el ID de chat, que se puede obtener usando el bot @myidbot para descubrir el ID asociado con tu cuenta de Telegram.

configInitCamera:

```
53 void configInitCamera() //funcion de inicio de la camara
54     camera_config_t config;
55     config.ledc_channel = LEDC_CHANNEL_0;
56     config.ledc_timer = LEDC_TIMER_0;
57     config.pin_d0 = Y2_GPIO_NUM;
58     config.pin_d1 = Y3_GPIO_NUM;
59     config.pin_d2 = Y4_GPIO_NUM;
60     config.pin_d3 = Y5_GPIO_NUM;
61     config.pin_d4 = Y6_GPIO_NUM;
62     config.pin_d5 = Y7_GPIO_NUM;
63     config.pin_d6 = Y8_GPIO_NUM;
64     config.pin_d7 = Y9_GPIO_NUM;
65     config.pin_xclk = XCLK_GPIO_NUM;
66     config.pin_pclk = PCLK_GPIO_NUM;
67     config.pin_vsync = VSYNC_GPIO_NUM;
68     config.pin_href = HREF_GPIO_NUM;
69     config.pin_sscb_sda = SIOD_GPIO_NUM;
70     config.pin_sscb_scl = SIOC_GPIO_NUM;
71     config.pin_pwdn = PWDN_GPIO_NUM;
72     config.pin_reset = RESET_GPIO_NUM;
73     config.xclk_freq_hz = 20000000;
74     config.pixel_format = PIXFORMAT_JPEG;
75
```

- Esta función inicializa la configuración de la cámara ESP32-CAM.

- Establece los pines GPIO y otras configuraciones relacionadas con la cámara.
- Configura la resolución de la imagen, calidad JPEG y otros parámetros.

handleNewMessages:

```
99 void handleNewMessages(int numNewMessages) {
100     Serial.print("Handle New Messages: ");
101     Serial.println(numNewMessages);
102
103     for (int i = 0; i < numNewMessages; i++) {
104         String chat_id = String(bot.messages[i].chat_id);
105         if (chat_id != CHAT_ID){
106             bot.sendMessage(chat_id, "Unauthorized user", "");
107             continue;
108         }
109
110         // imprime mensaje de en el chat de telegram
111         String text = bot.messages[i].text;
112         Serial.println(text);
113
114         String from_name = bot.messages[i].from_name;
115         if (text == "/iniciar@Cultivo32cam_bot") { //comando de inicio del bot
116             String welcome = "Bienvenido , " + from_name + "\n";
117             welcome += "Para interactuar con la Esp32-cam puedes utilizar los siguientes comandos: \n";
118             welcome += "/foto : Toma una foto\n";
119             bot.sendMessage(CHAT_ID, welcome, "");
120         }
121
122         if (text == "/foto@Cultivo32cam_bot") { // comando de toma de fotos
123             String wel = "Hola, " + from_name + "\n";
124             wel += "En unos segundos se enviara la foto...\n";
125             bot.sendMessage(CHAT_ID, wel, "");
126             sendPhoto = true;
127             Serial.println("Nueva solicitud de foto");
128         }
129     }
130 }
```

- Esta función maneja los mensajes entrantes desde Telegram.
- Verifica el remitente del mensaje y responde en consecuencia.
- Muestra un mensaje de bienvenida y proporciona información sobre los comandos disponibles.

sendPhotoTelegram:

```
131
132 ✓ String sendPhotoTelegram() { // conexion al chat de telegram y envio
133     const char* myDomain = "https://api.telegram.org";
134     String getAll = "";
135     String getBody = "";
136
137     camera_fb_t * fb = NULL;
138     fb = esp_camera_fb_get();
139     if(!fb) {
140         Serial.println("Camera capture failed");
141         delay(1000);
142         ESP.restart();
143         return "Camera capture failed";
144     }
145
146     Serial.println("Connect to " + String(myDomain));
147
148
149 ✓ if (clientTCP.connect(myDomain, 443)) {
150     Serial.println("Connection successful");
151
152     String head = "--electronicclinic\r\nContent-Disposition: form-data; name=\"chat_id\"; \r\n\r\n" +
153     String tail = "\r\n--electronicclinic--\r\n";
154
155     uint16_t imageLen = fb->len;
156     uint16_t extraLen = head.length() + tail.length();
157     uint16_t totallen = imageLen + extraLen;
```

- Envía una foto capturada al bot de Telegram.
- Se conecta al servidor de Telegram mediante el cliente TCP seguro.
- Utiliza el método POST para enviar la foto como un archivo multipart/form-data.
- Espera la respuesta del servidor de Telegram y muestra información de depuración.

setup:

```
215 void setup() {
216     WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
217     // Iniciador del monitor serial
218     Serial.begin(115200);
219
220     // Establecer flash LED como salida
221     pinMode(FLASH_LED_PIN, OUTPUT);
222     digitalWrite(FLASH_LED_PIN, flashState);
223     // Configurar e iniciar la cámara
224     Serial.print("Iniciando camara...");
225     configInitCamera();
226     Serial.println("Ok!");
227     // Inicializar MicroSD
228     Serial.print("Initializing the MicroSD card module... ");
229     initMicroSDCard();
230     // Conectar al WIFI
231     WiFi.mode(WIFI_STA);
232     Serial.println();
233     Serial.print("Connecting to ");
234     Serial.println(ssid);
235     WiFi.begin(ssid, password);
236     clientTCP.setCACert(TELEGRAM_CERTIFICATE_ROOT); // Agregar certificado raíz para api.telegram.org
237     while (WiFi.status() != WL_CONNECTED) {
238         Serial.print(".");
239         delay(500);
240     }
241     Serial.println();
242     Serial.print("ESP32-CAM IP Address: ");
243     Serial.println(WiFi.localIP());
```

- Inicializa el entorno del programa al comenzar.
- Desactiva la protección contra apagones de la ESP32.
- Inicializa la comunicación serie para la depuración.
- Configura el pin del LED de flash y la cámara.
- Inicializa la tarjeta MicroSD y establece la conexión Wi-Fi.

loop:

```
246 void loop() {
247     tomarecorrente();
248     if (sendPhoto) {
249         Serial.println("Preparing photo");
250         sendPhotoTelegram();
251         sendPhoto = false;
252     }
253     if (millis() > lastTimeBotRan + botRequestDelay) {
254         int numNewMessages = bot.getUpdates(bot.last_message_received + 1);
255         while (numNewMessages) {
256             Serial.println("got response");
257             handleNewMessages(numNewMessages);
258             numNewMessages = bot.getUpdates(bot.last_message_received + 1);
259         }
260         lastTimeBotRan = millis();
261     }
262 }
263 }
```

- Es la función principal que se ejecuta en un bucle continuo.
- Llama a tomarecorrente para tomar fotos periódicamente.
- Maneja nuevos mensajes de Telegram llamando a handleNewMessages.
- Mantiene actualizado el bot de Telegram llamando a bot.getUpdates.

initMicroSDCard:

```
265 void initMicroSDCard(){
266     // iniciar MicroSD
267     Serial.println("Starting SD Card");
268     if(!SD_MMC.begin()){
269         Serial.println("SD Card Mount Failed");
270         return;
271     }
272     uint8_t cardType = SD_MMC.cardType();
273     if(cardType == CARD_NONE){
274         Serial.println("No SD Card attached");
275         return;
276     }
277 }
```

- Inicializa la tarjeta MicroSD.
- Comprueba si la tarjeta está presente y lista para su uso.

tomarecorrente:

```
279 void tomarecorrente() {
280
281     // Esperar hasta que haya pasado 1 minuto
282     static unsigned long lastMillis = millis();
283     unsigned long currentMillis = millis();
284
285     // Comparar el tiempo transcurrido desde la última ejecución
286     if (currentMillis - lastMillis >= 1000 * 60*30) {
287         // Path where new picture will be saved in SD Card
288         String path = "/picture" + String(pictureNumber) + ".jpg";
289         Serial.printf("Picture file name: %s\n", path.c_str());
290         // Tu código que se ejecutará cada minuto
291         takeSavePhoto(path);
292         pictureNumber++;
293         // Actualizar el tiempo de referencia
294         lastMillis = currentMillis;
295     }
296 }
```

- Programa la captura de fotos a intervalos regulares.
- Utiliza un temporizador para determinar cuándo debe tomar la siguiente foto.
- Llama a takeSavePhoto para capturar y guardar la foto.

takeSavePhoto:

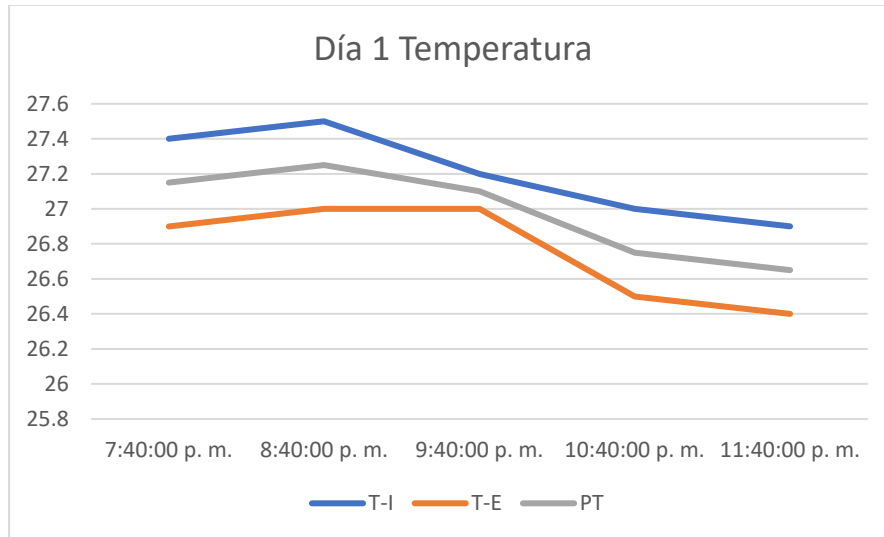
```
296 }
297 void takeSavePhoto(String path){
298     // Tomar foto
299     camera_fb_t * fb = esp_camera_fb_get();
300     if(!fb) {
301         Serial.println("Camera capture failed");
302         return;
303     }
304
305     // guardar foto en la SD
306     fs::FS &fs = SD_MMC;
307     File file = fs.open(path.c_str(), FILE_WRITE);
308     if(!file){
309         Serial.println("Failed to open file in writing mode");
310     }
311     else {
312         file.write(fb->buf, fb->len);
313         Serial.printf("Saved file to path: %s\n", path.c_str());
314     }
315     file.close();
316
317     //devolver el frame buffer al controlador para su reutilización
318     esp_camera_fb_return(fb);
319 }
```

- Captura una foto con la cámara y la guarda en la tarjeta MicroSD.
- Utiliza las funciones de la biblioteca de la cámara y el sistema de archivos para realizar estas operaciones.

DESCRIPCIÓN DE LAS GRÁFICAS POR DÍA

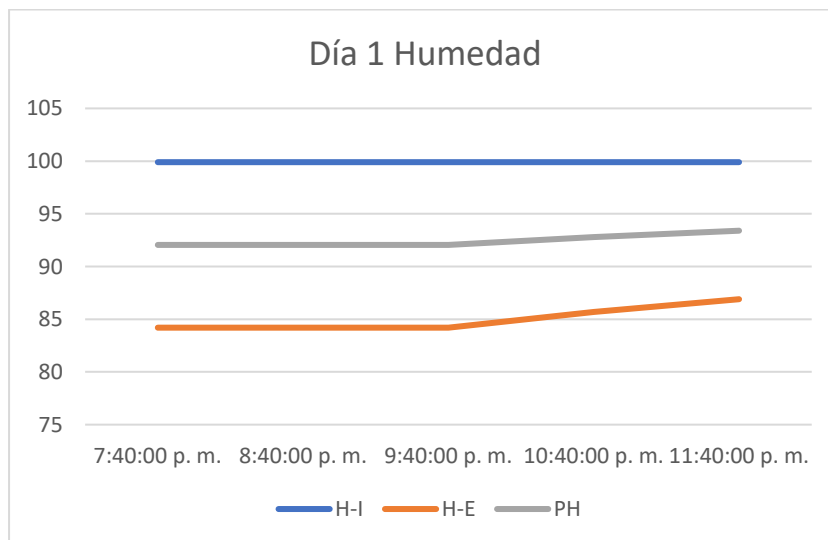
Gráficas día 1

➤ Gráfica de temperatura con sensores DTH22



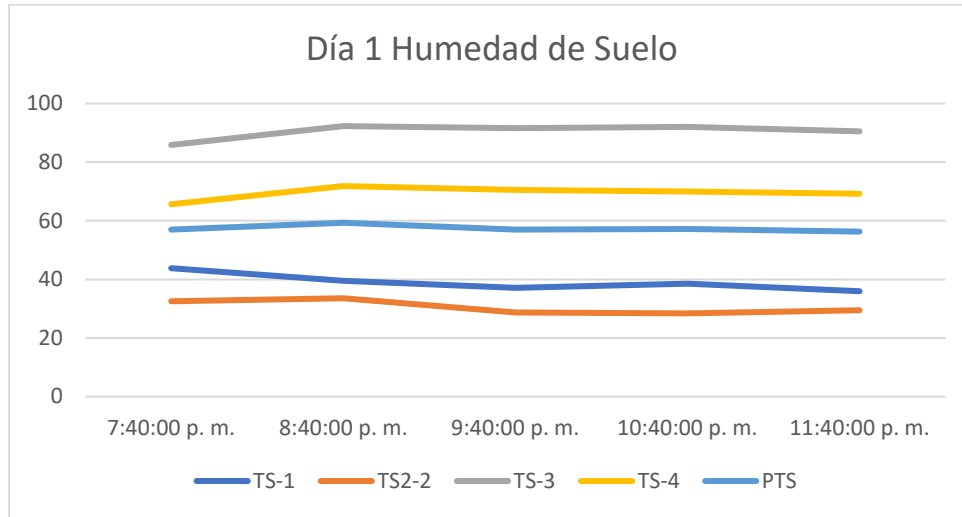
Descripción: Se muestran los datos de la temperatura en la casa de cultivo durante el día 1, tanto de la temperatura interior (color azul), exterior (color naranja) y el promedio de las temperatura.

➤ Gráfica de humedad con sensores DTH22



Descripción: Se muestran los datos de humedad en la casa de cultivo durante el día 1, tanto de la humedad interior (color azul), exterior (color naranja) y el promedio de la humedad (color gris).

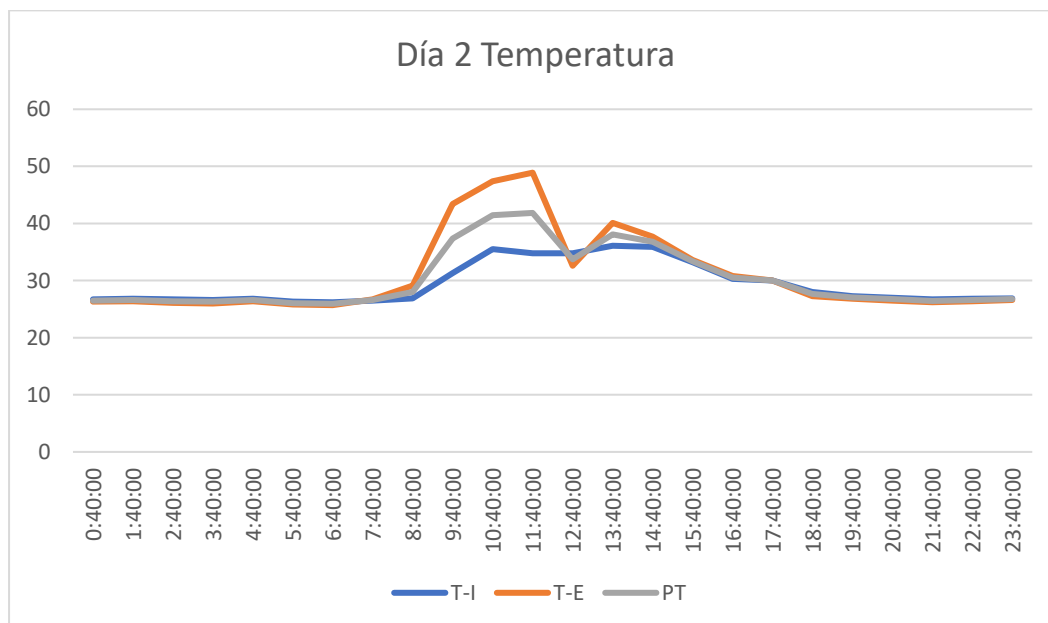
➤ Gráfica con sensores de humedad de suelo



Descripción: Se muestran los datos de humedad del suelo en la casa de cultivo durante el día 1, tanto de la humedad de las 4 surcos de cultivos sembrados. Primera línea (color azul), segunda línea (color naranja), tercera línea (color amarillo), cuarta línea (color celeste) y el promedio de la humedad (color gris).

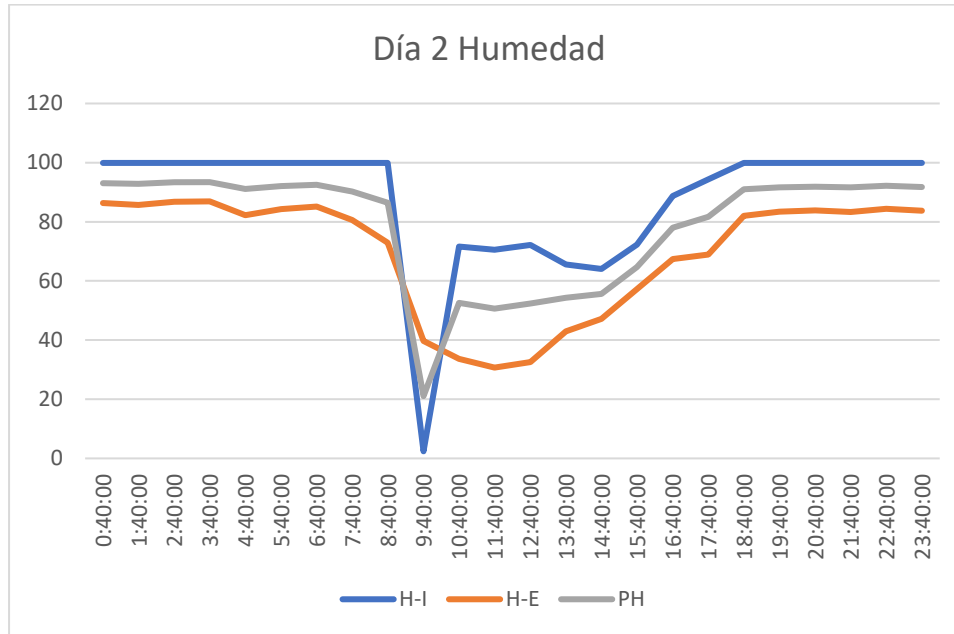
Gráficas día 2

➤ Gráfica de temperatura con sensores DTH22



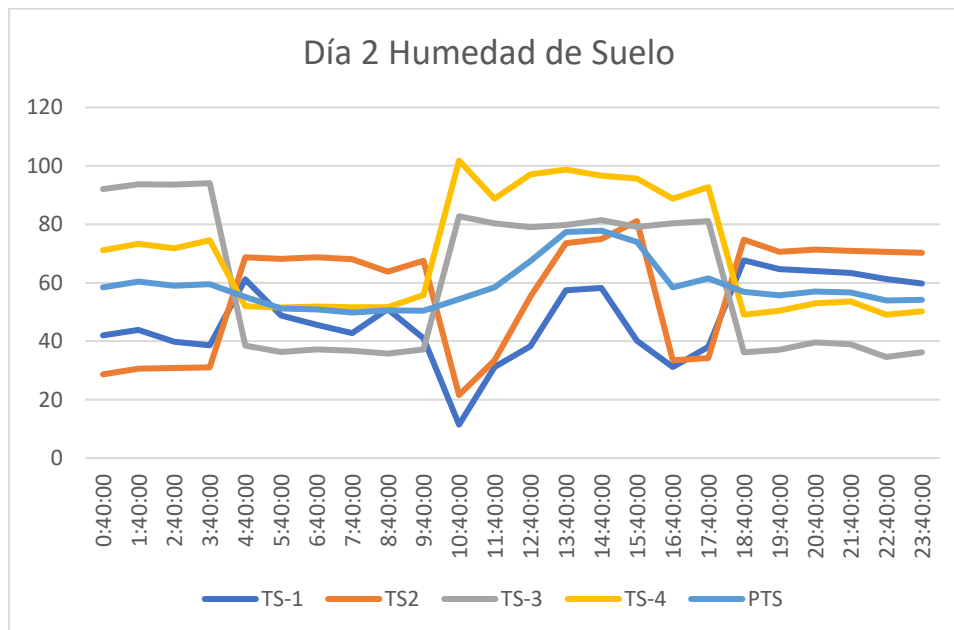
Descripción: Se muestra información detallada sobre las temperaturas registradas en una casa de cultivo durante el día 2. Los datos incluyen la temperatura tanto en el interior (color azul) como en el exterior (color naranja) de la casa de cultivo. Además, se proporciona el promedio de estas temperaturas (color gris).

➤ Gráfica de humedad con sensores DTH22



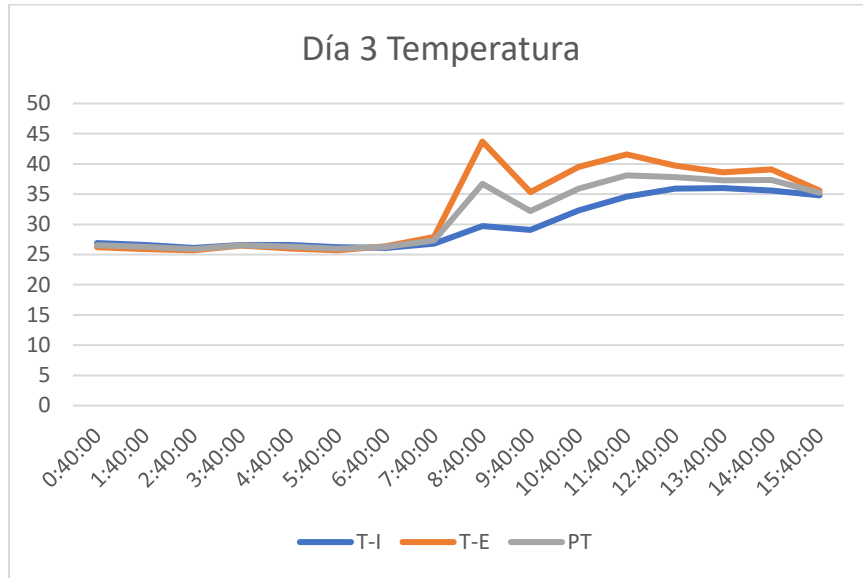
Descripción: Se muestran los datos de humedad en la casa de cultivo durante el día 2, en donde se interpretan los datos tanto de la humedad interior (color azul), exterior (color naranja) y el promedio de la humedad (color gris).

➤ Gráfica con sensores de humedad del suelo



Descripción: Se muestran los datos de humedad del suelo en la casa de cultivo durante el día 2, tanto de la humedad de las 4 surcos de cultivos sembrados. Primera línea (color azul), segunda línea (color naranja), tercera línea (color amarillo), cuarta línea (color celeste) y el promedio de la humedad (color gris).

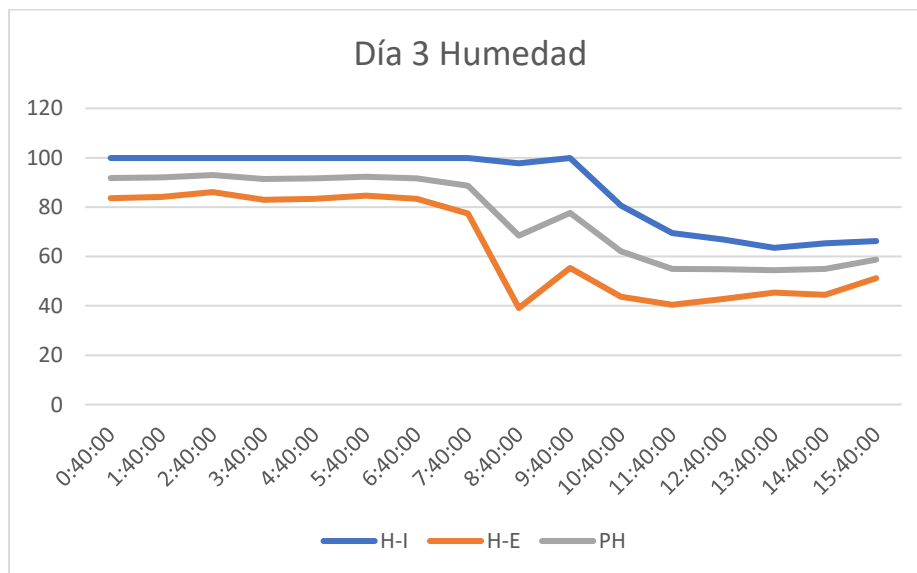
➤ Gráfica de temperatura con DTH22



Descripción: Se muestra información detallada sobre las temperaturas registradas en una casa de cultivo durante el día 3. Los datos incluyen la temperatura tanto en el interior (color azul) como en el exterior (color naranja) de la casa de cultivo. Además, se proporciona el promedio de estas temperaturas (color gris).

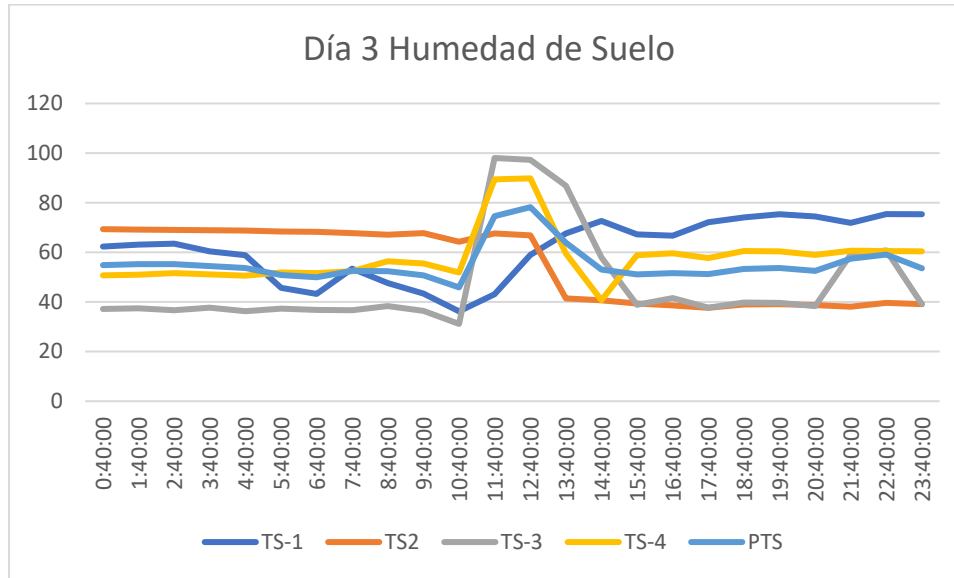
Gráficas día 3

➤ Gráfica de humedad con DTH22



Descripción: Se muestran los datos de humedad en la casa de cultivo durante el día 3, en donde se interpretan los datos tanto de la humedad interior (color azul), exterior (color naranja) y el promedio de la humedad (color gris).

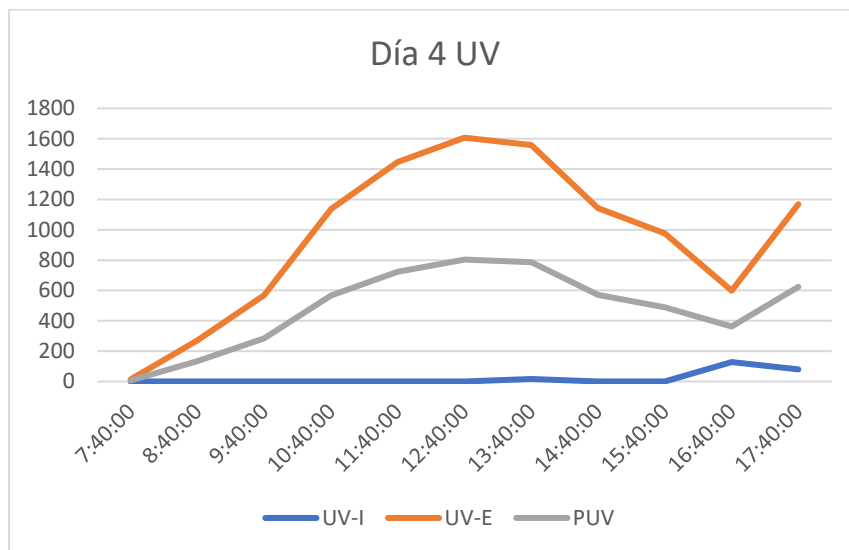
➤ Gráfica con sensores de humedad del suelo



Descripción: Se muestran los datos de humedad del suelo en la casa de cultivo durante el día 2, tanto de la humedad de las 4 surcos de cultivos sembrados. Primera línea (color azul), segunda línea (color naranja), tercera línea (color amarillo), cuarta línea (color celeste) y el promedio de la humedad (color gris).

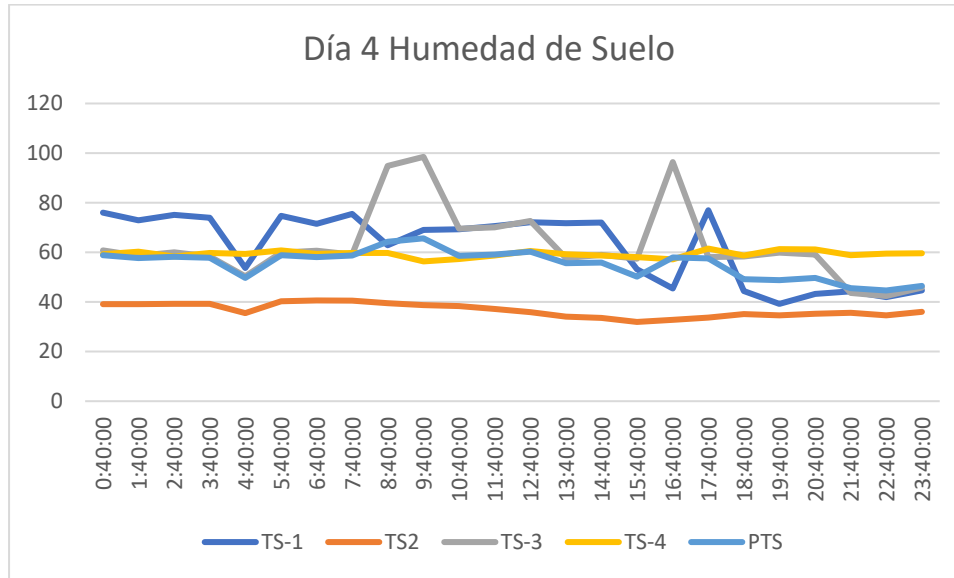
Gráficas día 4

➤ Grafica con sensores UV



Descripción: Se muestra información detallada sobre el sensor de ultravioleta, registradas en una casa de cultivo durante el día 4. Los datos incluyen el sensor ultravioleta tanto en el interior (color azul) como en el exterior (color naranja) de la casa de cultivo. Además, se proporciona el promedio de estas temperaturas del sensor UV (color gris).

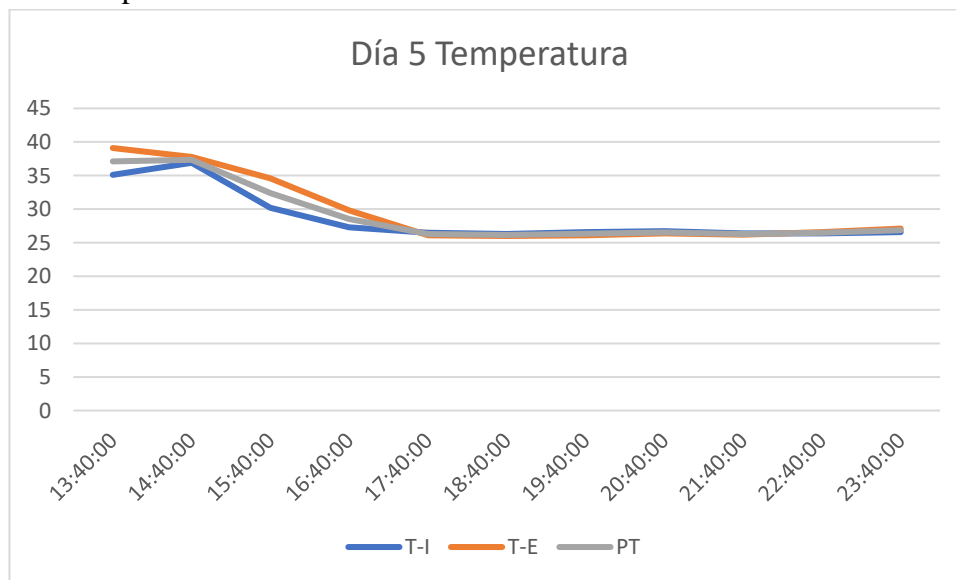
➤ Gráfica con sensores de humedad de suelo



Descripción: Se muestran los datos de humedad del suelo en la casa de cultivo durante el día 4, tanto de la humedad de las 4 surcos de cultivos sembrados. Primera línea (color azul), segunda línea (color naranja), tercera línea (color amarillo), cuarta línea (color celeste) y el promedio de la humedad (color gris).

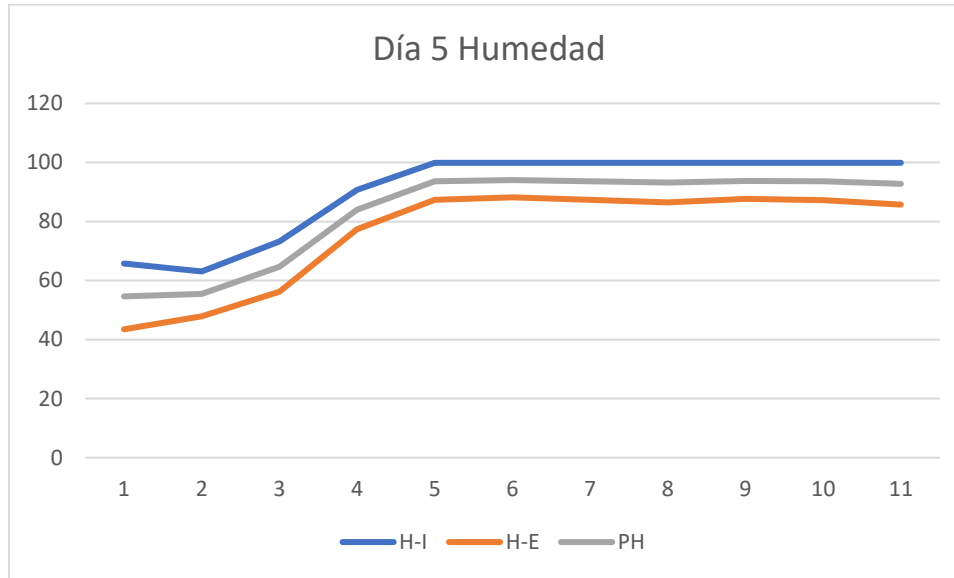
Gráficas día 5

➤ Gráfica de temperatura con DTH22



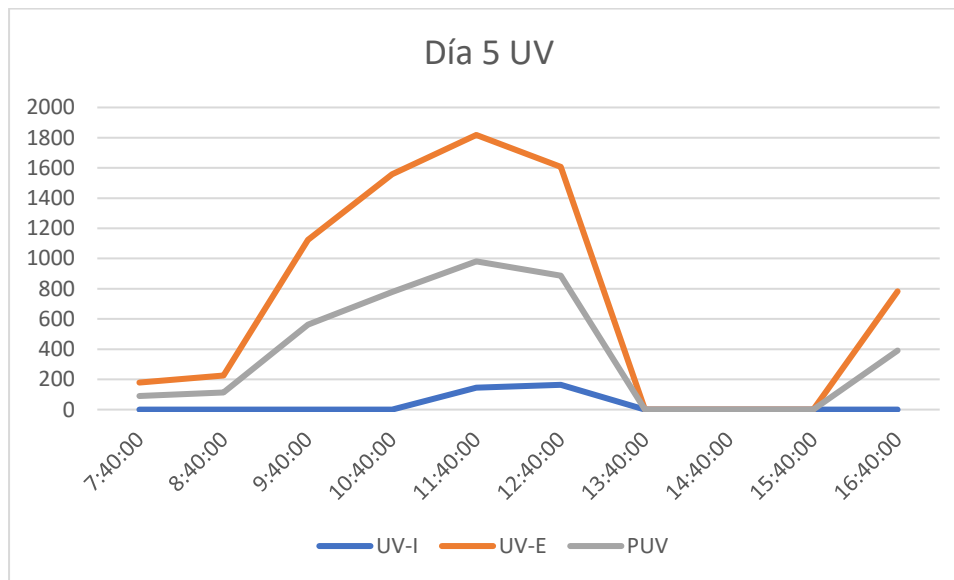
Descripción: Se muestra información detallada sobre las temperaturas registradas en una casa de cultivo durante el día 5. Los datos incluyen la temperatura tanto en el interior (color azul) como en el exterior (color naranja) de la casa de cultivo. Además, se proporciona el promedio de estas temperaturas (color gris).

➤ Gráfica de humedad con DTH22



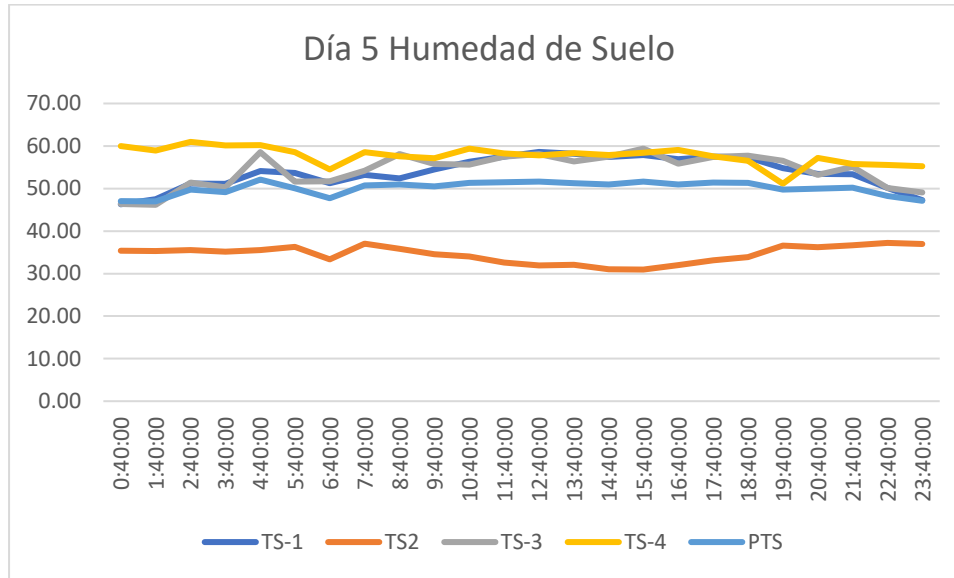
Descripción: Se muestran los datos de humedad en la casa de cultivo durante el día 5, en donde se interpretan los datos tanto de la humedad interior (color azul), exterior (color naranja) y el promedio de la humedad (color gris).

➤ Gráfica con sensores UV



Descripción: Se muestra información detallada sobre el sensor de ultravioleta, registradas en una casa de cultivo durante el día 5. Los datos incluyen el sensor ultravioleta tanto en el interior (color azul) como en el exterior (color naranja) de la casa de cultivo. Además, se proporciona el promedio de las temperaturas del sensor UV (color gris).

➤ Gráfica con sensores de humedad de suelo



Descripción: Se muestran los datos de humedad del suelo en la casa de cultivo durante el día 5, tanto de la humedad de las 4 surcos de cultivos sembrados. Primera línea (color azul), segunda línea (color naranja), tercera línea (color amarillo), cuarta línea (color celeste) y el promedio de la humedad (color gris).

OBSERVACIONES

Durante la realización del proyecto nos topamos con diversos problemas técnicos y de programación derivados del uso de una sola Esp32 para el proyecto, los cuales se describirán a continuación:

- **Module RTC y SD:** Durante la prueba uno de los 2 módulos estaba dañado con corto y el sobrante dejó de funcionar el día del montaje del circuito.
- **Sensores DHT22:** Los sensores DHT22 funcionan en los pines I/O de la Esp32, enviando una señal digital que la Esp32 recibirá, bien sabiendo esto, no teníamos el conocimiento hasta que lo investigamos que al usar los pines VN (39) y VP (36) de la Esp32 causaban interrupciones con los pines I/O de la Esp32. Por lo que perdimos mediciones de varios días, al tratar de buscar el error en conexiones mal soldadas, sensores sueltos y lógica errónea en el código.
- **Sensores UV:** Los sensores UV son sensores que envían datos analógicos que la esp32 recibe y convierte en digitales para poder visualizarlos. La Esp32 tiene 12 pines ADC, divididos equitativamente en ADC1 y ADC2, 6 para cada uno. Los ADC2 tienen una particularidad que es la de que si el programa usa WIFI los ADC2 quedan desactivados para la recepción de datos analógicos, lo cual nos dejó solo 6 pines para 7 sensores ADC, por lo que se descartó un sensor de suelo y se colocaron los 2 sensores UV.

- Caudalímetro: El caudalímetro envía señales digitales ya que convierte los datos en su circuito interior ADC, lo cual al inicio no sabíamos. No hubo el tiempo de arreglarse antes de los 5 días por lo que no se tiene mediciones.

IMÁGENES TOMADAS CON LA ESP32 CAM



Ilustración 28 imagen tomada día 1



Ilustración 29 Imagen tomada día 1 en horas de la tarde



Ilustración 30 Día 3 en horas de la noche



Ilustración 31 Día 4 en horas de la mañana



Ilustración 32 Día 5 en horas de la mañana

APRENDIZAJES SIGNIFICATIVOS

Marichell Araúz

- Aprendí las funciones básicas de los sensores que utilizamos, que fueron de temperatura, de humedad, de suelo, de radiación UV, estos indicaban el momento exacto en que las plantas necesitaban agua y así se mediante la programación realizada se podrían regar de manera automatizada.
- Aprendí que el constante monitoreo de los datos es sumamente importante porque así nos damos cuenta si estamos cumpliendo con las necesidades reales de las plantas y podrían hacerse ajustes para que puedan crecer como es debido y también obtenemos datos precisos del sistema realizado.

Doris Arcia

- La implementación de un sistema de riego basado en IoT en las casas de cultivos tienden a ser más económicas si se les da buen uso ya que no solo es de ayuda económica también lo es para el ambiente ya que se reducirá el uso del agua debido a que con la ayuda de los sensores del suelo que miden la humedad que se tiene en esa parte de tierra se puede determinar en que momento los plantones requieren que se abra la válvula para regarlos y con esto se evitaría el malgaste de agua.
- A pesar de que la casa de cultivo se mantenía cubierta por todos sus lados y la parte de arriba a triple malla la lluvia se filtraba y mantenía los plantones húmedos tal efecto no permitió que la electroválvula se abriera debido a que siempre estaba húmedo; estos aspectos se deben tomar en cuenta ya que en nuestro caso si dos sensores de humedad del suelo marcaban 0 era que se activaba la electroválvula de lo contrario se mantenía cerrada. Dentro de la casa de cultivo los rayos UV casi la mayoría del tiempo se mantenían en 0 por la sombra de la maya.

Caroline Bethancourth

- El proyecto enfrentó desafíos, como problemas con los sensores UV, cambios en la disposición de los pines y otras dificultades operativas durante el monitoreo ya que habíamos tenido la oportunidad de realizar un proyecto similar anteriormente, pero esta vez se necesitaban medir varios cambios ambientales y la necesidad de más sensores y otros componentes no antes usados, La capacidad para gestionar y solucionar problemas en tiempo real es esencial en proyectos de este tipo. Esto incluye la adaptabilidad para ajustar el sistema y mantener su funcionalidad a pesar de los contratiempos que puedan surgir tomando en cuenta que tenemos un tiempo asignado para la toma de datos.
- La incorporación de sensores específicos para la agricultura, como los de humedad del suelo y radiación UV, proporciona conocimientos prácticos sobre cómo estas tecnologías pueden mejorar la eficiencia en el cultivo de plantas y cómo responder a los desafíos específicos del entorno agrícola y a la necesidad en este caso de riego. El proyecto también destaca la importancia de consideraciones ambientales y económicas al proponer un sistema de riego basado en IoT. Comprender cómo estas soluciones pueden contribuir al ahorro de recursos como el agua ya que al momento de los sensores indicar que el área está seca automáticamente se activa el riego por un tiempo determinado y tener un impacto económico positivo ayudará a sector agrícola.

Kevin García

- La Esp32 dependiendo del programa puede variar en el uso de sus pines, ya que un código de pruebas donde todo funciona se adapta a uno con WIFI y conexión a Telegram hacen que por el WIFI dejen de funcionar algunos pines, dejándonos con errores que no aparecían anteriormente y retrasando mediciones.
- Una metodología de programación que ayuda mucho a la hora de encontrar errores es el try except, ya que permite que el código se siga ejecutando con el error que sea y mandando una notificación, muy importante a la hora de trabajar con componentes, ya que ponemos aislar la causa del error y solventarla, de la manera que lo requiera, si es un dispositivo desconectado, lógica errónea o errores de pines.

Jorge Mendoza

- El aprendizaje que he tenido en este proyecto es grandísimo, pude aprender, observar y pensar que para tener grandes proyectos caseros como estos tipos de proyecto no es necesario gastar mucho dinero, hay personas que gastan miles de dólares en dispositivos y los mismo ciertas veces no realizan muy bien el trabajo y dañan al medio ambiente con materiales tóxicos, la verdad solamente es cuestión de ingeniársela e instruirse mucho, estos tipos de proyectos son importantes y ayudan demasiado tanto a las personas agrícolas como al medio ambiente.
- Aprendí como se programaba la construcción de un sistema de riego con el caudalímetro, como funcionaba los sensores los cuales van en el suelo detectando la humedad la misma mandaba datos del consumo de agua del cultivo y se guardaban en una microSD, también aprendí que tan importante es la programación en estos casos, ya que puede marcar la diferencia en la gestión del agua y la productividad de este.

Cesar Quijada

- En el transcurso de este proyecto, se llevó a cabo la instalación de un sistema de riego en una casa de cultivo. Durante el proceso, se implementó un monitoreo exhaustivo de la temperatura tanto dentro como fuera de la estructura. Además, se realizaron capturas de información proveniente de sensores de ultravioleta y de humedad del suelo. Todos estos datos se canalizaron y enviaron a través de un Bot personalizado de Telegram, donde se registraron y almacenaron los datos provenientes de los sensores instalados en la casa de cultivo. Este enfoque integral permite un control más preciso y una gestión eficiente de las condiciones ambientales, contribuyendo así al éxito del cultivo al proporcionar información en tiempo real sobre factores clave para el desarrollo de los cultivos.
- En la casa de cultivo, implementamos un sistema de riego que incorpora un caudalímetro y una electroválvula para una gestión precisa del agua. El caudalímetro mide con exactitud la cantidad de agua suministrada, permitiendo ajustes acordes a las necesidades de las plantas. La electroválvula, por su parte, controla automáticamente el flujo de agua en el sistema, abriendo y cerrando según las programaciones establecidas. Este sistema se distribuyó eficientemente en cuatro surcos de plantones, posibilitando un manejo individualizado del riego en cada área y asegurando condiciones óptimas para el crecimiento saludable de las plantas.

Isabel Rodríguez

- Durante el desarrollo del proyecto de la casa de cultivo, uno de los desafíos más significativos fue la gestión de la humedad excesiva dentro de la estructura, causada principalmente por la filtración de agua de lluvia a pesar de estar cubierta. Este problema afectó directamente la efectividad del sistema de riego, ya que la electroválvula solenoide, se activaba en función de la medición de sensores de humedad del suelo, a menudo donde indicaba niveles altos debido a la humedad ambiental. De modo que se ajustaron los umbrales de activación de la electroválvula solenoide en el código del sistema. Aunque esta fue una solución temporal, permitió reducir la sensibilidad de los sensores de humedad del suelo, evitando activaciones innecesarias. Se modificó la frecuencia de lectura de los sensores de humedad del suelo para que no se tomaran mediciones constantes.
- Al implementar ciertas funciones, como la conexión a Wi-Fi y la transmisión de datos a través de Telegram, algunos pines específicos de la ESP32 dejaban de funcionar correctamente. Esto afectó la operación de sensores analógicos, como los sensores UV, que requerían pines ADC específicos. Se revisaron las asignaciones de pines de la ESP32 y se dio prioridad a las funciones esenciales. Se reasignaron pines para acomodar tanto la conexión a Wi-Fi.

Javier Rodríguez

- Gané experiencia en el manejo y programación de dispositivos IoT, en este caso, la ESP32 y la ESP32 CAM. Esto incluyó la configuración de conexiones Wi-Fi, la gestión de energía y la programación para realizar tareas específicas.
- Integré los conocimientos teóricos adquiridos en mis estudios de informática y electrónica en un proyecto práctico y funcional. Esta aplicación directa me ayudó a consolidar y contextualizar la teoría, fortaleciendo mi comprensión global de los conceptos aprendidos.
- Comprendí los principios detrás del control de una electroválvula para la automatización del riego. Donde utilizamos el sensor de humedad del suelo para conocer las necesidades del cultivo y así saber cuándo activar y desactivar la electroválvula, optimizando el uso del agua.

Joseline Sánchez

- La integración de sensores UV, temperatura, humedad y suelo en un sistema de riego con electroválvula y una ESP32 permite un control preciso del entorno, ajustando la frecuencia y cantidad de riego según las condiciones específicas de la planta, optimizando así el uso del agua y mejorando el desarrollo vegetal.
- La utilización de un relé en este proyecto posibilita la automatización del sistema al conectar la electroválvula al controlador (ESP32), permitiendo activar o desactivar el riego de manera remota o programada en función de la información recopilada por los diferentes sensores, lo que contribuye a una gestión eficiente y sostenible del cultivo.

Osvaldo Torrero

- Aprendí sobre la importancia de la constancia en el cuidado de los cultivos, por lo tanto, la optimización y automatización es crucial.
- Como la electrónica impacta positivamente en nuestras necesidades diarias como la de conseguir alimentos de calidad.

- Como la electrónica digital trabaja con distintos dispositivos que unidos a la programación pueden hacer aportes importantes en la resolución de problemáticas sociales.

Carlos Ureña

- Poder entender cómo funcionan internamente los envíos de datos de los Bot en Telegram, cómo los puedo conectar mediando códigos, en este caso mediante la API de Telegram.
- Poder aprender a calibrar el sensor de suelo, ya que dependiendo del voltaje que se le suministra cambia el rango de valores en la lectura y para poder sacar el porcentaje de valores ADC se necesita poder entender estos valores

APORTES INDIVIDUALES

Marichell Araúz

Pude asistir 3 días a la casa del compañero donde se realizó el proyecto, ayude en las mejoras de la arquitectura de la casa del cultivo, luego ayude a colocar las tuberías debíamos cortar, pegar para lograr tener un correcto sistema de riego, también ayude el día que se colocaron los componentes necesarios para el proyecto y una vez obtenidos los resultados de los sensores elabore las graficas con los datos donde se puede observar el promedio de cada sensor.

Doris Arcia

Fui donde se realizó el proyecto 4 veces en esas veces ayude con preparación de la tierra, llevada de plántones de tomates, buscar varas para armar estructura, armar estructura, siembra de plántones, cortar tubos, armar la estructura de la tubería dentro de la casa de cultivo, armar el circuito dentro de la casa de cultivo, descripción del proyecto, descripción de software y hardware del proyecto, realice el diagrama esquemático de la ESP32 con los componentes y el de la ESP32 CAM, descripción de cada parte del diagrama esquemático de la ESP32 y la ESP32 CAM, unión del informe, recolectar y hacer tablas de los datos durante los 5 días y también realice mi aporte económicamente para la compra de los materiales que hacían falta

Caroline Bethancourth

Aporte en asistir en dos ocasiones a la casa del compañero para la construcción de la casa cultivo; buscando materiales como ramas de árboles para su elaboración (casa cultivo). Aporte en la siembra de plántones y en la limpieza del lugar donde se iba a plantar; realice aporte económicamente para la compra de materiales para implementar la parte tecnológica; en la construcción de las conexiones con tubería de PVC para que los plántones se regaran; en la instalación de los sensores una vez terminado los programas para que empezaran a tomar datos.

Kevin García

Programación de parte del programa que se cargó en la ESP32 y la ESP32 CAM, soldadura del circuito en la PCB, conexión de los sensores a la PCB, constante revisión del código, para corregir errores durante las mediciones.

Jorge Mendoza

En este proyecto fue en la casa del compañero en el cual el aporte fue en 2 ocasiones en la primera en lo que ayude fue buscando materiales, cortando ramas, haciendo pequeñas estacas para ponerle a la

maya en las esquinas de ala parte de afuera, también en la puesta de la maya sobre la estructura asegurando de que no hubiese problemas con el viento, ni que se soltara los nudos. En la segunda ida al proyecto mi aporte fue buscando materiales nuevamente como madera en forma de Y para la instalación de la luz en el proyecto, también estuve cortando las tuberías de PVC marcadas por los compañeros para armar el sistema de riego y que empezara regar las plantas, armado de tuviera el cual quedo funcionando perfectamente.

Cesar Quijada

Mi participación durante este proyecto fue activa en la confección y armado de la casa de cultivo en donde ayude en cortar los palos para la estructura que se diseñó para el mismo. También aporte en sembrar los cultivos y también en extender la malla que se utilizó para la estructura de la casa de cultivo. Aporte también en el cableado de la casa de cultivo y de la conexión de las tuberías a la electroválvula y el caudalímetro.

Isabel Rodríguez

Participo activamente en la armadura y construcción de la casa de cultivo, colaborando desde el inicio con la colocación de postes con palos para reforzar los alrededores y darle una estructura sólida. Ayudé a atar cada poste con hilos resistentes, asegurándolos firmemente. También cooperé en extender y asegurar las redes de malla que cubrirían completamente la edificación, atando las esquinas y amarrando los bordes para que quedara bien firme.

Estuve presente en el proceso de armado de la estructura de tubos de PVC, ayudando a cortarlos, ensamblarlos y colocarlos según el plano que se tenía para que los sellos no tuvieran filtraciones ni pérdidas de agua.

Además, participo en colocar, conectar y enterrar los sensores de humedad, luminosidad y temperatura que monitorearían las condiciones del suelo y ambiente. También pasé los cables de conexión, protegiéndolos en tuberías de UTP para que no sufran daños. Finalmente participo y coloqué un pequeño margen de palos que se hizo para cubrir y proteger la zona por donde entraban los cables al interior de la casa de cultivo.

Javier Rodríguez

Iniciar dándole cuidado a las plantitas del cultivo. Cortar estacas para darle más soporte al alrededor del cultivo. Realicé una excavación cuidadosa para exponer la tubería de agua potable, facilitando su conexión al sistema de la casa y mejorando la eficiencia en el suministro hídrico del cultivo. Hacer empates en conexión para el fluido eléctrico.

Además, mi compromiso se reflejó en mi participación diaria en todas las actividades del proyecto. Estuve activamente involucrado en la recolección constante de datos de la ESP32-CAM, asegurándome de que los códigos fueran probados y subidos a la ESP32 de manera efectiva. Mi labor incluyó la verificación continua para garantizar que los sensores continuaran registrando datos sin interrupciones, y me aseguré de que ningún cable se desconectara, manteniendo así la integridad y la estabilidad del sistema en todo momento.

Joseline Sánchez

En este gran proyecto tuve la intervención en la estructura del cultivo ya sea, buscando madera para darle forma o para reforzarlo. Pude aportar en lograr llevar electricidad hasta la casa del cultivo colocando postes para que el cableado no saliera afectado por el agua del suelo.

Osvaldo Torrero

Estuve un día en el lugar donde se desarrolló el proyecto y participe en la búsqueda de estacas para estructura del cableado, apoyo en construcción de estructura para el cableado del circuito e instalación de este. Apoyo en la instalación del cableado de la electroválvula y caudalímetro. Recopilación de material gráfico. (Fotos y videos). Producción de video recopilatorio del proyecto.

Carlos Ureña

Programación de la esp32 para que captara los datos del sensor de suelo a su vez calibrarlos y crear las condiciones necesarias para interactuar con la electroválvula y el caudalímetro.

Creación de Bot en Telegram para la comunicación con los sensores a la vez que creación de un grupo para recibir los datos de los Bots

LINK DEL VIDEO

<https://drive.google.com/file/d/1nsmcnResqbJ2due4P0VfYU8x6Xp3Rwam/view>

CONCLUSIÓN

El desarrollo del proyecto se centró en la aplicación de diversas tecnologías vinculadas a la Teleinformática y la Electrónica Digital para instaurar un sistema automatizado de monitoreo y control de riego en una Casa de Cultivo. Se empleó una placa de circuito impreso (PCB) para la implementación del circuito, gestionando la ESP-32, el módulo de almacenamiento y el módulo de alimentación de 5V para Protoboard. En este proceso, se integraron sensores de humedad, luminosidad y temperatura para evaluar las condiciones del suelo y del entorno, junto con una electroválvula y un caudalímetro para regular el riego.

La operatividad del sistema sigue una secuencia específica: los sensores capturan datos sobre humedad, luminosidad y temperatura, los cuales se transmiten a la ESP-32. Esta última procesa la información y emite señales a la electroválvula y al caudalímetro para gestionar el riego. Los datos recopilados durante cinco días se almacenan en una memoria SD y se publican en un servidor accesible a través de una página web.

En las gráficas proporcionadas, derivadas de los datos descriptivos sobre las condiciones de temperatura, humedad y radiación UV en una casa de cultivo durante cinco días, se pueden extraer las siguientes observaciones:

- Las temperaturas, tanto internas como externas, fluctúan a lo largo de los días, manteniendo la temperatura interna relativamente estable y, en promedio, algunos grados por encima de la temperatura exterior.
- La humedad varía diariamente, siendo mayor en el exterior que en el interior de la casa de cultivo. No obstante, el promedio de humedad tiende a mantenerse estable durante el período analizado.
- La humedad del suelo se mantiene uniforme en diferentes surcos de cultivo, con variaciones graduales día a día y un promedio que se conserva sin cambios significativos.
- Los sensores UV interior y exterior registran valores de radiación fluctuantes día a día, siendo la radiación UV exterior generalmente más alta que la interior.

Además, durante la ejecución del proyecto, se enfrentaron diversos inconvenientes técnicos derivados de la concentración de componentes y lógica de programa en una sola placa ESP32:

1. Se experimentaron fallas en los módulos RTC y SD, afectando el registro de datos en un momento determinado.
2. La interferencia en los pines de lectura de los sensores DHT22, al utilizar pines de alimentación específicos de la ESP32, resultó en la pérdida de mediciones durante varios días.
3. La limitación en los pines ADC disponibles al activar WiFi llevó a descartar una medición de humedad del suelo inicialmente planeada.
4. El desconocimiento inicial sobre la salida digital del caudalímetro impidió su integración y lectura de datos a tiempo.

En resumen, la concentración de múltiples componentes en una sola placa ESP32, junto con fallos puntuales y desconocimientos técnicos, generaron contratiempos para obtener todas las mediciones deseadas.

Los datos obtenidos durante los datos registrados por los sensores y las fotografías tomadas por la ESP32 CAM se transmitieron a través de internet y se visualizaron en tiempo real en el grupo de Telegram creado. De esta manera, se puede hacer un monitoreo remoto del estado y crecimiento de la plantación.

Además de la implementación del sistema de monitoreo y control automático de riego, los estudiantes participaron en la preparación del suelo, la construcción de la estructura de la casa de cultivo, la siembra de los plántones y la instalación de los sensores y el sistema de riego. Asimismo, realizaron la edición de un video que documenta el proyecto desde su inicio hasta su conclusión.

BIBLIOGRAFÍA

IoT Camera using ESP32 Cam and Telegram. (2023, noviembre 12).

Llamas, L. (2023, agosto 25). Cómo usar las entradas analógicas ADC en un ESP32. Luis Llamas.
<https://www.luisllamas.es/esp32-adc/>

Random Nerd Tutorials. (2019, abril 27). Random Nerd Tutorials. <https://randomnerdtutorials.com/>

EVALUACIÓN

Informe escrito (35%)	Puntos				
a. Presentación*	2				
b. Introducción	2				
c. Metodología					
• Descripción general del proyecto	2				
• Descripción de software y hardware utilizado	3				
• Diagrama esquemático	3				
• Descripción de cada parte del diagrama esquemático	2				
• Descripción del código en ESP32	3				
• Descripción del código en ESP32 CAM	3				
• Descripción de figuras	2				
d. Aprendizajes significativos**	5				
e. Aportaciones individuales	6				
f. Referencia***	2				
Subtotal	35				
Sustentación (25 %)					
a. Video tutorial****	15				
b. Preguntas y respuestas	10				
Subtotal	25				
Funcionamiento (40%)					
a. Registro en microSD de datos de T&H	4				
b. Registro en microSD de datos de Humedad de suelo	4				
c. Registro en microSD de datos de radiación solar	4				
d. Registro en microSD de datos de consumo de agua para el riego.	4				
e. Presentación en internet de datos de T&H	4				
f. Presentación en internet de datos de Humedad de suelo	4				
g. Presentación en internet de datos de radiación solar	4				
h. Presentación en internet de datos de consumo de agua para el riego	4				
i. Activación correcta de la electroválvula	8				
j. Opcional. Envío de imágenes fotográfica a través de internet.	20				
Subtotal	40/60				
TOTAL	100/120				