UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Diseño e implementación de un sistema IoT mediante la plataforma ESP32 para la automatización del proceso de compostaje de residuos orgánicos domésticos

Trabajo final de máster

Máster universitario en ingeniería mecatrónica

AUTOR: Alberto García Martínez

TUTOR: Francisco José Gimeno Sales

CURSO: 2020 - 2021

DOCUMENTOS QUE CONFORMAN EL PROYECTO

Documento I. MEMORIA Y ANEXOS

Documento II. PLIEGO DE CONDICIONES

Documento III. PRESUPUESTO

RESUMEN

El siguiente trabajo final de máster trata el diseño y construcción de un prototipo para monitorizar y controlar las variables que determinan la eficacia del proceso de compostaje aeróbico de los residuos orgánicos domésticos. El sistema utiliza tecnologías del internet de las cosas (IoT) y está enfocado a su implementación a pequeña escala para uso en viviendas.

Se programará una placa de desarrollo basada en ESP32 para realizar la lectura de los diferentes sensores y la puesta en marcha de medidas correctivas que mantengan el proceso en condiciones óptimas para los microorganismos que compostan la materia orgánica.

Los datos recogidos por los diferentes sensores se envían mediante WiFi a un servidor local alojado en una Raspberry Pi, donde se almacenan en una base de datos SQL. El usuario puede conectarse al servidor desde el navegador web de cualquier equipo conectado a la red local para consultar los datos del proceso o descargarlos en otro dispositivo. Conociendo el valor de variables como la temperatura y humedad del compost, es posible saber en qué punto del proceso se encuentra o si se está desarrollando de manera adecuada.

ABSTRACT

This project covers the design and construction of a prototype to monitor and control the variables that determine the efficacy of the domestic organic waste aerobic composting process. The system uses Internet of Things (IoT) technologies and is focused on its implementation in a small scale for home use.

A development board based on ESP32 will be programmed to read the different sensors and perform corrective measures to maintain the conditions of the process in optimum conditions for the composting microorganisms to digest the organic residues.

Data collected form the sensors will be sent using WiFi to a local server held on a Raspberry Pi, where it will be stored in an SQL Database. The user can connect to this server from any web browser inside the local network to check the state of the project or download data to a different device. Knowing the different values of variables such as the compost temperature and humidity, users can interpretate the state of the process and whether it is evolving correctly or not.

AGRADECIMIENTOS

A la comunidad Maker de internet y en especial a Rui Santos y Jhimmy Astoraque por compartir su conocimiento de manera desinteresada.

A mi amigo José Navarrete Carbonell por ayudarme a depurar el código cuando yo no era capaz de encontrar los errores del programa.

A mi tutor y a todos los que me han ayudado con este proyecto.

Gracias.

Documento I. MEMORIA Y ANEXOS

Contenido de la memoria

Índice de figuras

1. INTRODUCCIÓN

1.1. Objetivo

El objetivo de este proyecto es introducirse en la programación de microcontroladores orientada al internet de las cosas (IoT). Se esContudiará la transmisión de datos inalámbrica mediante WiFi entre una placa de desarrollo basada en ESP32 y un servidor local donde se almacenará la información para poder ser consultada por el usuario.

Para aprender sobre la programación de los diferentes protocolos y estructuras propias del IoT desde un punto de vista práctico, se llevará a cabo el diseño y construcción de un prototipo de sistema para la automatización del proceso de compostaje de residuos orgánicos domésticos. El prototipo pretende ser de bajo coste, utilizando plataformas de hardware libre y código abierto y manteniéndose simple tanto en diseño como construcción. De esta forma, se desea implementar un sistema que permita su instalación y uso en viviendas por parte de cualquier usuario con nociones de electrónica. Este prototipo no se proyecta con otra intención distinta a la puramente educacional, sin ninguna ambición de ser comercializado.

El diseño del sistema de automatización del proceso de compostaje requerirá la selección de diferentes sensores que permitan conocer el estado del proceso y actuadores para la toma de medidas correctivas que mantengan la materia orgánica en condiciones óptimas. También será necesaria la programación de un controlador que procese la información de dichos sensores y accione los actuadores cuando sea necesario. Este controlador también realizará los envíos de información hacia el servidor local, cuya arquitectura y programación también se tratarán en esta memoria.

Tras su implementación, el prototipo permitirá al usuario controlar de forma automática las condiciones del proceso de compostaje, optimizándolo al máximo. Además, permitirá conocer en qué punto se encuentra la fermentación de la materia orgánica conectándose de forma remota al servidor para consultar los datos almacenados en él o descargarlos en un formato que permita obtener estadísticas detalladas.

1.2. Estado del arte: Internet de las cosas y hardware libre

1.2.1. Internet de las cosas

Durante los últimos 20 años, los avances en conectividad máquina a máquina (M2M) y el abaratamiento de sistemas microelectrónicos y sensores han provocado un enorme desarrollo del internet de las cosas (IoT, del inglés). El término *Internet of Things* fue utilizado por primera vez en 1999 por el investigador del MIT Kevin Ashton, quien expuso la necesidad de crear protocolos estandarizados para que los ordenadores entendieran el mundo real, recopilando e interpretando datos del entorno sin ayuda humana. El término IoT se refiere a las diferentes tecnologías que permiten a los dispositivos físicos conectarse entre sí para intercambiar información, bien de forma directa o bien a través de internet. [1] [2]

Los avances en la tecnología IoT están causados por el desarrollo en cinco áreas clave. Como ya se ha introducido anteriormente, la disponibilidad de sensores de bajo coste y baja potencia permite recoger información de forma fiable sin grandes limitaciones. Las nuevas tecnologías de comunicación y protocolos de red han permitido conectar a internet una enorme cantidad de dispositivos de manera fácil, consiguiendo transmitir la información eficientemente. La creación de plataformas informáticas en la nube ha permitido a los sistemas IoT escalar desde aplicaciones pequeñas hasta ideas altamente complejas. También es destacable la relación de simbiosis entre el *Machine Learning* y el IoT, que permite analizar gran cantidad de datos de manera rápida facilitando la toma de decisiones. Las tecnologías de *Machine Learning* se nutren a su vez de la información recogida por los dispositivos IoT para desarrollarse. Por último, la evolución del internet de las cosas va de la mano de los avances en inteligencia artificial. Esta tecnología permite a los sistemas informáticos interpretar datos de su entorno de forma autónoma y decidir qué acciones son necesarias para alcanzar con éxito los objetivos o tareas marcadas. [2] [3]

La tecnología del internet de las cosas tiene multitud de aplicaciones prácticas tanto a nivel industrial como doméstico. El IoT es parte clave de la industria 4.0, permitiendo desarrollar procesos de fabricación inteligentes que utilizan datos a tiempo real para adaptar los métodos de producción o prever fallos y roturas.
Tener un mayor control sobre el estado de los procesos industriales es de gran utilidad para mejorar su eficiencia y hacerlos más resilientes. La logística o la agricultura

inteligente no podrían existir sin la tecnología IoT. El internet de las cosas tiene también aplicaciones en el campo de la medicina, permitiendo a los médicos monitorizar el estado del paciente de forma más eficiente y realizar diagnósticos más eficaces que lleven a tratamientos más específicos para cada enfermo.

A nivel doméstico, son muchos los dispositivos interconectados que podemos encontrar en nuestro día a día. Los asistentes virtuales se basan en esencia en el internet de las cosas, acercando la domótica al gran público. Los relojes inteligentes, hornos, frigoríficos y en general casi cualquier objeto de nuestro día a día puede conectarse a internet para mejorar la interacción con el usuario y funcionar de manera más eficiente.

1.2.2. Plataformas de hardware libre

Las plataformas de hardware libre son todos aquellos dispositivos creados mediante diseños públicos y accesibles para sus consumidores. Esto permite a los usuarios estudiar, modificar, reutilizar, adaptar o incluso redistribuir los dispositivos de forma independiente. Su aparición es una evolución del software libre, adaptando los principios de este a los dispositivos físicos. El desarrollo de hardware libre se ha visto potenciado en las últimas dos décadas por el abaratamiento de los componentes electrónicos y el auge de la filosofía del *do it yourself,* o hazlo tú mismo.

La ambición por reparar, adaptar o mejorar todo tipo de utensilios es una de las bases de la filosofía del “hazlo tú mismo”. La comunidad de aficionados a esta idea se agrupa tanto a través de internet como en asociaciones locales de *Makers,* donde comparten información y consejos desarrollando todo tipo de productos propios. Los Makers utilizan hardware libre para construir sus propias creaciones, utilizando métodos de fabricación tradicionales, pero también nuevas técnicas como la impresión 3D. [4]

Un potenciador clave del hardware libre son las plataformas electrónicas Arduino. Arduino es un proyecto que desarrolla plataformas de creación electrónica de hardware y software libre. Las placas Arduino tienen todos los elementos para programar y utilizar de forma sencilla microcontroladores de la familia ATMEL. A través de una IDE propia y con un lenguaje de programación simple basado en C, cualquier usuario puede conectar sensores y actuadores a las entradas y salidas de las placas Arduino. [5]

El ecosistema creado alrededor del proyecto Arduino ha propiciado la aparición de nuevas placas de desarrollo utilizando distintos microcontroladores. Es el caso, entre otros, de WeMos D1 R32, la placa de control utilizada en este proyecto y que se basa en un circuito integrado de la familia ESP. Esta placa de desarrollo añade conectividad WiFi y bluetooth a las funcionalidades típicas de este tipo de microcontroladores. Cuenta además con un potente procesador que la hace versátil para todo tipo de proyectos. La Wemos D1 R32 es totalmente compatible con la IDE y librerías de Arduino. En sucesivos puntos de esta memoria se profundizará en su funcionamiento.

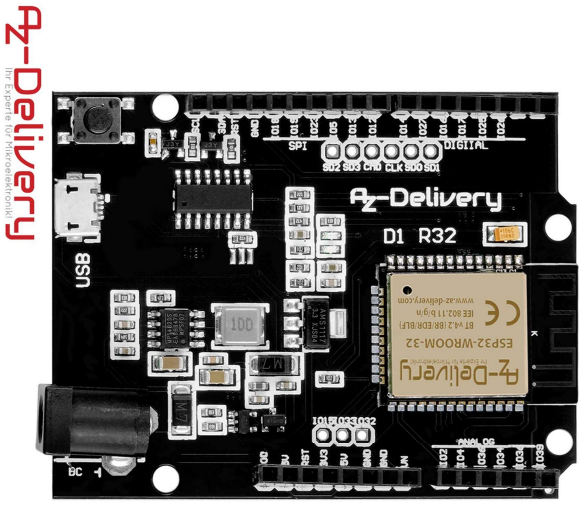


Figura 1: Placa de desarrollo D1 R32, versión producida por AZ Delivery. Obtenida de amazon.es

Otro agente esencial en el bum del hardware y software libres es la Raspberry Pi Fundation. Esta organización sin ánimo de lucro pretende acercar el mundo de la programación y la creación digital al gran público y, en especial, a los niños. Para hacerlo, desarrollaron la Raspberry Pi, un ordenador de placa única y del tamaño de una tarjeta de crédito. El dispositivo es de bajo coste y trae todas las funcionalidades de un ordenador con sistema operativo Linux. Su tamaño, conectividad, programabilidad y precio han llevado a la Raspberry Pi a ser utilizada en multitud de proyectos de robótica, domótica o automatización tanto a nivel doméstico como industrial. [6]

Las plataformas de hardware y software libre también han llevado el internet de las cosas a manos de miles de aficionados a la electrónica y la programación alrededor del mundo. Las opciones de conectividad que ofrece la tecnología IoT permite llevar proyectos caseros a un nuevo nivel.

1.3. Proceso de obtención de compost y variables a controlar

El compost es un abono obtenido mediante la descomposición de todo tipo de residuos orgánicos. Su uso sustituye al de fertilizantes químicos en agricultura y jardinería. La práctica del compostaje permite reducir la cantidad de basura que termina en vertederos y devolver al suelo materia orgánica enriqueciéndolo de forma natural.

La preocupación por el medio ambiente y por formas de vida más sostenibles lleva a cada vez más personas a producir su propio compost, reduciendo así su generación de residuos. Tal es la necesidad de una gestión más racional de los desechos domésticos, que El Ministerio de Medio Ambiente y Medio Rural y Marino publicó en 2009 el Manual de Compostaje [7]. El manual es base fundamental de este trabajo final de máster ya que en él se explica de forma clara y sencilla cómo llevar el proceso de compostaje a los hogares.

El proceso de compostaje consiste en crear las condiciones idóneas para que organismos descomponedores fabriquen abono a partir de materia orgánica. Los residuos compostables que se pueden generar a nivel doméstico son diversos, y se clasifican según la velocidad a la que se descomponen en tres categorías:

- **Materiales de rápida descomposición:** En esta categoría entran desechos como los restos de siega de césped, la maleza y los restos de poda frescos.
- **Materiales de descomposición lenta:** Aquí se incluyen los restos de frutas y verduras, posos de café, bolsas de infusiones, restos de poda más consistentes y otros restos de comida evitando lácteos y grasas. También se pueden considerar residuos compostables de descomposición lenta el cartón, las servilletas y bolsas de papel y el papel de periódico.
- **Materiales de descomposición muy lenta:** Los materiales que más tardan en descomponerse son las hojas de otoño, ramas podadas, cáscaras de huevo, lana y tejidos naturales y huesos de fruta entre otros.

Para que el compost generado sea de calidad, hay que alternar capas de materia orgánica con distinta velocidad de descomposición. De esta forma conseguimos un abono consistente y rico en nutrientes para nuestras plantas. El material que se añada a la pila de compost tendrá que estar en trozos lo más pequeños posibles y con una relación 2/1 entre materia húmeda y seca.

Si bien el compost puede producirse en una simple pila de residuos sobre el suelo, a nivel doméstico resulta más práctico utilizar un contenedor al que llamaremos compostera. La compostera es un recipiente con tapa generalmente de madera o plástico que mantiene los restos orgánicos juntos durante el proceso de compostaje. Introducir los residuos en una compostera ayuda a conservar la temperatura de la materia orgánica constante y evita que se inunde cuando llueve.

Figura 2: Ejemplo de compostera con residuos en su interior. Imagen obtenida de blog.oxfamintermon.org/como-hacer-compost-casero/ (2021)

Es importante que la compostera permita un fujo de aire adecuado en el interior del compost, por lo que deberá tener ranuras u orificios en sus paredes. A la hora de introducir residuos orgánicos, la primera adición deberá suponer un 50% del volumen de la compostera, para garantizar que en el interior de la pila se pueda mantener a una temperatura constante. En sucesivas adiciones, se mezclará la materia nueva con la ya en descomposición. De esta forma, aceleramos el proceso y evitamos que los nuevos residuos atraigan moscas.

La descomposición de la materia orgánica creando compost es idealmente un proceso de oxidación en el que intervienen distintos microorganismos presentes en el ambiente. Esto implica que durante la degradación de los residuos es esencial la presencia de oxígeno. De esta forma, los microorganismos llevarán a cabo una fermentación aeróbica de la materia orgánica, aportando nitratos y fosfatos al compost. La falta de aire provoca que el proceso se vuelva anaeróbico o de putrefacción, obteniendo un abono de peor calidad. [8]

El proceso de compostaje suele llevar entre cuatro y seis meses, dependiendo de la materia orgánica utilizada y las condiciones ambientales. Es por tanto un proceso lento y que se divide en tres grandes bloques:

1) **Fase de latencia y crecimiento:** Esta fase dura entre dos y siete días, si bien en sistemas domésticos puede alargarse. En ella los microorganismos presentes en la materia orgánica y el aire se asientan en su nuevo medio, comenzando a degradar los compuestos más simples. La pila de materia orgánica comienza a calentarse por la acción de las bacterias mesófilas que habitan el compost, llegando al final de esta fase hasta unos 45ºC.

2) **Fase termófila:** Al alcanzar el compost los 45ºC durante la etapa de latencia, aparecen los organismos termófilos, que dan nombre a esta fase. Estos hongos y bacterias son capaces de degradar la materia orgánica de forma más rápida, provocando un aumento de la temperatura hasta situarse entre 60 y 70ºC. En sistemas de fermentación lenta como son los sistemas domésticos, esta fase dura entre uno y dos meses. Durante este tiempo, las altas temperaturas consiguen eliminar los gérmenes patógenos, así como posibles larvas y semillas.

3) **Fase de maduración:** Esta última etapa es la más lenta y puede durar hasta tres meses. La fermentación continúa descomponiendo la materia orgánica más resistente. La temperatura va disminuyendo poco a poco y con ella la presencia de organismos termófilos. Estos microorganismos ceden terreno a otro tipo de hongos y bacterias que degradarán la parte más dura. En ocasiones es beneficioso añadir lombrices y escarabajos al compost para acelerar la maduración. Esta fase termina cuando la temperatura del compost es cercana a la temperatura ambiente.

Conociendo ya el proceso de compostaje, es momento de analizar las distintas variables que deberá controlar el sistema de automatización a crear. Al tratarse de un proceso natural, existirán variables que no se podrán hacer evolucionar artificialmente y por tanto su control se limitará a la monitorización de su estado. De otra parte, podremos actuar sobre otras garantizando unas condiciones óptimas para los microorganismos que crean el compost y evitando la colonización de la materia orgánica por parte de insectos o bacterias nocivas.

Los factores más importantes a la hora de obtener compost son la temperatura, humedad y aireación. A continuación, se explicará en detalle la influencia de cada uno en el proceso y a qué niveles es recomendable mantenerlos según el manual de compostaje del ministerio de medio ambiente [7].

- **Temperatura:** La temperatura no es constante a lo largo del proceso de compostaje. Depende de la etapa en la que se encuentre el compost y los microorganismos que habiten la pila. La fase de latencia se produce a entre 15 y 45ºC. Durante la etapa termófila las temperaturas se mueven entre los 45 y 70ºC, mientras que a lo largo de la fase de maduración la temperatura se mantendrá por debajo de los 40ºC. Puesto que se trata de una variable que depende de la actividad de los microorganismos, no es posible controlar la temperatura del compost, pero sí es conveniente monitorizarla para saber en qué fase se encuentra el proceso como se muestra en la figura 3.

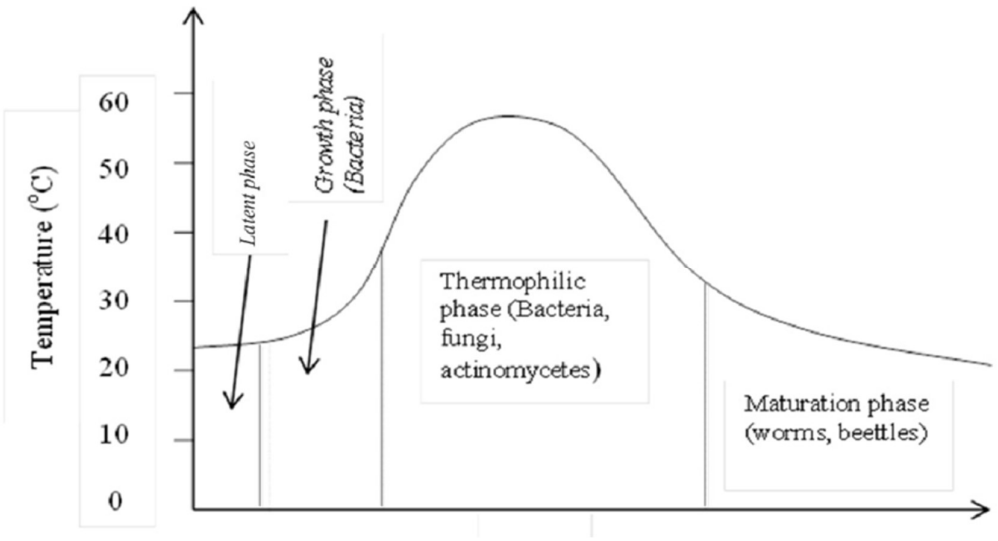


Figura 3: Evolución de la temperatura a lo largo del tiempo. Gráfica obtenida de "Sustainable Operation of Composting in Solid Waste Management" [8]

- **Humedad:** Se trata de un factor fundamental para los microorganismos que habitan en el compost. Un exceso de agua podría provocar falta de oxígeno, alejando el proceso de las condiciones aeróbicas. La falta de humedad, por su parte, dificulta la proliferación de microorganismos y ralentiza significativamente el compostaje. Para crear unas condiciones idóneas, es necesario mantener una humedad de entre el 40 y 60%.

- **Aireación:** La fermentación aeróbica consume gran cantidad de oxígeno. La falta de aire puede provocar fermentación anaeróbica, produciendo malos olores y atrayendo insectos y roedores. Para asegurar un buen compostaje hay que mantener la materia orgánica ventilada y el nivel de metano lo más bajo posible.

A partir de la información de este punto, se programará el microcontrolador para mantener la humedad y aireación a niveles adecuados. También se hará un seguimiento de la temperatura que el usuario podrá interpretar para saber el estado del sistema.

1.4. Alcance del proyecto

Esta memoria pretende explicar en detalle la elaboración del proyecto final de máster “Diseño e implementación de un sistema IoT mediante la plataforma ESP32 para la automatización del proceso de compostaje de residuos orgánicos domésticos”. Se profundizará en las áreas de control, sensores y comunicaciones por ser las más relacionadas con el máster en ingeniería mecatrónica.

A lo largo del documento, se detallará paso a paso el diseño y construcción de un prototipo de sistema que permita automatizar el proceso de compostaje de residuos orgánicos domésticos. Se empezará por explicar el diseño y partes de la compostera inteligente, la arquitectura del sistema de control y el diseño electrónico con la selección de componentes. A continuación, se detallará el funcionamiento del sistema de control, explicando su programación y el funcionamiento de la interfaz que permite al usuario visualizar los datos. Para terminar, se comentará la construcción del prototipo y su puesta en marcha.

La construcción del prototipo deberá ser lo más simple posible, pudiendo reproducirse por cualquiera que desee hacerlo. El proyecto se lleva a cabo sin ningún fin comercial, por lo que se obviarán los requisitos necesarios para escalar y producir en masa el sistema. A continuación, se detallan los requisitos que debe cumplir el prototipo.

1.4.1. Requerimientos mínimos

Para considerar el proyecto como concluido, el prototipo deberá cumplir una serie de requisitos concretos para controlar el proceso a la vez que permita al usuario hacer un seguimiento del compostaje. Además, se añaden otras condiciones que acotan el proyecto y facilitan la construcción del prototipo. Los requisitos mínimos se listan a continuación:

1) El sistema deberá mantener las condiciones de humedad y aireación adecuadas para la proliferación de los microorganismos que descomponen la materia orgánica produciendo compost. Esto es: un nivel de humedad de entre el 40 y 60% y un ambiente libre de metano.

2) El sistema de riego encargado de mantener la humedad no podrá depender de la disponibilidad de tomas de agua corriente cerca de la compostera. Por tanto, la alimentación se realizará desde un depósito utilizando una bomba.

3) El sistema de aireación deberá ser resistente a la humedad y a los sólidos presentes en la compostera, sin atascarse las salidas de aire por tapones de materia orgánica.

4) El sistema deberá almacenar los datos en una base de datos desde la que poder descargarlos para analizarlos si se desea.

5) Los usuarios podrán conectarse al servidor de forma cómoda para visualizar el estado del proceso, mostrando los valores almacenados en la base de datos. Esto incluirá las temperaturas y humedad ambiente y del compost, la presencia de metano en la compostera y la disponibilidad de agua para riego.

Cumpliendo estos requisitos, se podrán considerar alcanzados los objetivos que marca este trabajo final de máster.

1.4.2. Otros requisitos

Naturalmente, los requisitos anteriores no son los únicos que marcan el desarrollo del proyecto. Existen otras condiciones autoimpuestas que también ayudan a acotar el diseño, si bien su cumplimiento es más flexible.

El prototipo debe ser simple en su construcción, utilizando materiales y componentes que resulten fáciles de encontrar en comercios online o físicos. También se procurará que el diseño de la compostera y la elección de componentes ajenos al sistema de control sea flexible, pudiendo utilizar componentes distintos en caso de querer reproducirla.

Para crear el prototipo, se utilizarán componentes de bajo coste. Esto se aplica a la parte electrónica del proyecto, pero también al resto de elementos de la compostera. Así, se pretende que el producto final resulte asequible y reproducible por cualquiera sin implicar grandes desembolsos.

Por último, al no tener finalidad comercial, el prototipo creado será libre de ser reproducido, modificado o mejorado por cualquier persona que lo desee. Esto se aplica tanto al conjunto como a los componentes y códigos de programa. Se sigue así en la línea marcada por la filosofía del software y hardware libre utilizados en este proyecto.

2. DISEÑO DEL SISTEMA

En este punto se detallará el proceso seguido para diseñar el sistema de compostaje automático. Se explicará el diseño de la compostera y sus partes, la arquitectura del sistema de control y el proceso de selección de los componentes electrónicos.

2.1. Diseño de la compostera inteligente

Como se ha mencionado anteriormente, el diseño de la compostera debe ser simple y fácil de reproducir. Para ello, se utilizarán únicamente componentes comerciales y de fácil acceso. Se puede dividir el sistema en tres partes: la compostera, entendido como el propio contenedor donde se realiza el compostaje; el depósito de agua e instalación de riego y la central donde se aloja la electrónica del sistema de control. El sistema de aireación no se considera pieza a parte por estar dentro de la propia compostera.

Teniendo en cuenta que el sistema está orientado a un uso doméstico, las dimensiones de este tienen que ser comedidas, pudiéndose instalar en una terraza o jardín. Esto delimitará el tamaño de la compostera, así como el del depósito de agua cuya selección se detallan a continuación. Para tener una idea más completa sobre los elementos que forman el sistema, se puede consultar el esquema de la figura 4.

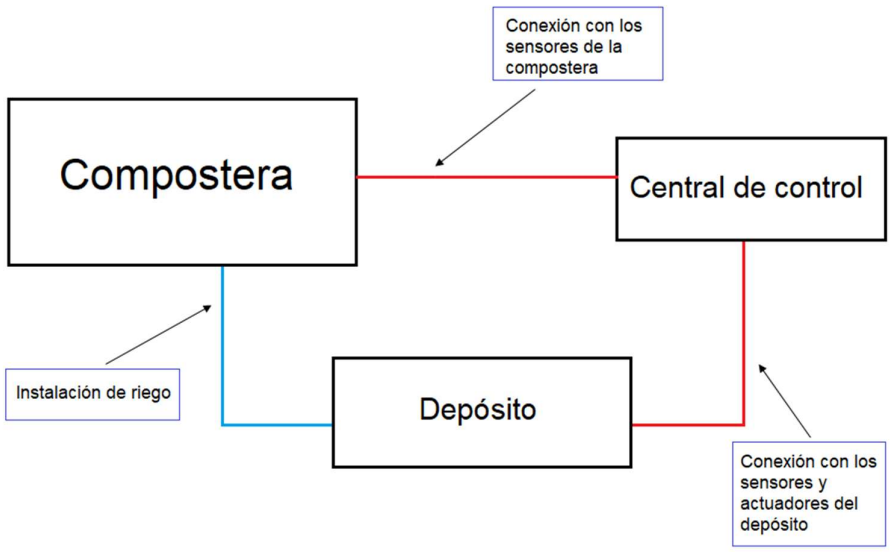


Figura 4: Esquema de las partes que componen el sistema. Producido por el autor.

2.1.1. Elección de la compostera

Existen muchas soluciones comerciales diseñadas para introducir los residuos orgánicos durante el proceso de compostaje. Suelen ser tanto de plástico como de madera y tener diferentes formas y volúmenes. Sin embargo, la condición de mantener el coste del sistema lo más bajo posible hace incompatible el uso de este tipo de productos.

El manual de compostaje del Ministerio de Medio Ambiente [7] da diversas ideas sobre cómo construir composteras en casa. La principal condición que menciona la guía es que el recipiente a utilizar permita el paso de aire al interior de la compostera para garantizar un proceso aeróbico y lo mantenga separado del suelo.

Una de las opciones más interesantes que se menciona en el manual consiste en construir una compostera a base de palés. A pesar de ser bastante económico, el gran tamaño de los palés lo hace incompatible con el uso en algunas viviendas. Otros diseños de menor tamaño incluyen cajas de plástico para frutas. Aunque resulta atractiva la posibilidad de dar una nueva vida a estas cajas, es difícil tener acceso a ellas por lo que se descarta.

Para cumplir con los requisitos de tamaño compacto y coste bajo, se utilizará en el prototipo otra de las posibilidades que ofrece la guía: Una maceta de grandes dimensiones. En concreto la compostera empleada en este prototipo será una maceta de polietileno de 30L con asas de orificios en su base y laterales.

Figura 5: Maceta de polietileno utilizada como compostera. Imagen obtenida de amazon.es

2.1.2. Sistema de aireación

Para el sistema de aireación se partió de un diseño complejo que implicaba todo un entramado de tubos por los que circulara el aire. La idea estaba basada en la primera versión del *Compost Professor*, un proyecto de Darian Johnson que persigue el mismo objetivo que este pero que se aproxima al problema desde una perspectiva distinta. [9]

Al estudiar esta solución, se dudó de la eficacia que pudiera tener usando únicamente ventiladores y sin utilizar un compresor. Además, los conductos de aire se podían atascar fácilmente y la humedad podría condensarse en su interior. Buscando un compresor de pequeño tamaño, se llegó a la solución final.

Para el sistema de aireación se utilizará una bomba de aire para acuario. En concreto, por su precio moderado y gran caudal se utilizará la Ireenuo Q7. Esta bomba tiene una potencia de 3W y dos salidas con un caudal de 150L/h cada una. Además, utiliza como difusor del aire dos piedras porosas que evitan problemas de atascos.

Utilizar una bomba para acuario comercial en lugar de un sistema más complejo también aporta flexibilidad al prototipo, pudiéndose cambiar por cualquier otro modelo de forma simple.

Figura 6: Bomba de aire para acuario Ireenuo Q7. Imagen obtenida de amazon.es

2.1.3. Sistema de riego

El sistema de riego consta de un depósito de agua, una bomba pequeña y tuberías para riego por goteo. Se decide utilizar un sistema de riego por goteo por ser más eficiente en el uso del agua y requerir de menos presión que otras soluciones.

El depósito utilizado será un cubo de polipropileno de 30L. Se escoge por su bajo coste y por tener una capacidad suficiente como para dar autonomía en el riego durante varios días. En la tapa del cubo se colocará un sensor que permita al sistema comprobar si el nivel de agua es suficiente como para regar. Cerca de la base se hará un agujero donde se colocará un racor para conectar la entrada de la bomba.

Se utilizarán un único gotero regulable de hasta 60L/h con capacidad de difusión para humedecer toda la superficie de la pila de compost por igual. El caudal máximo del gotero determina la elección de la bomba, que deberá ser capaz de aportar como mínimo 60L/h. Teniendo esto en cuenta, se elige una bomba sin escobillas con caudal de 500L/h y presión de 2,5m. La bomba trabaja a 12V DC y se puede conectar a un enchufe de 22V de alterna utilizando una fuente de alimentación.

Figura 7: Gotero difusor durante el riego. Imagen obtenida de leroymerlin.es

2.1.4. Central de control

Central de control es el término utilizado para referirse al alojamiento donde se asienta la electrónica de control de la compostera inteligente. Debe ser resistente al agua y polvo ya que se situará en el exterior junto al resto de componentes. De ella saldrán los cables de comunicación con los sensores de la compostera y depósito. Esta caja también incorporará el sensor de humedad y temperatura ambiente, que se situará en su exterior, pero adosado a la misma.

Para cumplir con estas condiciones además de la facilidad de montaje, la electrónica de control se alojará en una caja de conexión estanca con certificación IP55 y de dimensiones 170x200x85mm. Sobre esta caja se montarán empotrados los enchufes a los que conectar la bomba de agua y la de aire. La alimentación de estos enchufes dependerá de relés en el interior de la caja. Este método también resulta útil para sustituir la bomba y compresor de forma fácil si se desea utilizar el sistema de automatización en una compostera de distinto tamaño.

Figura 8: Caja de conexión estanca IP55. Imagen obtenida de leroymerlin.es

2.2. Arquitectura del sistema de control

El sistema de control está formado por una placa de desarrollo conectada a una serie de sensores y actuadores y un servidor local donde se almacena la información en una base de datos. La placa de desarrollo utilizada es una Wemos D1 R32, mientras que una Raspberry Pi actuará de servidor local y alojará la base de datos. Se entrará en detalle sobre los distintos componentes en sucesivos puntos de la memoria.

La función de la placa de desarrollo será realizar lecturas de los parámetros que afectan al proceso de compostaje. Estos son: Temperatura del compost, humedad y presencia de metano. En caso de que la humedad sea baja, el Wemos D1 R32 encenderá el riego por goteo. Si hay metano en el ambiente, activará la aireación.

Además de controlar la compostera, la placa de desarrollo se encargará de comprobar el nivel del depósito y medirá temperatura y humedad ambiente. El estado del depósito y las condiciones del ambiente se enviarán junto con los datos del compost a la base de datos en la Raspberry.

La Raspberry Pi recibirá la información y, tras darle el formato adecuado, la almacenará en una base de datos. La información almacenada se podrá consultar desde cualquier dispositivo conectado a la misma red local que el ESP y la Raspberry a través de un navegador web. Se explicará este proceso más adelante, si bien se incluye un pequeño esquema con algunos detalles más sobre la transmisión de los datos las en la figura 9.

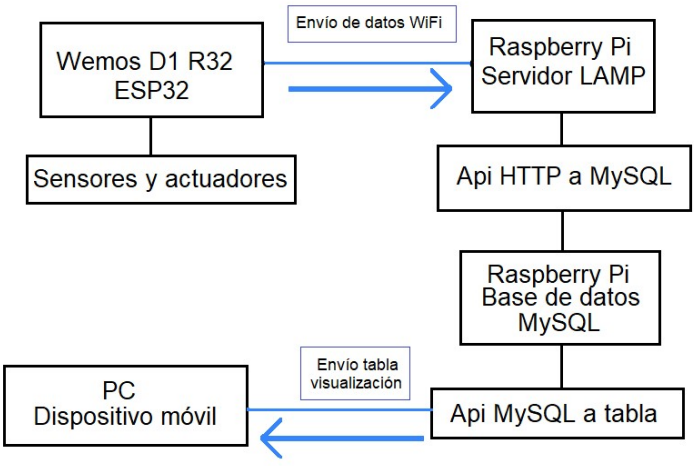


Figura 9: Esquema de la arquitectura del sistema de control. Producido por el autor

2.3. Diseño electrónico: Elección de componentes y funcionamiento

El siguiente punto trata la elección de los distintos componentes utilizados en el sistema de control. Se describirán los componentes utilizados y su funcionamiento, así como posibles consideraciones necesarias para su montaje. Se divide esta sección en tres grandes puntos: Plataforma ESP32, Raspberry Pi y sensores y actuadores.

2.3.1. Plataforma ESP32: Wemos D1 R32

Para este proyecto es necesario utilizar un procesador con capacidad de comunicación WiFi. Además, dado que este trabajo es una primera incursión en el mundo del Internet de las cosas, su programación debe ser lo más simple posible.

Uno de los a *System on a Chip* (SoC) más utilizados en proyectos IoT a nivel casero es el ESP32 de modo que se elegirá utilizar una placa de desarrollo que incorpore este módulo por la cantidad de recursos disponibles sobre el mismo. El ESP32 también comienza a utilizarse en aplicaciones industriales, aunque su novedad lo hace aún poco común. [10]

El ESP32 es un SoC con capacidad WiFi y Bluetooth diseñado por la compañía china Espressif y fabricado por TSMC. Es una evolución del ESP8266, también de Espressif, que añade nuevas funcionalidades y mejora el rendimiento general. Utiliza un procesador Tesilica Xtensa de doble núcleo de 32bit a 160MHz con SRAM de 448KB y memoria flash de 520KB. El módulo trabaja a 3,3V y tiene modos de ultra bajo consumo.

El procesador cuenta con 34 pines GPIO, 16 de los cuales poseen un conversor analógico digital de 12bits. Del mismo modo, existen 17 pines configurables como PWM por software, uno de ellos específico para motores. También posee otras interfaces para periféricos como I2C, SPI, CAN bus o UART entre otros. [11]

El ESP32 se monta en multitud de placas de desarrollo, la mayoría de ellas compatibles con la IDE de Arduino, lo que facilita su programación permitiendo el uso de librerías y funciones desarrolladas para Arduino. Muchas incluyen accesorios como pantallas OLED, baterías etc. Sin embargo, para este proyecto se decidió elegir una placa de desarrollo lo más parecida posible en su construcción a un Arduino clásico.

Teniendo en cuenta las condiciones anteriores, se elige utilizar la placa de desarrollo Wemos D1 R32. Esta placa tiene un factor de forma y distribución de pines iguales al del Arduino Uno, lo que la hace compatible con shields de accesorios para Arduino. La comunicación con el PC para su programación se puede hacer a través de puerto serie mediante conexión USB. También se puede usar el USB para alimentar la placa a 5V, ya que cuenta con la electrónica necesaria para reducir la tensión de alimentación del ESP32 a 3,3V. Esto resulta útil porque proporciona pines de alimentación a 5V y permite usar los pines GPIO con entradas digitales de hasta 5V, aunque el nivel alto se seguirá considerando a partir de 3,3V.

La placa de desarrollo Wemos D1 R32 es un diseño de Wemos con licencia de hardware libre, lo que permite a otros fabricantes reproducirla y modificarla para su distribución. Es el caso de AZ Delivery, que comercializa esta placa con el nombre ESP32 D1 R32. Se añade a continuación una figura con el pinout de la placa a utilizar.

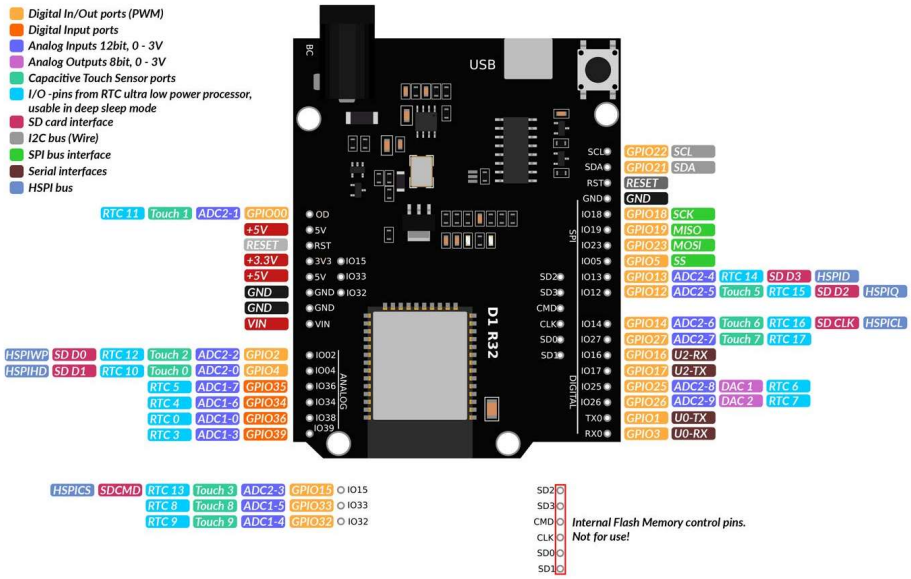


Figura 10: Pinout de la plaza ESP32 D1 R32. Imagen Obtenida de AZ Delivery.

2.3.2. Raspberry Pi: Servidor local

Una de las formas más fáciles de crear un servidor web con una base de datos donde poder almacenar información es utilizar la infraestructura LAMP. Estas siglas se refieren al conjunto de herramientas de software necesarias para crear el servidor, todas de código abierto. Linux como sistema operativo, Apache como el servidor HTTP, MySQL o MariaDB como gestor de la base de datos y PHP como el lenguaje de programación utilizado para desarrollo web.

Para instalar el servidor LAMP, es necesario por tanto un equipo con sistema operativo Linux. El servidor deberá estar conectado durante todo el proceso de compostaje, para poder así recibir los envíos del ESP32. Tener un ordenador convencional dedicado a ejercer de servidor LAMP es posible, pero resulta caro y, generalmente, poco práctico. Para solventar este problema, se decide utilizar una Raspberry Pi.

Se utilizará en concreto una Raspberry Pi 4 Model B. Como se ha explicado anteriormente, se trata de un ordenador de placa única, bajo coste y tamaño compacto. Utiliza sistema operativo Raspbian, una variante de Linux. Se alimenta mediante USB-C a 5V con lo que se puede usar cualquier cargador de móvil como fuente de alimentación.

La versión de utilizada de Raspberri Pi tiene procesador Broadcom BCM2711B0 quad-core Cortex-A72 de 1,5GHz y 4GB de RAM. El sistema operativo se instala en una tarjeta microSD que hace las veces de memoria. Posee conectividad inalámbrica WiFi y Bluetooth, puertos USB 2.0 y 3.0, conector de audio Jack, doble conexión micro HDMI además de 40 pines GPIO entre otros conectores para periféricos varios.

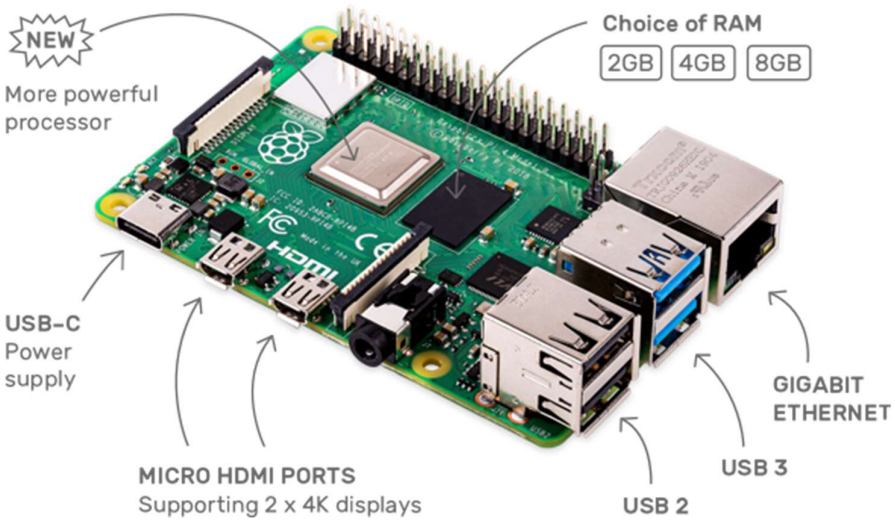


Figura 11: Raspberry Pi 4 model B. Obtenida de raspberrypi.org

2.3.3. Sensores y actuadores

En este subpunto se detallarán uno a uno los distintos sensores y actuadores utilizados para controlar el proceso de compostaje. Cada uno de los sensores monitorizará el estado de entre una y dos variables del proceso, mientras que los actuadores serán activados por el Wemos D1 R32 para regar y airear.

2.3.3.1. Sensor de humedad y temperatura ambiente: DHT11

Para poder apreciar las variaciones de temperatura en el compost, es necesario comparar la temperatura del compost con la temperatura ambiente. Del mismo modo, conocer la humedad del ambiente ayuda a entender si el compost se está secando por debido al proceso de fermentación de la materia orgánica o simplemente a la combinación de temperatura ambiente alta y humedad baja.

El sensor utilizado es un DHT11 que viene montado en un módulo KY-015 y está calibrado de fábrica para obtener tanto temperatura como humedad mediante una señal digital. Su bajo coste y consumo energético lo hace ideal para esta aplicación. Su operación es simple en Arduino utilizando la librería DHT que incluye funciones para configurar el sensor y obtener de forma rápida la temperatura y humedad medidas.

El módulo KY-015 sobre el que se monta el sensor, tiene 3 pines para alimentación, tierra y envío de datos mediante un protocolo de comunicación serie indeterminado. Funciona tanto a 3,3V como a 5V. Mide temperaturas de entre 0ºC y 50ºC con un error de ±2ºC y resolución de 1ºC y humedad relativa con un rango que va del 20% al 90% con un error de ±5%. [12]

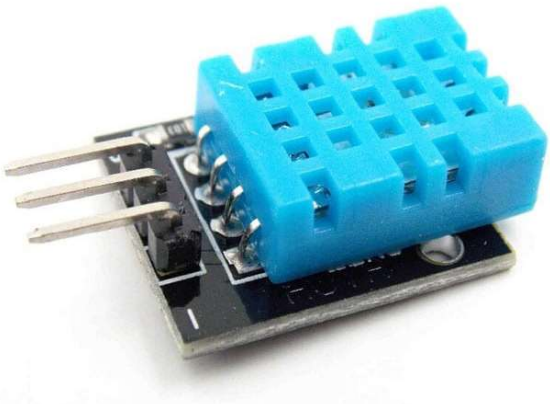


Figura 12: Sensor DHT11 sobre módulo KY-015. Imagen obtenida de uelectronics.com

2.3.3.2.Sensor de temperatura del compost: Sonda DS18B20

Para medir la temperatura del compost es necesario un sensor que cuente con un encapsulado que lo haga resistente a la humedad, al agua y a la suciedad presentes en la compostera. Uno de los sensores más utilizados en este tipo de aplicaciones es el DS18B20 por su precio reducido y su facilidad de uso. [13]

La sonda tiene tres cables, dos de ellos para alimentación y masa a entre 3,3V y 5V y el tercero para el envío de las temperaturas medidas. Utiliza el protocolo One Wire para el envío de datos, lo que permite utilizar varios sensores DS18B20 utilizando un único cable y por tanto un único pin de comunicación en la placa de desarrollo. Se utilizará la librería OneWire para gestionar la comunicación y la DallasTemperature para traducir los datos del sensor a grados centígrados. El sensor viene calibrado y puede medir temperaturas entre -55ºC y 125ºC con un error de ±0,5ºC en el rango de los -10 y 85ºC.

Figura 13: Sonda del sensor de temperatura DS18B20. Obtenido de aliexpress.com

2.3.3.3. Sensor de humedad del compost: YL-69

La opción más económica y simple de medir la humedad del compost es mediante un sensor cuya conductividad cambie según el porcentaje de agua que contenga. Una de las sondas más populares es la YL-69, utilizada junto a una tarjeta comparadora basada en LM393. Se puede alimentar tanto a 3,3V como a 5,5V y, según el fabricante, viene calibrado de fábrica. Las sondas diseñadas para enterrarse en el sustrato son sensibles a la corrosión y oxidación, pero su bajo coste hace que sean sustituibles sin demasiado perjuicio cuando su funcionamiento empiece a verse comprometido.

Este sensor permite hacer lecturas tanto analógicas como digitales. Las digitales cambian entre nivel alto y bajo según el nivel de humedad marcado mediante el ajuste de un potenciómetro. Las analógicas varían el valor de la tensión según la humedad del compost. El Wemos D1 R32 puede hacer lecturas analógicas desde la mayoría de sus pines con una resolución de 12bits, lo que traduce las entradas a valores entre 0 y 4095.

La calibración de fábrica está hecha para dar valor de máxima conductividad cuando la sonda está sumergida en agua y de mínima conductividad cuando la sonda está al aire. Para obtener el porcentaje de humedad del suelo se suele mapear los valores de la lectura analógica para hacer coincidir, en caso de tener una resolución de 12bits, el 100% con el valor 0 y el 0% con el valor 4095. Las pruebas realizadas con el sensor indican un valor de lectura analógica de 4095 cuando el sensor está al aire y de 370 cuando la sonda estaba totalmente sumergida en agua. Se decidió mapear entre estos dos valores para obtener la humedad del compost con mayor precisión.

Este método para obtener el porcentaje de humedad del compost suponiendo una relación directamente proporcional entre la conductividad y la cantidad de agua resulta bastante simplista. Obvias variables como la cantidad de sales en el terreno y agua, provocando errores medición relativamente grandes. La literatura consultada no asegura un valor concreto para esta aproximación lineal, si bien las estimaciones realizadas indican que no debería supera en exceso el ±10% de error.

I Setoyowati propone en el artículo *Preliminary design and soil moisture sensor YL-69 calibration for implementation of smart irrigation* [14] un método de calibración utilizando curvas polinómicas de orden 3 para interpretar las lecturas de conductividad. Con esto se consigue un error de entre el 2,92 y el 5%.

Hay que tener en cuenta que el compost tiene que estar a una humedad de entre el 40% y el 60%, como ya se ha explicado antes. Esto supone que incluso con el error máximo de la aproximación tradicionalmente utilizada, el proceso se podría encontrar dentro de los parámetros óptimos cuando se le marque mantener una humedad del 50%. Se fijará este valor en la programación del microcontrolador para mantener la humedad de forma segura y simplificar la programación asumiendo los posibles errores.

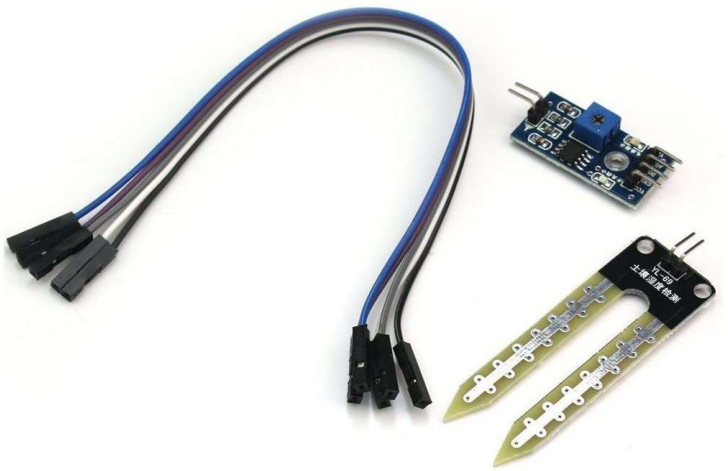


Figura 14: Sonda YL-69 y tarjeta comparadora LM393. Imagen obtenida de amazon.es

2.3.3.4. Sensor de metano MQ-4

La presencia de metano en el compost es crítica. Por eso se hace necesario utilizar un sensor capaz de detectarlo. El sensor se colocará fuera de la pila de compost, junto a la tapa de la compostera. El MQ-4 es el sensor elegido puesto que no solo es compatible con la placa de desarrollo D1 R32, sino que además tiene un precio bastante ajustado. Se trata de un sensor que mide cambios en la resistividad de un hilo conductor recubierto de dióxido de estaño, material cuya conductividad cambia en presencia de metano.

El sensor trabaja a una tensión de 5V y tiene salidas analógica y digital. A partir de la señal analógica, entre 0 y 5V, se puede calcular la concentración de metano en el aire. A mayor cantidad de metano, mejor será la conductividad del hilo de cobre y por tanto mayor el voltaje que devuelve el sensor por su salida analógica. La salida digital es de sensibilidad ajustable mediante un potenciómetro, devolviendo señal baja cuando detecta presencia de gas y alta cuando no.

El sensor mide concentraciones de metano entre 300 y 10000 ppm. Durante las pruebas del sensor, se observó que el led indicador que incorpora el módulo se encendía antes de recibir señal analógica de variación de la conductividad apreciable. Es por eso que se decidió ajustar mediante el potenciómetro el sensor a su máxima sensibilidad y utilizar su salida digital en lugar de la analógica. De esta forma, también se simplifica el código reaccionando ante la presencia de cualquier concentración de metano detectable por el sensor.

La salida digital es de 5V, en el límite de lo admisible por la D1 R32. Aunque debería ser soportada por la placa de desarrollo, la experiencia indica que esto puede dañar la placa. Por eso, será necesario utilizar un divisor de tensiones para no quemar el controlador. Para realizar las comprobaciones de presencia de metano, solo habrá que comprobar el estado del pin al que se conecta el sensor: Alto si no hay metano y bajo cuando lo hay.

Figura 15: Sensor de metano MQ4. Imagen obtenida de amazon.es

2.3.3.5. Sensor ultrasonidos HC-SR04: Nivel del depósito

Para comprobar si el depósito tiene agua suficiente para regar, es necesario conocer el nivel de agua en su interior. Para comprobarlo la solución más eficaz es medir la distancia entre la tapa del depósito y la lámina de agua.

Uno de los sensores más utilizados para medir distancias es el sensor de ultrasonidos HC-SR04. Este sensor permite medir distancias con agua y otros elementos no opacos. Su funcionamiento se basa en el envío de pulsos de ultrasonidos (A frecuencia de 40KHz) en la dirección a medir y la recepción del eco producido al rebotar las ondas de sonido con el objeto respecto al que se quiere conocer la distancia. Para calcular esta distancia solo habrá que aplicar una pequeña fórmula multiplicando por la velocidad del sonido el tiempo transcurrido entre el envío del pulso y su recepción.

El sensor HC-SR04 tiene un rango de medición entre 2cm y 400cm con una resolución de 0.3cm. Su precisión depende en realidad de la distancia a medir junto con la orientación de la superficie sobre la que rebota el sonido, entre otras cosas. El error que se pueda cometer en esta aplicación es asumible puesto que se programará un nivel de agua con suficiente margen como para no dañar la bomba evitando que absorba aire.

El precio es muy reducido y es totalmente compatible con la placa de desarrollo D1 R32. Tiene cuatro pines: Dos para la alimentación a 5V uno para el envío de pulsos de ultrasonidos y otro para la recepción del eco. No es necesario utilizar librerías específicas para su control, aunque se explicará su programación en los siguientes puntos de esta memoria.

Figura 16: Módulo sensor de ultrasonidos HC-SR04. Imagen obtenida de amazon.es

2.3.3.6. Actuadores: Módulo relé de 2 canales con optoacoplador.

Los actuadores utilizados para el sistema de riego y aireación ya se han seleccionado en el apartado *2.1. Diseño de la compostera inteligente*. Se utilizará una bomba de agua comercial junto a una bomba de aire para acuario. El uso de actuadores comerciales simplifica el mantenimiento, cambio o mejora de los sistemas de aireación y riego. Los elementos seleccionados como actuadores están preparados para conectarse a tomas de corriente alterna de 220V y 50Hz. Para abrir y cerrar circuitos de alterna mediante señales de continua es necesario un relé.

El módulo de relés utilizado tiene dos canales que se activan mediante señales de 5V. Las salidas digitales de la placa de desarrollo utilizada son de 3,3V de modo que será necesario utilizar un pequeño circuito para, mediante un transistor, abrir o cerrar la señal de 5V utilizando la salida digital de 3,3V del ESP32. Se tratará en mayor profundidad más adelante cuando se explique el esquema electrónico del prototipo.

En cuanto a sus características eléctricas y electrónicas, el módulo necesita alimentación a 5V en corriente continua por lo que posee dos pines dedicados a esto. Los otros dos pines disponibles serán para recibir la señal de activación de los dos canales del relé. Para mayor protección, cada relé cuenta con un optoacoplador que aísla de forma efectiva los dos circuitos. Los circuitos de alterna y continua soportan una tensión de 250V y 30V respectivamente y una corriente de 10A ambos. El relé cuenta con contactos tanto normalmente abiertos como normalmente cerrados.

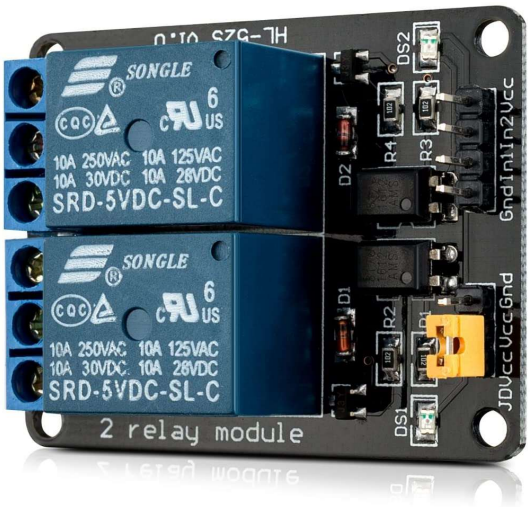


Figura 17: Módulo de relés dos canales 5V. Imagen obtenida de amazon.es

3. SISTEMA DE CONTROL

En este punto se explicará el funcionamiento del sistema de control. El proceso de lectura de datos desde los sensores, la toma de medidas correctivas en caso de ser necesario y el envío de las condiciones del sistema a la base de datos para poder ser consultado por el usuario. En definitiva, se profundizará en la arquitectura del sistema y se explicará el proceso necesario para crear el programa de control utilizado.

3.1. Funcionamiento del sistema de control

Como ya se ha explicado en el apartado *2.2. Arquitectura del sistema de control*, este sistema está formado por una placa de desarrollo con módulo ESP32, una Raspberry Pi, donde está alojado el servidor y base de datos y el conjunto de sensores y actuadores que se detallan en el apartado *2.3.3. Sensores y actuadores*. Como apunte, es importante recordar también que las variables medidas son la temperatura y humedad ambiente, la temperatura y humedad del compost, el nivel del depósito de riego y la presencia de metano en la compostera.

La placa de desarrollo Wemos D1 R32 está programada para realizar lecturas de los diferentes sensores una vez cada 30 minutos. Se decidió utilizar este tiempo entre mediciones porque, si bien el proceso de compostaje es lento, las condiciones del entorno pueden hacer que parámetros como la humedad del compost se vean alterados en poco tiempo situándose fuera de los niveles óptimos para el proceso. Este intervalo también es útil para captar mejor la variación de los datos de temperatura y humedad ambiente, así como para asegurar que el compost no pasa más de media hora fuera de los parámetros establecidos en caso de que el sistema de riego o aireación no haya actuado durante el tiempo suficiente.

Los datos medidos se envían a la Raspberry Pi para almacenarse en la base de datos. Como se ha comentado anteriormente se programará la Raspberry para ejercer de servidor LAMP con una base de datos MySQL. El proceso de envío de datos comienza con una petición HTTP Post al servidor por parte del ESP32. Esta petición contiene en un formato estandarizado y entendible por el servidor, los valores que definen el estado del compostaje. La petición es recibida por una api programada en PHP que ordena y

da el formato adecuado a los datos para ser almacenados en la base de datos MySQL. Una vez guardados, los datos podrán consultarse desde el navegador de cualquier equipo conectado a la red WiFi local. De nuevo interviene en el proceso una api programada en PHP que extrae la información de la base de datos y le da forma de tabla para mostrarla al usuario cuando se recibe petición de visualización.

A la vez que se miden los valores que determinan el estado del proceso de compostaje, el microcontrolador comprueba que la humedad y metano están a niveles correctos. Si la humedad es baja, se encenderá el riego durante dos minutos. Si hay metano en el ambiente se hará lo mismo con el compresor.

3.2. Diagrama de flujo

El diagrama de flujo es una herramienta que ayuda a estructurar el programa antes de escribir el código en el lenguaje de programación correspondiente. En el *Anexo I* de esta memoria se encuentra el diagrama de flujo del programa ejecutado por el microcontrolador. En él se explica de forma gráfica y simplificada el proceso que sigue el programa de control del proceso de compostaje. Su alcance se limita a los procesos que lleva a cabo la placa de desarrollo, sin entrar en el funcionamiento del servidor LAMP que se explicará más adelante.

Lo primero que hace el programa es incluir las librerías necesarias para el uso de funciones específicas y declarar las variables globales que intervienen a lo largo del programa. A continuación, se inicializan los sensores y las variables que así lo requieran. Acto seguido, se conectará el ESP32 a la red WiFi definida en el programa. Todas estas tareas se llevan a cabo antes o dentro del *setup* del programa Arduino y se ejecutan una única vez.

Justo después, el programa entra en el bucle de control o *loop,* que se ejecuta de forma infinita una y otra vez. Se toma tiempo para actualizar el tiempo actual. Si es la primera vez que se ejecuta el programa o si la última lectura se realizó hace más tiempo que el marcado por el intervalo de medición programado, el microcontrolador leerá uno a uno la señal de los diferentes sensores y actualizará el tiempo de última lectura.

Las variables a monitorizar que no influyen en el funcionamiento del programa se miden y guardan para su envío. La humedad del compost, metano en la compostera y nivel del depósito no solo se miden, sino que se comparan con los valores de consigna para marcar la necesidad de riego y aireación o para marcar la imposibilidad de regar en caso de que el nivel del depósito no sea adecuado.

Tras la lectura, se comprueba la conexión WiFi y, en caso de estar conectado, se preparan los datos para el envío y se realiza la petición HTTP Post al servidor. El tratamiento de esta petición una vez llega al servidor se detallará más adelante. Por el momento bastará con decir que terminan almacenados en una base de datos MySQL.

Realizado el envío de datos, el programa vuelve a actualizar el tiempo actual y comprueba la necesidad de medición. Cuando aún no ha pasado el intervalo de medición, el diagrama de flujo nos lleva a una nueva sección en la que se comprueba la necesidad de riego y aireación.

Si la marca de riego indica que es necesario activar la bomba, se comprobará la disponibilidad de agua en el depósito. Al tratarse de un bucle infinito, pasaremos por este punto muchas veces durante el periodo de riego. Esta comprobación nos permitirá apagar la bomba cuando el nivel baje del mínimo durante el riego. A continuación, se comprueba el tiempo de riego, apagando la bomba si se ha superado el intervalo definido. En caso de no superarlo, se enviará señal de activación al relé de la bomba. Si la bomba no se había encendido previamente, habrá que marcar el encendido y tomar el tiempo actual como tiempo de inicio de riego para poder compararlo posteriormente.

La rutina de aireación es muy similar a la de riego y actúa tras esta. Comprobada la marca de necesidad de aireación, se pasa a comprobar si el tiempo de encendido del compresor supera el establecido. Si no se supera, se marca el compresor como encendido, se toma el tiempo y se activa el relé. Por seguridad se comprobará que la bomba de agua no está activada. Esto ayuda a evitar salpicaduras y mal funcionamiento del sistema en general. En caso de estarlo, se apagará el compresor y se tomará de nuevo tiempo marcando el compresor como apagado. Si se ha superado el tiempo de airear se procederá del mismo modo, pero además se desactivará el marcador de necesidad de aireación.

3.3. Software de control

El diagrama de flujo del punto anterior resulta útil para estructurar el programa, pero su lectura no es unívoca, pudiendo generar diversos códigos a partir del mismo. Para explicar las funciones que no quedan suficientemente detalladas en el diagrama, se dividirá el código en partes y se explicará en este punto. Además, es posible leer el código completo consultando el Anexo II de la memoria.

Para introducir la explicación de este punto, se resumirá al lector de forma básica el flujograma expuesto en el punto anterior.

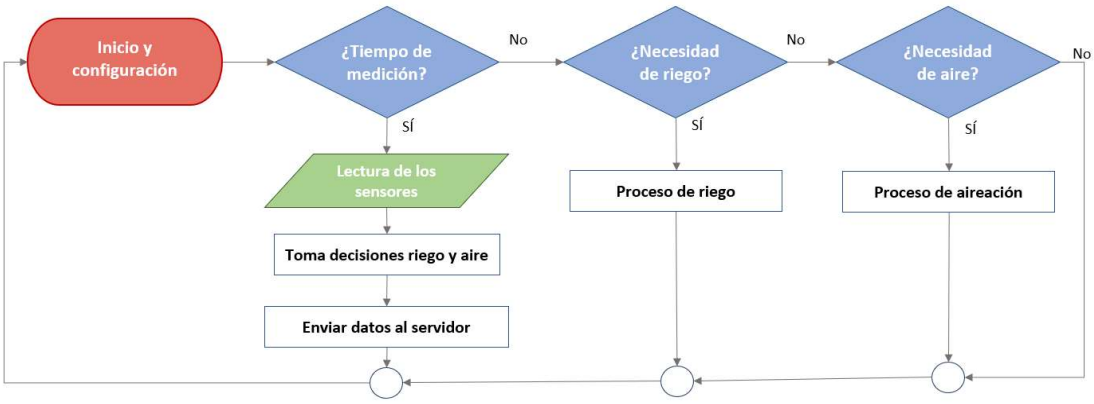


Figura 18: Resumen del diagrama de flujo. Imagen producida por el autor.

3.3.1. Variables globales: Datos a enviar, marcadores y tiempos

El programa de control utiliza una serie de variables globales declaradas al principio del código. Estas son susceptibles de ser utilizadas por cualquier función del programa. Las variables globales utilizadas se pueden agrupar en tres categorías:

- **Marcadores**: Son variables de tipo Booleano. Esto es, que solo pueden tener dos valores: true o false, uno o cero. Los marcadores nos servirán para indicar la necesidad de riego o aireación, así como para señalizar si el nivel del depósito es suficiente para regar y si el compresor o la bomba se encuentran encendidos.

- **Datos a enviar**: Son de tipo String, o cadena de caracteres. Contienen la información que se enviará a la base de datos: Temperatura y humedad ambiente, temperatura y humedad del compost, metano en el compost y nivel del depósito. Estas dos últimas guardadas como OK o NOK según estén dentro de parámetros o no. El uso de cadenas de caracteres para almacenar también valores numéricos está justificado con la forma en la que se construye la petición HTTP Post con la que se realizará el envío al servidor. Se entrará en detalle en los siguientes puntos.

- **Tiempos:** Las variables de tiempo son de tipo Unsigned Long, valores numéricos de hasta 32bits sin signo, lo que permite almacenar valores entre 0 y 2^{32}-1. El tiempo se mide en milisegundos, por lo que esto da un máximo de algo más de 49 días medidos por cada variable. Llevar un registro del tiempo a nivel interno es esencial para controlar el proceso sin tomar datos de manera innecesaria o hacer trabajar a los actuadores durante demasiado tiempo. El control de los tiempos entre mediciones, intervalo de riego e intervalo de aireación se explicarán en el siguiente punto de esta memoria.

3.3.2. Temporizadores: Intervalos de medición, riego y aireación

El registro de los tiempos es crucial para coordinar las distintas tareas que lleva a cabo el programa. Los microcontroladores son susceptibles de verse atrapados cuando se utilizan bucles *for* o *while* para repetir tareas que se prolongan en el tiempo. Del mismo modo, usar en exceso las interrupciones o *delays* puede provocar un mal funcionamiento del procesador, en especial cuando interfieren con la conexión Wifi, por lo que en su lugar se utilizarán temporizadores.

La programación de temporizadores en Arduino se basa en el registro de los tiempos mediante la función *millis()*, que devuelve el tiempo medido desde la puesta en marcha del microcontrolador hasta el momento en que se llama. En cada vuelta del bucle de control, se comienza por actualizar la variable *currentMillis* con el tiempo actual. Al comenzar cada uno de los procesos, se tomarán los tiempos de inicio, que se almacenan en las variables *previousMillis_Sensors, previousMillis_Water* y *previousMillis_Air.*

Para repetir un proceso cada cierto tiempo, habrá que comprobar si el tiempo actual menos el tiempo registrado la última vez que se realizó la tarea supera el intervalo marcado. Este es el caso de la lectura de los sensores, que se repite cada media hora. Dentro del bucle de control, se comprueba si el tiempo actual menos el tiempo en que se midió por última vez las variables es mayor o igual al intervalo de 30min marcado. Al comenzar las tareas de medir y enviar, se actualizará el tiempo de última medición para así no volver a entrar en esa sección hasta que no pase media hora.

Para realizar tareas durante un periodo de tiempo determinado, habrá que seguir el mismo principio, solo que con una pequeña variación. Al activar por primera vez la bomba de riego o la de aire, se registrará el tiempo actual como el momento en que se enciende y se seguirá con las tareas correspondientes: Activar el relé, comprobar que hay suficiente agua en el depósito etc.

La siguiente vez que se ejecuta el código se comprobará que el tiempo actual menos el tiempo de activación es menor que el intervalo definido para el riego o aire. Si es así, la bomba seguirá activada y no se actualizará el tiempo de activación. En el momento en que pase el periodo de riego o aireación, se desactivará el relé y se hará cesar la marca que provocó el proceso. Además, se actualizará el tiempo de forma que la próxima vez que se compruebe la necesidad de actuar la resta de tiempos cumpla la condición para poder encender la bomba correspondiente.

El tiempo de riego y aireación es el mismo: 2 minutos. Esto se justifica con los caudales de riego y aireación. El caudal de riego es el caudal del gotero utilizado: 60L/h. En dos minutos equivale a 2L vertidos sobre el compost, cantidad suficiente para humedecerlo sin mojarlo en exceso. Del mismo modo, la bomba de aire es capaz de introducir 300L/h en la pila de compost, lo que equivale a 10L en los dos minutos de aireación. Se considera suficiente como para inyectar oxígeno al compost y desplazar el metano.

```
//CICLO DE RIEGO
if (Flag_Water == true && currentMillis - previousMillis_Water <= interval_WaterAir ) {

  if (Flag_Pump == false) { // Si la bomba no está encendida aun
    previousMillis_Water = millis();  //Guardo tiempo de puesta en marcha
    Serial.println("Riego en marcha"); //Mostramos mensaje de que se ha activado riego
    Flag_Pump = true;
  }
```

Figura 19: Fragmento del código donde se efectúan las comprobaciones de tiempo de riego. Imagen producida por el autor

3.3.3. Lectura de los datos del compost

Los datos del compost se miden de dos formas distintas. Por un lado, están los medidos directamente mediante funciones específicas de librerías y que coinciden con los datos a monitorizar, sobre los que no actúa el programa. De otra parte, están los datos medidos utilizando funciones propias creadas para ello. Estas funciones sirven para medir la humedad del compost, la presencia de metano y el nivel de agua en el depósito. No solo registran los datos, sino que también los comparan con los parámetros definidos como adecuados e indican la necesidad de actuar mediante los marcadores.

Los datos sobre los que no se puede actuar, pero necesarios para interpretar el proceso son: la temperatura del compost, la humedad del compost y la temperatura y humedad ambiente. Se leen cada media hora usando funciones de librerías y se guardan en cadenas de caracteres para preparar posteriormente preparar el envío.

La función para medir el metano es simple y solo se utiliza en el ciclo de lectura de sensores cada 30 minutos. Al utilizar la salida digital del sensor, lo único que deberá hacer el programa es comprobar si el pin del microcontrolador está a nivel alto o bajo. En caso de nivel alto, no habrá metano y se marcará como false la marca de aire. En caso de nivel bajo, habrá metano y por tanto se marcará como true la marca de aire para proceder a airear el compost. La función devolverá un string con un OK si no existe metano y un NOK si hubiera presencia del gas en la compostera. Estos son los valores que se almacenarán en la base de datos para ser consultados por el usuario.

En el caso de la humedad, la función se vuelve algo más compleja. De nuevo la humedad solo se mide cada media hora, pero al ser una señal analógica, se trata de forma distinta. Para empezar, el higrómetro solo se alimentará durante el tiempo que dure la medición de la humedad, por lo que el primer paso será poner el pin de alimentación a nivel alto. Esto ayuda a prolongar la vida útil del sensor ya que reduce significativamente la corrosión de la sonda. En segundo lugar, se hará una lectura analógica del pin de entrada del sensor. Esta señal estará entre 370 y 4095 como se explica en el apartado *2.3.3.3* de esta memoria. El valor obtenido se mapeará para llevarlo a un valor entre cero y cien correspondiente con el porcentaje de humedad del compost. Se comprueba si la humedad es inferior al 50% y se marca el riego como necesario o no según proceda. A continuación, se transformará el dato de porcentaje en una cadena de caracteres que será la variable devuelta por la función cuando se llama.

El caso de la medición del nivel del depósito es distinto, ya que esta función es de tipo void y por tanto no devuelve ningún valor cuando se le llama. Si los dos casos anteriores se programaban como funciones externas al loop para poder simplificar la lectura del programa, en el caso de la lectura del nivel se añade una razón más: Evitar repetir innecesariamente partes del código en diferentes secciones del programa. El nivel del depósito no solo se comprueba cada media hora, sino que se mide durante el tiempo de riego para desactivarlo si no hubiera agua suficiente evitando dañar la bomba.

El proceso de comprobación de nivel del depósito comienza con el envío de un pulso de ultrasonidos de 10 microsegundos en dirección a la superficie del agua. A continuación, el pin al que se conecta la salida echo del sensor se abrirá en a recibir datos y permanecerá abierto hasta que llegue el pulso de ultrasonidos rebotado. Esto se consigue mediante la función *pulseIn()*, que devuelve el tiempo en microsegundos que tarda un pin determinado en cambiar de estado.

Para obtener la distancia en centímetros con la lámina de agua, solo habrá que multiplicar el tiempo medido en microsegundos por la velocidad del sonido en metros por segundo y dividirlo entre dos. Se divide entre dos porque hay que tener en cuenta que el eco recibido ha recorrido la distancia hasta la lámina de agua dos veces, una durante el envío y otra tras rebotar. Se comprobará que la distancia obtenida es menor de 25cm, marcando el nivel como NOK si fuera mayor. Al medir el nivel desde arriba, hay que tener en cuenta que distancias menores implican más cantidad de agua en el depósito. El límite de 25cm corresponde a la distancia entre el sensor y la lámina de agua cuando esta está aproximadamente 2cm por encima de la toma de agua de la bomba.

```
//FUNCIÓN PARA MEDIR NIVEL DEL AGUA
void Levelmeasure() {
  digitalWrite(TRIGGER_pin, LOW);
  delayMicroseconds(5); //Apagar trigger 5us para evitar ecos
  digitalWrite(TRIGGER_pin, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIGGER_pin, LOW); //Encendemos trigger durante 10 us

  long duration = pulseIn(ECHO_pin, HIGH); //Lee el echo y devuelve el tiempo de vuelta del sonido en us
  float distance = duration * SOUND_VEL / 2;

  if (distance >= 25) { //Si la distancia es mayor a 25cm el nivel de agua es bajo
    Flag_Level = false; //Marca nivel como NOK
  }
  else {
    Flag_Level = true; //Marca nivel como OK
  }
} //Final de la función Levelmeasure
```

Figura 20: Fragmento del código correspondiente a la función para medir el nivel del depósito. Producida por el autor.

3.3.4. Envío de datos al servidor

El envío de los datos medidos al servidor es una de las partes fundamentales de este trabajo y, posiblemente, la que más novedosa resulte respecto a los conocimientos previos de programación adquiridos en el máster. En puntos sucesivos de la memoria se explicará la configuración del servidor y base de datos, pero antes es importante hablar de cómo funciona la remisión desde el microcontrolador.

Los envíos se realizan mediante peticiones HTTP Post. Las peticiones en el protocolo HTTP pueden realizarse de diversas formas según su semántica. Cada tipo de petición tiene unas características distintivas que la hacen adecuada para un cierto tipo de acción a realizar con un recurso determinado. Las peticiones de tipo Post son unas de las más útiles para el envío de información a un servidor con el objetivo de crear o actualizar un recurso contenido en él. Este tipo de peticiones se utilizan para enviar un recurso específico al servidor provocando un cambio en su estado o en efectos secundarios del mismo. Son de especial utilidad para el envío de información del compost al servidor puesto que no existen limitaciones sobre la longitud de los datos a enviar. [15] [16]

Tras la lectura de los sensores partimos de los datos a enviar almacenados en variables tipo string. Se llama a continuación a la función específica de envío de datos, de tipo void y fuera del bucle de control únicamente para facilitar la lectura del programa. La función comienza por configurar el ESP32 como cliente http y de iniciar la conexión en modo cliente.

Configurado el ESP32 es momento de confeccionar la petición. Se empieza por definir la cabecera sobre el contenido a enviar, que define cómo se ordenan los datos en el envío. En este caso, se define que los datos irán en duplas de clave y valor donde la clave indica a qué corresponde el valor y comienza con un símbolo & terminando con un igual seguido del dato a enviar. La petición tiene formato de cadena de caracteres, por lo que el siguiente paso será unir las string que contienen los datos en una. Se unen añadiendo antes de cada dato su clave identificativa con la forma previamente explicada.

Por último, se realiza la petición HTTP Post enviando al servidor la cadena creada. La respuesta del servidor se guardará para imprimirla como mensaje de depuración. A continuación, se cerrará la conexión con el servidor terminando así el proceso de envío.

3.4. Configuración del servidor LAMP y base de datos

Como se ha explicado en secciones anteriores de esta memoria, los datos recogidos por el ESP32 se guardan en una base de datos MySQL desde donde pueden consultarse por el usuario del sistema de compostaje automático. Esta base de datos se encuentra dentro de la infraestructura LAMP implementada sobre la Raspberry Pi que actúa de servidor.

Recordemos también que un servidor LAMP implica una combinación de herramientas como son un sistema operativo Linux, un servidor HTP Apache, el gestor de bases de datos MariaDB o MySQL y la flexibilidad de crear otros elementos programados en PHP. En este punto se explicarán por orden los pasos a seguir para instalar y utilizar estas herramientas con el objetivo de crear la infraestructura necesaria para recibir los envíos del ESP32, almacenar la información contenida en una base de datos y visualizarla desde el navegador de cualquier dispositivo conectado a la red local.

Se decide no abrir la consulta a dispositivos fuera de la red local doméstica para evitar problemas de seguridad. La configuración de los distintos elementos sería igual solo que con pequeñas variaciones y abriendo los puertos del router correspondientes.

3.4.1. Instalación de la infraestructura LAMP

El primer paso para crear el servidor LAMP es instalar en la Raspberry Pi todos los elementos que lo forman. Se comenzará por actualizar el sistema operativo de la Raspberry Pi para asegurarse de que tiene la última versión de todos sus paquetes. A continuación, se instalarán los distintos programas en el siguiente orden:

1) **Apache2**: Es el software de servidor web más utilizado. Se encargará de gestionar las peticiones de acceso a páginas web generando el documento requerido en formato .php o .html entre otros. Una vez instalado, será posible acceder al servidor web creado desde cualquier dispositivo conectado a la red local introduciendo la IP de la Raspberry en el navegador.

2) **PHP:** PHP son las siglas de Hypertext Processor, un lenguaje de programación utilizado para desarrollar aplicaciones web dinámicas. Con el se generarán los recursos del servidor que guardarán o mostrarán información en la base de datos.

3) **MySQL:** Se conoce por este nombre a un tipo de bases de datos relacionales de código abierto. Para crear bases de datos MySQL se instalará en la Raspberry el servidor MariaDB y el software de control de bases de datos PHP-MySQL.

4) **phpMyAdmin**: Es una herramienta programada en php y que sirve para gestionar bases de datos MySQL de forma simple mediante una interfaz web. Para acceder solo hay que conectarse desde el navegador de cualquier equipo conectado a la red WiFi del servidor introduciendo como URL http://IPdeRaspberry/phpmyadmin.

Con los cuatro pasos anteriores ya está lista la Raspberry Pi para ejercer de servidor LAMP y almacenar bases de datos MySQL. El paso siguiente será configurar la infraestructura creada para la aplicación específica de almacenar y mostrar los datos del proceso de compostaje.

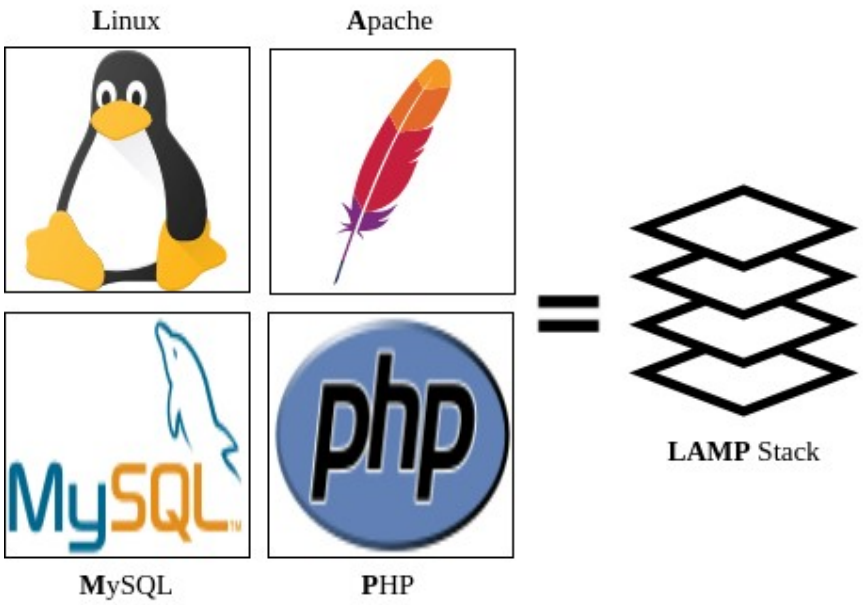


Figura 21: Esquema de los componentes de un servidor LAMP. Imagen obtenida de section.io

3.4.2. Creación de la base de datos MySQL

Existen varias formas de crear bases de datos en MySQL. Utilizando phpMyAdmin resulta bastante simple incluso para usuarios sin experiencia previa utilizando bases SQL. Como se ha explicado en el punto anterior, para acceder a phpMyAdmin habrá que introducir en el navegador la dirección IP de la Raspberry seguida de barra y phpmyadmin.

Una vez dentro de la aplicación, habrá que iniciar sesión mediante las credenciales proporcionadas durante la instalación de phpMyAdmin. Pulsando en el botón “Bases de Datos” se rellenan los campos del formulario “crear base de datos” con el nombre de la base de datos y el cotejamiento utilizado. En este caso, se llamará a la base de datos esp_data y utilizará un cotejamiento de tipo utf8mb4_general_ci. Tras rellenar el formulario, se pulsa el botón crear y la base de datos estará lista.

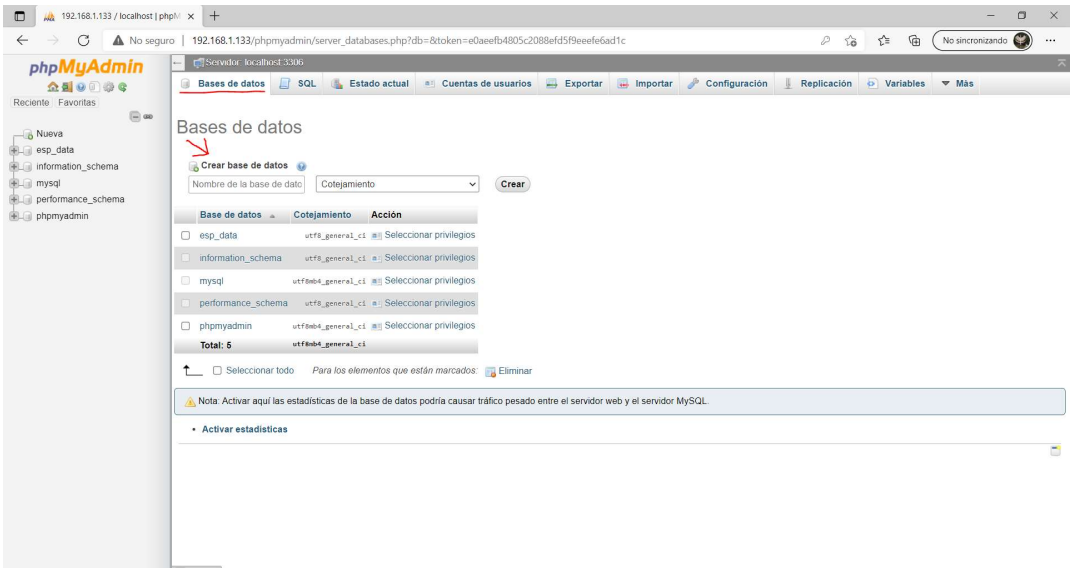


Figura 22: Menú de creación de la base de datos. Imagen producida por el autor.

Creada la base de datos, es necesario crear una tabla SQL que ordene y estructure la información. Como paso previo, habrá que definir la información que debe almacenar la base de datos y mostrar la tabla, así como el orden en que se muestra. Se puede ver una preforma de la tabla en la siguiente figura, donde también se indican el tipo de datos a guardar y tamaño. Se marcan en salmón los más relevantes para el seguimiento del proceso por parte del usuario.

Ciclo	Temp Ambiente	Hum Ambiente	Temp Compost	Hum Compost	CH4 Compost	Nivel Deposito	Tiempo
Int	Varchar	Varchar	Varchar	Varchar	Varchar	Varchar	Time
6	10	10	10	10	5	5	NA

Figura 23: Tabla preforma de la base de datos. Producida por el autor.

Para aquellos sin experiencia con el lenguaje SQL, la aplicación phpMyAdmin ofrece la posibilidad de crear tablas mediante una interfaz gráfica. Decididos los datos que debe contener, se crea la data CompostData. En el formulario de creación de tablas de phpMyAdmin habrá que introducir el nombre de la tabla y sus columnas. Se puede ver el formulario en la figura 23.

A cada columna se le asigna un nombre, un tipo de dato y un tamaño máximo, manteniendo siempre el orden de introducción. El tamaño máximo de las variables Varchar se mide en caracteres. La variable de ciclo será autoincremental y se añadirá automáticamente delante de los datos. La fecha y hora será un timestamp actualizado con el tiempo actual en cada llegada de datos.

Una vez configurada, ya está creada la tabla y se puede observar desde phpMyAdmin y exportar en distintos formatos. Naturalmente, la tabla aún se encuentra vacía por lo que será necesario rellenarla. En el siguiente punto de esta memoria se explicará cómo.

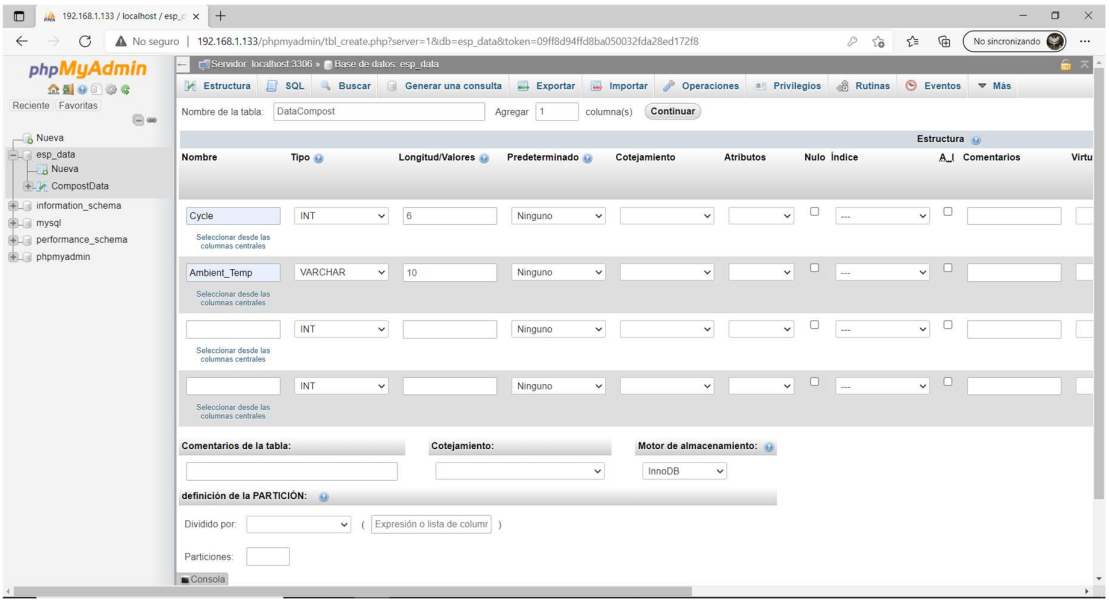


Figura 24: Formulario de creación de tabla SQL. Imagen producida por el autor.

3.4.3. API Post-ESP-Data: Almacenamiento en base de datos

Una vez creada la base de datos y tabla, hay que insertar información en esta para ser guardada. Para hacerlo, se crea una API que se encargará de recibir la información del ESP32 y añadirla a la base de datos MySQL. Una API es una interfaz de programación de aplicaciones. Algo así como un programa que ejerce de intermediario ejecutando una serie de subrutinas que preparan la información antes de guardarla.

La API se creará como un script PHP, basándose en el script original de Rui Santos en su artículo *ESP32/ESP8266 Publish Data to Raspberry Pi LAMP Server* [17]. El código completo se encuentra en el Anexo III de la memoria. Para crear el script se utilizará el editor Geany, instalado por defecto en la Raspberry. El archivo debe guardarse con extensión .php en la carpeta /var/www/html donde se guardan los recursos del servidor.

El script comienza definiendo la base de datos a la que queremos añadir la información y el usuario y contraseña de esta. Define también la clave de la API, que debe coincidir con la clave al principio de los datos enviados por el ESP32. A continuación se crea la estructura de los datos a recibir.

El siguiente punto es el más importante del script. Comprueba que la petición que llega al servidor es del tipo HTTP Post y que contiene la clave adecuada. A continuación, sustituye los valores en la estructura previamente creada comprobando uno a uno su etiqueta y valor. Se utiliza la función test_input para limpiar cada fragmento de la petición post y darle el formato adecuado para almacenarse en la base de datos.

Definida la estructura, se crea la conexión con la base de datos MySQL. Una vez conectado, se envía la información por orden a la base de datos mediante el comando "INSERT INTO CompostData" seguido del nombre de las variables por orden y de los valores almacenados en el mismo orden.

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $api_key = test_input($_POST["api_key"]);
    if($api_key == $api_key_value) {
        $Ambient_Temp = test_input($_POST["Ambient_Temp"]);
        $Ambient_Humid = test_input($_POST["Ambient_Humid"]);
        $Compost_Temp= test_input($_POST["Compost_Temp"]);
        $Compost_Moist= test_input($_POST["Compost_Moist"]);
        $Compost_CH4= test_input($_POST["Compost_CH4"]);
        $Deposit_Level= test_input($_POST["Deposit_Level"]);
```

Figura 25: Fragmento del código que da forma a los datos a almacenar. Imagen producido por el autor

3.4.4. API ESP Data: Visualización de los datos

Una vez introducidos los datos del compost en la base de datos, pueden ser consultados y descargados desde phpMyAdmin. Esto resulta útil cuando se quieren obtener estadísticas detalladas, pero no es práctico si se quiere revisar el estado del compost de manera rápida. Para solucionarlo, se crea una nueva API en PHP que muestre la información en forma de tabla cuando se consulta desde cualquier navegador web de un dispositivo conectado a la red WiFi del servidor.

La API se llama ESP Data y de nuevo está basada en un script de Rui Santos desarrollado para una aplicación similar [17]. Se puede consultar el código completo en el Anexo IV de la memoria. Al igual que la API anterior, también hay que guardar el script en la ruta /var/www/html de la Raspberry

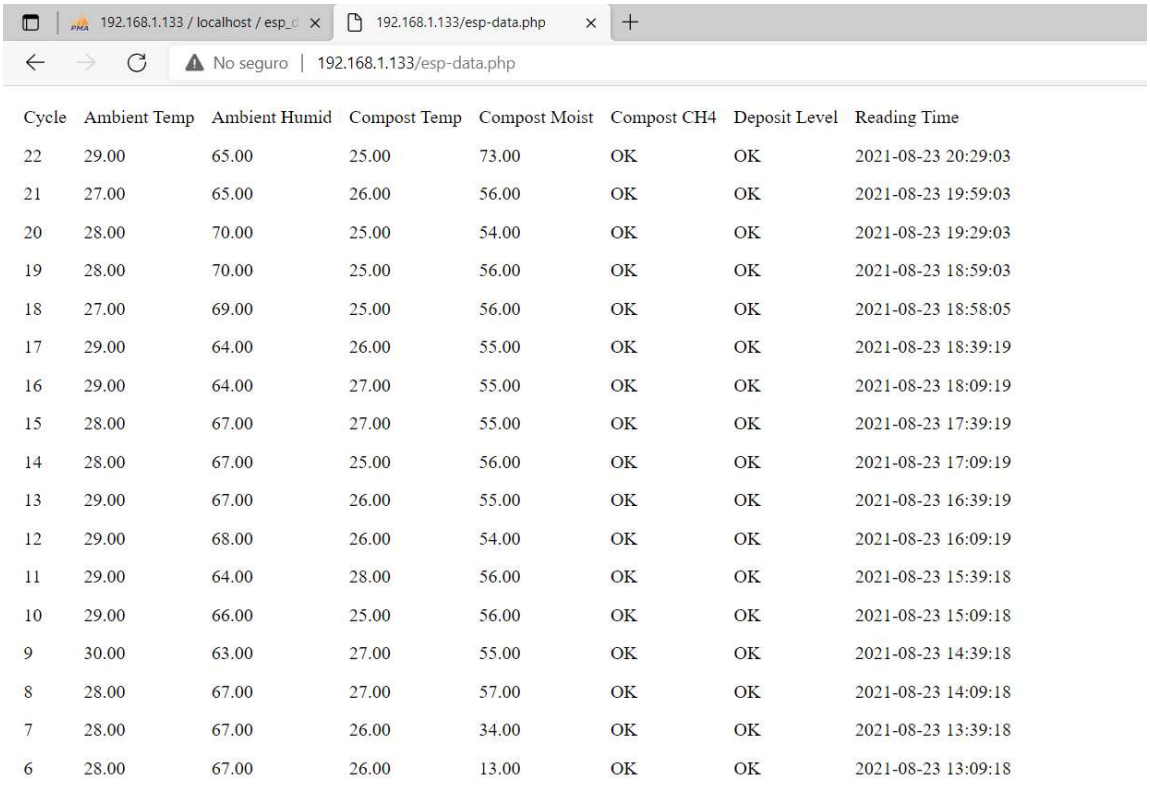
El funcionamiento de esta API es más sencillo que el de la anterior. Se comienza definiendo el servidor, base de datos y toda la información necesaria para realizar las conexiones. A continuación, el script conecta con la base de datos MySQL. Después se utiliza el comando “SELECT” seguido del nombre de las variables a visualizar, de la base de datos de la cual se quiere extraer y del orden en que se quieren mostrar. Se mostrarán los datos de más reciente a más antiguo.

Una vez configurados los parámetros anteriores, se creará la tabla que mostrará la API en el navegador. Se separarán los valores de la tabla en columnas con el nombre de la variable en el encabezado. Definida la tabla, se van insertando los datos en filas manteniendo el orden definido anteriormente. Este proceso se lleva a cabo mediante en el código de la siguiente figura.

```
if ($result = $conn->query($sql)) {
    while ($row = $result->fetch_assoc()) {
        $row_Cycle = $row["Cycle"];
        $row_Ambient_Temp = $row["Ambient_Temp"];
        $row_Ambient_Humid = $row["Ambient_Humid"];
        $row_Compost_Temp = $row["Compost_Temp"];
        $row_Compost_Moist = $row["Compost_Moist"];
        $row_Compost_CH4 = $row["Compost_CH4"];
        $row_Deposit_Level = $row["Deposit_Level"];
        $row_ReadingTime = $row["ReadingTime"];

        echo '<tr>
                <td>' . $row_Cycle . '</td>
                <td>' . $row_Ambient_Temp . '</td>
                <td>' . $row_Ambient_Humid . '</td>
                <td>' . $row_Compost_Temp . '</td>
                <td>' . $row_Compost_Moist . '</td>
                <td>' . $row_Compost_CH4 . '</td>
                <td>' . $row_Deposit_Level . '</td>
                <td>' . $row_ReadingTime . '</td>
              </tr>';
    }
    $result->free();
```

Figura 26: Fragmento del script que muestra los datos de la tabla en el navegador. Imagen producida por el autor.

Terminada la API se pueden consultar los datos del compost utilizando en el navegador la ruta http://IPdeRaspberry/esp-data.php/. El resultado de la consulta será una tabla similar a la que se muestra en la figura 25.



Cycle	Ambient Temp	Ambient Humid	Compost Temp	Compost Moist	Compost CH4	Deposit Level	Reading Time
22	29.00	65.00	25.00	73.00	OK	OK	2021-08-23 20:29:03
21	27.00	65.00	26.00	56.00	OK	OK	2021-08-23 19:59:03
20	28.00	70.00	25.00	54.00	OK	OK	2021-08-23 19:29:03
19	28.00	70.00	25.00	56.00	OK	OK	2021-08-23 18:59:03
18	27.00	69.00	25.00	56.00	OK	OK	2021-08-23 18:58:05
17	29.00	64.00	26.00	55.00	OK	OK	2021-08-23 18:39:19
16	29.00	64.00	27.00	55.00	OK	OK	2021-08-23 18:09:19
15	28.00	67.00	27.00	55.00	OK	OK	2021-08-23 17:39:19
14	28.00	67.00	25.00	56.00	OK	OK	2021-08-23 17:09:19
13	29.00	67.00	26.00	55.00	OK	OK	2021-08-23 16:39:19
12	29.00	68.00	26.00	54.00	OK	OK	2021-08-23 16:09:19
11	29.00	64.00	28.00	56.00	OK	OK	2021-08-23 15:39:18
10	29.00	66.00	25.00	56.00	OK	OK	2021-08-23 15:09:18
9	30.00	63.00	27.00	55.00	OK	OK	2021-08-23 14:39:18
8	28.00	67.00	27.00	57.00	OK	OK	2021-08-23 14:09:18
7	28.00	67.00	26.00	34.00	OK	OK	2021-08-23 13:39:18
6	28.00	67.00	26.00	13.00	OK	OK	2021-08-23 13:09:18

Figura 27: Datos del compost vistos desde navegador web. Imagen producida por el autor.

4. IMPLEMENTACIÓN DEL SISTEMA

Este proyecto está orientado a la construcción de un prototipo funcional que automatice el proceso de compostaje de residuos orgánicos domésticos. Hasta ahora, se había tratado el tema de forma teórica, reservando el montaje y puesta en marcha de los distintos elementos que forman el sistema para esta sección. A continuación, se detalla el proceso seguido para la implementación práctica del diseño anteriormente descrito.

4.1. Montaje de la compostera

El primer paso para la implementación del sistema es plantear los elementos que forman la compostera: El contenedor para el compost, el depósito y los sistemas de riego y aireación. La estación de control también se incluirá en este punto. Como se ha explicado anteriormente, el montaje está pensado para alojarse en una terraza o jardín. Es de fácil configuración y estructura simple.

Para contener los residuos orgánicos durante el periodo de compostaje, se ha decidido utilizar una maceta de 30L. La maceta va perforada por sus laterales, garantizando una buena aireación de la mezcla. Según el manual de compostaje del Ministerio de Medio Ambiente [7], Es recomendable elevar ligeramente del suelo las macetas cuando se hace compost para que pueda salir el exceso de humedad en el fondo de esta y evitar así que se pudra la materia del fondo de la compostera. En este caso, al estar el prototipo en una terraza, se decidió introducir la maceta en un barreño para no manchar el suelo, calzando la compostera para evitar que toque el suelo del barreño.

El sistema de riego consta de un depósito para almacenar el agua, una bomba y los tubos de goma y conectores necesarios para llevar el agua hasta la compostera. El depósito es un cubo de polipropileno de 30L al que se hará un orificio cerca de la base para colocar un racor que permita acoplar la manguera de entrada a la bomba. La bomba deberá trabajar por impusión, de modo que se debe colocar a un nivel igual o inferior al de la base del depósito. Además, para facilitar el proceso de riego, es recomendable que el depósito esté ligeramente elevado respecto a la compostera. Para las conexiones a la bomba se utilizarán conectores rápidos. De esta forma, se facilita el montaje y desmontaje en caso de avería. El gotero se colocará centrado en la maceta para intentar distribuir el agua por toda su superficie.

Figura 28: Detalle de las conexiones entre bomba, depósito y tubería de riego mediante conectores rápidos. Imagen producida por el autor.

El sistema de aireación es el más simple de los que componen la compostera automática. La bomba de aire para acuario se enchufa directamente a una toma de corriente. De ella salen dos tubos de goma que trasportan el aire hasta la compostera. El caudal de salida se puede ajustar mediante un potenciómetro, colocándose para este caso en la posición de caudal máximo. A la salida se conectan dos difusores de piedra para dispersar mejor el aire y evitar que entre suciedad al tubo. Los difusores se entierran en la maceta intentando separarlos entre sí para airear varias zonas al máximo.

Figura 29: Detalle de los difusores de piedra para la salida del aire y el gotero de riego. Imagen producida por el autor.

Por último, queda comentar el montaje de la estación de control. Como se ha explicado anteriormente, esta estación aloja la electrónica del sistema de compostaje automático. La estación es, en esencia, una caja de conexiones con protección IP55 para evitar que entre agua cuando llueve. Va conectada a una toma de corriente doméstica, desde donde se alimentará tanto la placa de desarrollo como los enchufes a los que conectar las bombas de agua y aire. Estos enchufes irán sobre la tapa, por lo que hay que taladrarla para poder instalarlos utilizando una corona del diámetro adecuado. Además, para ir en consonancia con el grado de protección ante el agua que requiere la aplicación, se utilizarán enchufes de exterior también con protección IP55. La caja también tendrá conectores para los distintos sensores externos a la estación de control, de modo que habrá que taladrar también los espacios para estos.

Se puede ver el resultado final del montaje en la siguiente figura.

Figura 30: Sistema de compostaje desde diferentes perspectivas. Imagen producida por el autor.

4.2. Diagrama eléctrico y de conexiones

El *Anexo V* de esta memoria se puede ver el diagrama de conexiones junto con el esquema electrónico del sistema. Están producidos con Fritzing de forma que sean fáciles de entender por cualquiera que intente replicar el sistema. En la figura 31, se puede ver la central de control ya montada.

Existen ciertas discrepancias entre el esquema de conexiones y el esquema electrónico en cuanto a los pines utilizados. Esto se debe a un error en el protocolo del editor de los circuitos. Para poder utilizar el programa del *Anexo II*, se deberá seguir la distribución de pines que se muestra en el diagrama de conexiones. Utilizar la disposición del esquema electrónico no supondría ningún problema siempre y cuando se modificara la configuración de los pines en el software antes de cargarlo en el microcontrolador.

A fin de simplificar el esquema, se ha obviado la parte eléctrica, que afecta a la alimentación de la placa y a los contactos de los relés. La alimentación de la placa se llevará a cabo mediante una fuente de alimentación de cargador de móvil con una salida USB de tensión de 5V y corriente de 3A. Irá unida al cable de alimentación de la central de control mediante una regleta. Los relés, por su parte, se encargarán de cortar la fase de las tomas de corriente donde se conectarán la bomba de aire y la de agua.

Los sensores se conectan directamente a la placa siempre que sea posible. Para hacerlo, se utilizan conectores de 3 pines, 4 en el caso del sensor de ultrasonidos. Estos conectores llevarán los cables de la parte interior de la central, conectados a la placa de desarrollo por su otro extremo. En su parte exterior se podrá conectar de forma rápida el cable que comunique con cada sensor. Es importante etiquetar bien los cables para no confundir las entradas de la placa. El uso de conectores facilita la sustitución de los sensores en caso de rotura sin tener que desmontar la central de control.

Figura 31: Conector roscado de 3 pines como los utilizados en el prototipo. Imagen obtenida de dynamoelectronics.com

Las conexiones que no se pueden realizar directamente con el microcontrolador, se harán utilizando una placa protoboard. Solo se utilizará este método para el sensor de gas metano MQ-4 y para los Relés.

En el caso del sensor de gas, se utilizará un divisor de tensiones para rebajar la tensión que llegará al microcontrolador hasta los 3V. Los pines GPIO pueden soportar entradas de 5V, pero la experiencia con este sensor fue conflictiva. Se utilizará un divisor por seguridad.

Para el caso de los relés, el problema es el opuesto. La señal de activación del relé debe ser de 5V, pero la salida del pin GPIO a nivel alto del microcontrolador es de 3.3V. Se utilizará un sistema simple mediante transistores para solucionarlo. La función del transistor será, conectar 5V al relé cuando le llegue señal de 3V por su colector.

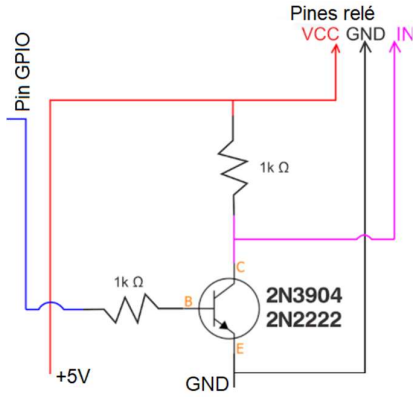


Figura 32: Diagrama del circuito de activación del relé. Imagen obtenida de qastack.mx

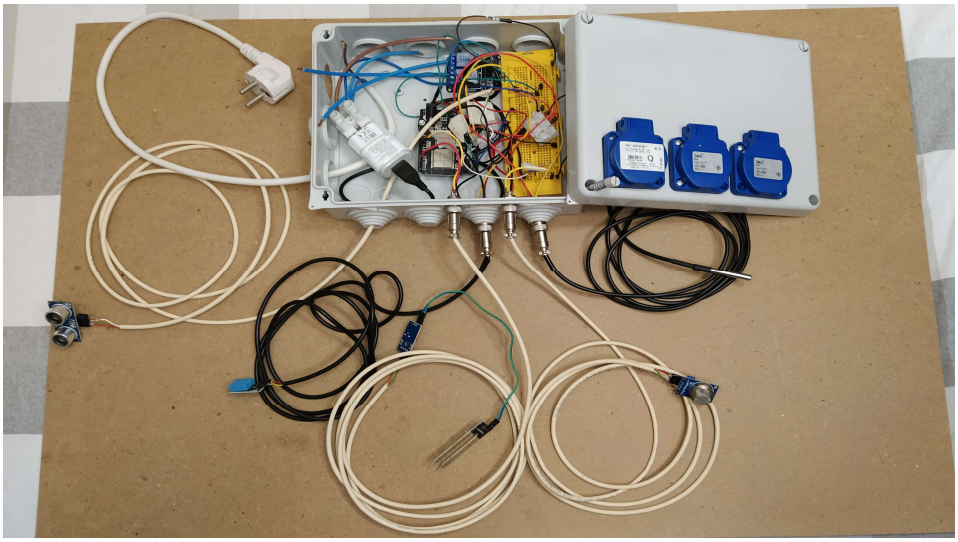


Figura 33: Detalle de la estación de control ya montada. Imagen producida por el autor.

4.3. Puesta en marcha

Para la puesta en marcha del prototipo, se deberán seguir por orden los siguientes pasos:

1. Sustituir las credenciales de la red WiFi doméstica en el código de programa contenido en el Anexo II.

2. Conectar la placa de desarrollo al ordenador mediante cable USB y cargar el software utilizando la IDE de Arduino.

3. Utilizar el código del anexo IV y V para crear las Api en la Raspberry Pi, dentro de la carpeta /var/www/html y con la extensión adecuada.

4. Editar las API modificando el nombre de la base de datos y el usuario y contraseña de phpMyAdmin. Mantener la Raspberry Pi siempre encendida.

5. Conectar los sensores y actuadores a la placa de desarrollo Wemos D1 R32 según lo descrito en el Anexo V: Diagrama eléctrico y de conexiones y el apartado 4.2 de la memoria.

6. Montar los componentes de la compostera inteligente según lo descrito en el apartado 4.1. Montaje de la compostera.

7. Colocar el higrómetro dentro de la pila de compost, hacer lo propio con la sonda de temperatura y dejar en el borde de la compostera el sensor de metano. Colocar el sensor de US en la tapa o soporte del depósito.

8. Conectar el cable de alimentación de la central de control.

El sistema comenzará a funcionar si se han seguido los pasos anteriores correctamente y teniendo en cuenta las consideraciones expuestas a lo largo de la memoria. Se podrá comprobar la primera medición a través del navegador a los pocos segundos de conectar la central de control. El prototipo quedará operativo durante todo el proceso de compostaje.

5. CONCLUSIÓN

El principal objetivo de este proyecto ha sido introducirse en el mundo del internet de las cosas de manera práctica. Estas tecnologías son novedosas y cuentan cada vez con más aplicaciones, pero solo se tratan a nivel teórico en las asignaturas de este máster.

La aplicación elegida implica crear un sistema funcional, de montaje simple y económico para gestionar el proceso de compostaje a nivel doméstico. También construir un prototipo replicable por cualquiera interesado en producir compost en su casa. Tras el desarrollo de este trabajo final de máster se concluye:

- Se ha desarrollado un prototipo funcional para el control del proceso de compostaje de residuos orgánicos domésticos. El sistema riega o airea el compost según las necesidades del proceso de forma automática.

- El sistema almacena los datos que determinan el proceso de compostaje en una base de datos, desde donde el usuario puede descargarlos para poder analizarlos.

- Se ha conseguido implementar la visualización de los datos del compost de forma rápida desde cualquier navegador web de un equipo conectado a la red local. De esta forma el usuario puede saber el estado del proceso en directo y de manera fácil.

- Los principales componentes utilizados son de hardware libre. También son de software libre los programas utilizados. El proyecto queda por tanto abierto a mejoras, cambios y es libre de ser reproducido por cualquier persona interesada.

- Este proyecto ha permitido aprender sobre las posibilidades que ofrece el internet de las cosas y las arquitecturas más habituales de estos sistemas. También ha supuesto el desarrollo de nuevas habilidades relacionadas con la programación e implementación de sistemas IoT, profundizando también en los conocimientos de electrónica adquiridos en el máster.

6. TRABAJOS FUTUROS

A pesar de que los objetivos marcados eran ambiciosos, siempre hay margen para mejoras tanto a nivel constructivo como de funcionalidades que incluya el prototipo. Uno de los más evidentes es el montaje del circuito electrónico sobre un soporte más adecuado para su uso a largo plazo. En lugar de utilizar una protoboard para las conexiones, sería recomendable utilizar una PCB u otro tipo de soporte que no esté pensado para montar y desmontar los componentes con frecuencia.

A nivel de funcionalidad, sería interesante abrir el acceso a los datos de forma segura desde cualquier parte del mundo. A pesar de que los niveles de humedad están definidos, una buena función podría ser añadir la capacidad de configurar de forma remota el nivel de consigna a mantener. De esta forma, podría adaptarse el nivel de humedad al tipo de materia orgánica que se introduzca en la compostera. Siguiendo con esta idea, la capacidad de configurar el tiempo de riego de forma remota resultaría muy útil para adaptar la cantidad de agua que se vierte sobre el compost de forma que el riego resulte lo más eficaz posible.

Por último, se podría mejorar la visualización de datos para hacerla más interactiva. Podrían incluirse gráficos y estadísticas generadas automáticamente cuando el usuario consulta los datos desde el navegador. De esta forma, resultaría más intuitiva la lectura y se evitará tener que descargar los datos desde phpMyAdmin para analizarlos manualmente.

Bibliografía

[1] F. Jariego, «La nueva revolución industrial: el internet de las cosas,» El Mundo, 30 junio 2014. [En línea]. Available: https://www.elmundo.es/economia/2014/06/30/53b14869e2704e97368b4577.html.

[2] I. Wigmore, «Internet de las cosas (IoT),» searchdatacenter, [En línea]. Available: https://searchdatacenter.techtarget.com/es/definicion/Internet-de-las-cosas-IoT.

[3] Oracle, «¿Qué es el IoT?,» Oracle, [En línea]. Available: https://www.oracle.com/es/internet-of-things/what-is-iot/.

[4] Nesta, «Open Hardware: Making awesome digital products,» Nesta, [En línea]. Available: https://www.nesta.org.uk/feature/digital-social-innovation/open-hardware/.

[5] Arduino.cc, «What is Arduino?,» Arduino official website, [En línea]. Available: https://www.arduino.cc/en/Guide/Introduction.

[6] RaspberryPi, «About Us,» Raspberry Pi Fundation, [En línea]. Available: https://www.raspberrypi.org/about/.

[7] Ministerio de medio Ambiente y Medio Rural y Marino, Manual de Compostaje, Madrid: Centro de publicaciones del ministerio de medio ambiente y medio rural y marino, 2009.

[8] S. Vigneswaran, J. Kandasamy y M. Johir, «Sustainable Operation of Composting in Solid Waste Management,» *Procedia Environmental Sciences,* vol. 35, pp. 408-415, 2016.

[9] D. Johnson, «The Compost Professor: A smart composting system,» hackster.io, 25 Octubre 2017. [En línea]. Available: https://www.hackster.io/darian-johnson/the-compost-professor-a-smart-composting-system-6a02f9.

[10] J. Beningo, «Cómo seleccionar y usar un módulo ESP32 con WiFi/bBluetooth adecuado para una aplicación IoT industrial,» Digi-Key, [En línea]. Available: https://www.digikey.es/es/articles/how-to-select-and-use-the-right-esp32-wi-fi-bluetooth-module.

[11] L. Llamas, «ESP32, el hermano mayor del ESP8266 con WiFi y Bluetooth,» Luís Llamas, 1 Abril 2018. [En línea]. Available: https://www.luisllamas.es/esp32/.

[12] M. Peláez, «KY-015 Placa sensora basada en DHT11,» Fantasy Studios, [En línea]. Available: https://www.fantasystudios.es/arduino/pages/Componentes/placas/sensores/sensor_ky-015.html.

[13] M. Integrated, «DS18B20 Datasheet,» Maxim Integrated, [En línea]. Available: https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf.

[14] I. Setowati, D. Novianto y E. Purnomo, «Preliminary design and soil moisture sensor yl-69 calibration for implementation of smart irrigation,» *Journal of Physics: Conference Series,* vol. 1517, nº 012078, 2020.

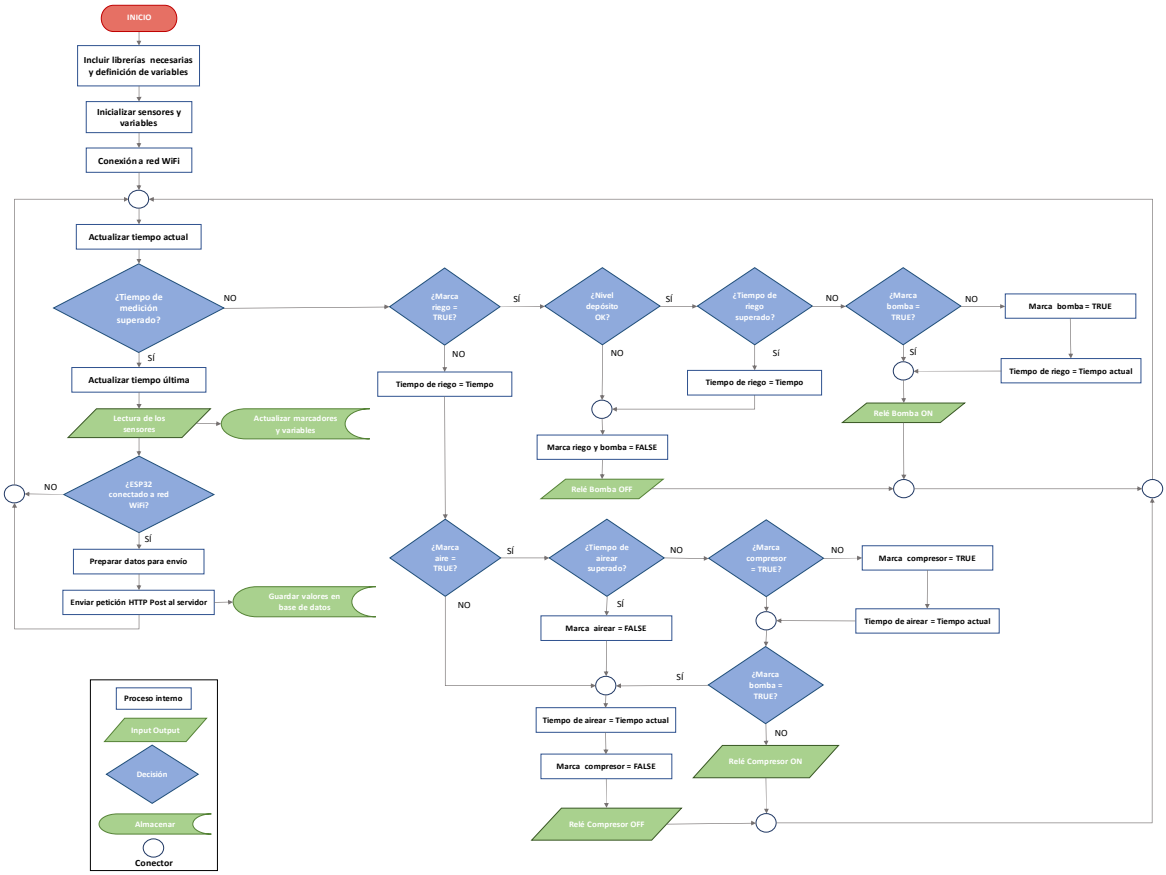
[15] W3schools, «HTTP Request methods,» W3schools, [En línea]. Available: https://www.w3schools.com/tags/ref_httpmethods.asp.

[16] Motzilla, «Métodos de petición HTTP,» MDN Web docs, [En línea]. Available: https://developer.mozilla.org/es/docs/Web/HTTP/Methods.

[17] R. Santos, «ESP32/ESP8266 Publishing Data to Raspberry Pi LAMP Server,» Random Nerd Tutorials, [En línea]. Available: https://randomnerdtutorials.com/esp32-esp8266-raspberry-pi-lamp-server/.

ANEXOS A LA MEMORIA

Anexo I: Diagrama de flujo



Anexo II: Software del programa

```
/*
 ************** PROGRAMA  AUTOMATIZACIÓN PROCESO COMPOSTAJE CON ESP32
************
 ************************** TRABAJO FINAL DE MÁSTER
*****************************
   MÁSTER UNIVERSITARIO EN INGENIERÍA MECATRÓNICA
   UNIVERSIDAD POLITÉCNICA DE VALENCIA
   AUTOR: Alberto García Martínez
   TUTOR: Francisco José Gimeno Sales
   CURSO: 2020-2021
   FECHA: Septiembre 2021
*/

// LIBRERÍAS PARA EL MANEJO DEL ESP32 COMO CLIENTE HTTP CONECTADO A
RED WIFI
#include <WiFi.h>
#include <HTTPClient.h>

//LIBRERÍAS PARA LOS SENSORES
#include<DHT.h> //Sensor DHT-11 humedad y temperatura
#include<OneWire.h> //Sensor DS18B20 temperatura comunicación
#include<DallasTemperature.h> //Sensor DS18B20 temperatura
interpretacion

//DATOS PARA CONEXION WIFI Y ENVIO AL SERVIDOR
const char* ssid = "MIWIFI_2G_vvLX";   //Nombre y contraseña de la red
WiFi
const char* password = "XYkzpEZu";
const char* serverName = "http://192.168.1.133/post-esp-data.php";
//Dirección del servidor y api de envío
String apiKeyValue = "tPmAT5Ab3j7F9"; //Clave de la api para procesar
la petición

//CONFIGURACION DEL SENSOR DHT-11
#define DHT_pin 5 //PIN conexion DHT
#define DHTTYPE DHT11 //Define tipo de sensor DHT11, DHT22...
DHT dht(DHT_pin, DHTTYPE, 27); // Instancia para la configuración del
sensor con pin, tipo y threshold (Cualquiera >11 funciona)

//CONFIGURACIÓN DEL SENSOR DS18B20
#define DS18B20_pin 13 //PIN Conexión DS18B20
OneWire oneWire(DS18B20_pin); //Setup del pin para comunicacion one
wire
DallasTemperature DS18B20(&oneWire); //Enlaza referencia de One Wire
como sensor para interpretar los datos

//CONFIGURACIÓN SENSOR US HC-SR04
#define ECHO_pin 17 //PIN conexión echo
#define TRIGGER_pin 16 //PIN conexión trigger
#define SOUND_VEL 0.034 //Velocidad del sonido para los cálculos

//PINES HIGRÓMETRO
#define HIGRO_pin 2
#define HIGRO_power 4

//PINES SENSOR DE CH4 MQ-4
#define MQ4_pin 14
```

```
//PINES RELÉS BOMBA Y COMPRESOR
#define REL_water 1
#define REL_air 3

//VARIABLES GLOBALES PARA ENVÍO DE DATOS
String Compost_Temp;
String Compost_Moist;
String Ambient_Temp;
String Ambient_Humid;
String Deposit_Level;
String Compost_CH4;

//FLAGS PARA TOMA DE DECISIONES
bool Flag_Water = false; //True cuando riego necesario false cuando no
bool Flag_Air = false; //True cuando aire necesario false cuando no
bool Flag_Level = true; //True cuando nivel OK false cuando nivel NOK
bool Flag_Pump = false; //False cuando NO se ha activado la bomba true
cuando SI
bool Flag_Comp = false; //False cuando NO se ha activado el compresor
true cuando SI

//TIEMPOS PARA INTERVALO ENTRE MEDIDAS
unsigned long previousMillis_Sensors = 0;
unsigned long previousMillis_Water = 0;
unsigned long previousMillis_Air = 0;
const long interval_Sensors = 1800000;   //30min
const long interval_WaterAir = 120000; //2min


void setup() { //PRINCIPIO DEL SETUP
  Serial.begin(115200); //Iniciamos monitor serie para mensajes debug

  //CONECTAR A LA RED WIFI
  WiFi.begin(ssid, password);
  Serial.println("Conectando");
  while (WiFi.status() != WL_CONNECTED) { //Muestra conectando
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connectado a la red WiFi con dirección IP: ");
  Serial.println(WiFi.localIP());


  // INICIAR SENSORES O CONFIGURAMOS PINES DE LECTURA
  dht.begin(); //Sensor DHT11
  DS18B20.begin(); //Sensor DS18B20
  pinMode(HIGRO_pin, INPUT); //Configurar pin  lectura higrómetro como
entrada
  pinMode(HIGRO_power, OUTPUT); //Configurar pin alimentación
higrómetro como salida
  pinMode(MQ4_pin, INPUT); //Configurar pin lectura sensor de metano
  pinMode(TRIGGER_pin, OUTPUT); //Pin del trigger como salida
  pinMode(ECHO_pin, INPUT); //Pin del echo como entrada

  //INICIAMOS PINES DE LOS RELÉS COMO SALIDAS
  pinMode(REL_water, OUTPUT);
  digitalWrite(REL_water, LOW);
  pinMode(REL_air, OUTPUT);
  digitalWrite(REL_air, LOW);
```

```
} //FINAL DEL SETUP

void loop() { //PRINCIPIO DEL LOOP

  unsigned long currentMillis = millis(); //Actualiza el valor del
tiempo en una variable tipo long

  //Realizamos mediciones
  if (currentMillis - previousMillis_Sensors >= interval_Sensors ||
previousMillis_Sensors == 0) { //Si ha pasado el intervalo o es la
primera vez que se ejecuta el programa

    previousMillis_Sensors = currentMillis; //Actualizamos tiempo

    //GUARDAMOS LAS VARIABLES QUE NO REQUIEREN DE FUNCION PROPIA NI
AFECTAN A FLAGS
    Compost_Temp = String(DS18B20.getTempCByIndex(0)); //Guarda la
temperatura del compost en su string
    Ambient_Temp = String(dht.readTemperature()); //Guarda la
temperatura ambiente en su string
    Ambient_Humid = String(dht.readHumidity()); //Guarda la humedad
ambiente en su string

    //GUARDAMOS LAS VARIABLES QUE AFECTAN A FLAGS Y TIENEN FUNCIÓN
PROPIA
    Compost_CH4 = CH4measure();
    Compost_Moist = Moistmeasure();

    //Tratamos de manera especial la función de nivel por modo
escritura
    Levelmeasure();
    if (Flag_Level == true) {
      Deposit_Level = "OK";
    }
    else {
      Deposit_Level = "NOK";
    }

    //MOSTRAMOS POR PANTALLA LOS DATOS MEDIDOS PARA DEPURAR
    Serial.print("Temperatura ambiente: ");
    Serial.println(Ambient_Temp);
    Serial.print("Humedad ambiente: ");
    Serial.println(Ambient_Humid);
    Serial.print("Temperatura compost: ");
    Serial.println(Compost_Temp);
    Serial.print("Humedad compost: ");
    Serial.println(Compost_Moist);
    Serial.print("Metano en el aire: ");
    Serial.println(Compost_CH4);
    Serial.print("Nivel del depósito: ");
    Serial.println(Deposit_Level);

    //ENVÍO DE DATOS
    if (WiFi.status() == WL_CONNECTED) { //Comprobamos conexión
      //Ejecutamos funcion de envío de datos
      send_data();
    }

  } //FINAL DEL CICLO DE MEDICIÓN Y ENVÍO
```

```
  //CICLO DE RIEGO
  if (Flag_Water == true && currentMillis - previousMillis_Water <=
interval_WaterAir ) { //Comprobamos necesidad de riego y no haber
pasado el tiempo de riego

    if (Flag_Pump == false) { // Si la bomba no está encendida aun
      previousMillis_Water = millis();  //Guardo tiempo de puesta en
marcha
      Serial.println("Riego en marcha"); //Mostramos mensaje de que se
ha activado riego
      Flag_Pump = true;
    }

    Levelmeasure(); //Medimos nivel del deposito antes de decidir si
la bomba se activa

    if (Flag_Level == true) { // Si el nivel está OK
      digitalWrite(REL_water, HIGH);
    }
    else {
      digitalWrite(REL_water, LOW); //Desactivamos riego
      Flag_Pump = false; //Marcamos bomba como apagada
      Flag_Water = false; //Marcamos riego como no necesario
      Serial.println("Imposible regar, nivel depósito bajo");
    }
    return;
  }
  else { //Si la humedad está ok o ha pasado el tiempo
    Serial.println("Riego detenido");
    digitalWrite(REL_water, LOW);
    Flag_Water = false; //Desactivamos riego
    Flag_Pump = false; //Marcamos bomba como apagada
    previousMillis_Water = millis();
  } //Cierre if condicion de riego


  //CICLO DE AIREAR
  if (Flag_Air == true && currentMillis - previousMillis_Air <=
interval_WaterAir) { //Si es necesario airear y no ha pasado el
intervalo
    if (Flag_Comp == false) { //Si el compresor no está encendido aun
      previousMillis_Air = millis(); //Guardamos tiempo de puesta en
marcha
    }
    digitalWrite(REL_air, HIGH); //Activa compresor
    Flag_Comp = true; //Marca como activo el compresor
    Serial.println("Aire activo");
  }
  else { //Si no es necesario airear o el tiempo ha pasado
    digitalWrite(REL_air, LOW); //Desactivamos compresor
    Serial.println("Aire desactivado");
    Flag_Comp = false; //Marcamos compresor como desactivado
    Flag_Air = false; //Marcamos airear como no necesario
    previousMillis_Air = millis();
  }

  if (Flag_Pump == true) { //Si el riego está encendido
    digitalWrite(REL_air, LOW); //Desactivamos compresor
    Serial.println("Riego en marcha, imposible encender compresor");
    Flag_Comp = false; //Marcamos compresor como desactivado para
poder tomar tiempo de activación después
```

```
    previousMillis_Air = millis(); //Ponemos el tiempo a cero para que
se cumpla la condición de poder empezar a airear
  }//Final bloque aireador

}//FINAL DEL LOOP


//FUNCIÓN PARA MEDIR METANO
String CH4measure() {
  int CH4 = digitalRead(MQ4_pin);
  String estado;
  if (CH4 == HIGH) { //Si el pin está en alto no hay metano
    Flag_Air = false; //Marca aire como NO necesario
    estado = "OK";
  }
  else { //Si NO está en HIGH entonces hay metano
    Flag_Air = true; //Marca aire como SI necesario
    estado = "NOK";
  }
  return estado; //Devuelve la string estado
} //Final función CH4measure

//FUNCIÓN PARA MEDIR HUMEDAD DEL COMPOST
String Moistmeasure() {
  String measure;
  digitalWrite(HIGRO_power, HIGH); //Alimentamos el higrometro
  delay(500); //Delay para dar tiempo al sensor a conectarse
  int moist = analogRead(HIGRO_pin); //Leemos conductividad entre
valores 4095 y 370
  digitalWrite(HIGRO_power, LOW); //Cortamos alimentación del sensor
  float moist_prcnt = map(moist, 4095, 370, 0, 100); //Mapeamos el
valor para pasarlo a porcentaje

  if (moist_prcnt >= 50) { //Comprobamos si la humedad del compost
está por encima del 50%
    Flag_Water = false; //Marca riego como NO necesario
  }
  else { //Si está por debajo del 50%
    Flag_Water = true; //Marca riego como SI necesario
  }
  measure = String(moist_prcnt); //Transformamos porcentaje de humedad
en string
  return measure; //Devolvemos el valor
}//Final de la funcion Moistmeasure

//FUNCIÓN PARA MEDIR NIVEL DEL AGUA
void Levelmeasure() {
  digitalWrite(TRIGGER_pin, LOW);
  delayMicroseconds(5); //Apagar trigger 5us para evitar ecos
  digitalWrite(TRIGGER_pin, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIGGER_pin, LOW); //Encendemos trigger durante 10 us

  long duration = pulseIn(ECHO_pin, HIGH); //Lee el echo y devuelve el
tiempo de vuelta del sonido en us
  float distance = duration * SOUND_VEL / 2;

  if (distance >= 25) { //Si la distancia es mayor a 25cm el nivel de
agua es bajo
    Flag_Level = false; //Marca nivel como NOK
  }
```

```
  else {
    Flag_Level = true; //Marca nivel como OK
  }
} //Final de la función Levelmeasure

//FUNCIÓN ENVÍO DE DATOS A SERVIDOR
void send_data() {
  //Configuramos ESP como cliente que hace peticiones http
  WiFiClient client;
  HTTPClient http;

  // Iniciamos conexión con servidor en modo cliente
  http.begin(client, serverName);

  // Define el header sobre contenido
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");

  // Confeccionamos la petición POST
  String httpRequestData = "api_key=" + apiKeyValue + "&Ambient_Temp="
+ Ambient_Temp
                           + "&Ambient_Humid=" + Ambient_Humid
                           + "&Compost_Temp=" + Compost_Temp
                           + "&Compost_Moist=" + Compost_Moist
                           + "&Compost_CH4=" + Compost_CH4
                           + "&Deposit_Level=" + Deposit_Level
                           + "";
  Serial.print("httpRequestData: "); //Imprimimos la peticion por
puerto serie para depurar
  Serial.println(httpRequestData);

  // Enviamos la petición con http.POST y  guardamos la respuesta del
servidor
  int httpResponseCode = http.POST(httpRequestData);

  //Imprimimos la respuesta o informamos del error
  if (httpResponseCode > 0) {
    Serial.print("HTTP Response code: ");
    Serial.println(httpResponseCode);
  }
  else {
    Serial.print("Error code: ");
    Serial.println(httpResponseCode);
  }

  //Terminamos la conexion
  http.end();
} //Final función envío de datos al servidor
```

Anexo III: Código de la API Post ESP Data

```
<?php
/*
 API PARA PROCESAR PETICIONES  HTTP POST Y AÑADIR CONTENIDO A MySQL
Basado en el código de Rui Santos con modificaciones de Alberto Garcia
Rui Santos: Complete project details at
https://RandomNerdTutorials.com/esp32-esp8266-mysql-database-php/

  Permission is hereby granted, free of charge, to any person
obtaining a copy of this software and associated documentation files.

  The above copyright notice and this permission notice shall be
included in all copies or substantial portions of the Software.
*/

$servername = "localhost";

// REPLACE with your Database name
$dbname = "esp_data";
// REPLACE with Database user
$username = "admin";
// REPLACE with Database user password
$password = "AlbertoRaspi";

// Keep this API Key value to be compatible with the ESP32 code
provided in the project page.
// If you change this value, the ESP32 sketch needs to match
$api_key_value = "tPmAT5Ab3j7F9";

$api_key= $Ambient_Temp = $Ambient_Humid = $Compost_Temp =
$Compost_Moist = $Compost_CH4 = $Deposit_Level = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $api_key = test_input($_POST["api_key"]);
    if($api_key == $api_key_value) {
        $Ambient_Temp = test_input($_POST["Ambient_Temp"]);
        $Ambient_Humid = test_input($_POST["Ambient_Humid"]);
        $Compost_Temp= test_input($_POST["Compost_Temp"]);
        $Compost_Moist= test_input($_POST["Compost_Moist"]);
        $Compost_CH4= test_input($_POST["Compost_CH4"]);
        $Deposit_Level= test_input($_POST["Deposit_Level"]);

        // Create connection
        $conn = new mysqli($servername, $username, $password,
$dbname);
        // Check connection
        if ($conn->connect_error) {
            die("Connection failed: " . $conn->connect_error);
        }

        $sql = "INSERT INTO CompostData (Ambient_Temp, Ambient_Humid,
Compost_Temp, Compost_Moist, Compost_CH4, Deposit_Level)
        VALUES ('" . $Ambient_Temp . "', '" . $Ambient_Humid . "','" .
$Compost_Temp . "','" . $Compost_Moist . "','" . $Compost_CH4 . "','"
. $Deposit_Level . "')";

        if ($conn->query($sql) === TRUE) {
            echo "New record created successfully";
        }
        else {
```

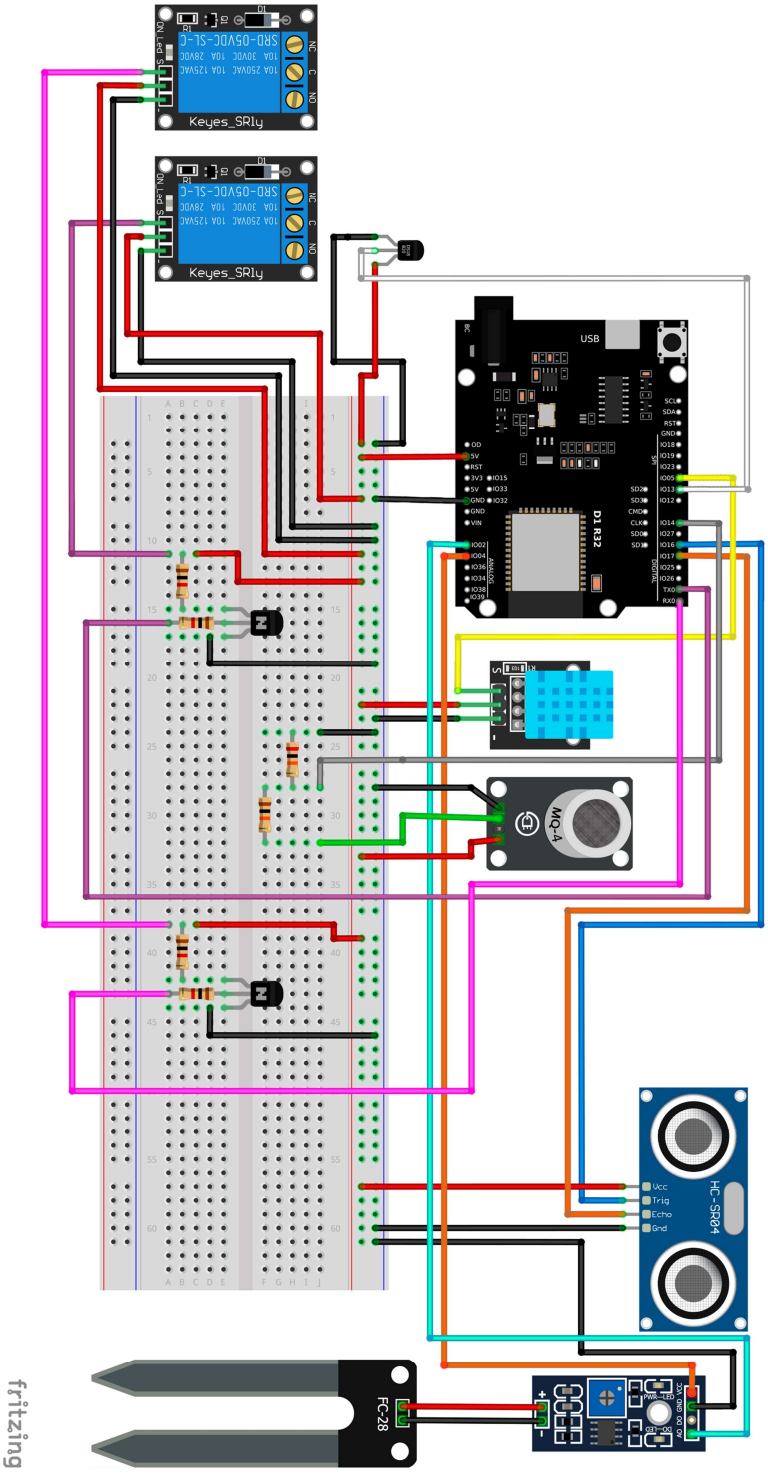
```
            echo "Error: " . $sql . "<br>" . $conn->error;
        }

        $conn->close();
    }
    else {
        echo "Wrong API Key provided.";
    }

}
else {
    echo "No data posted with HTTP POST.";
}

function test_input($data) {
    $data = trim($data);
    $data = stripslashes($data);
    $data = htmlspecialchars($data);
    return $data;
}
```

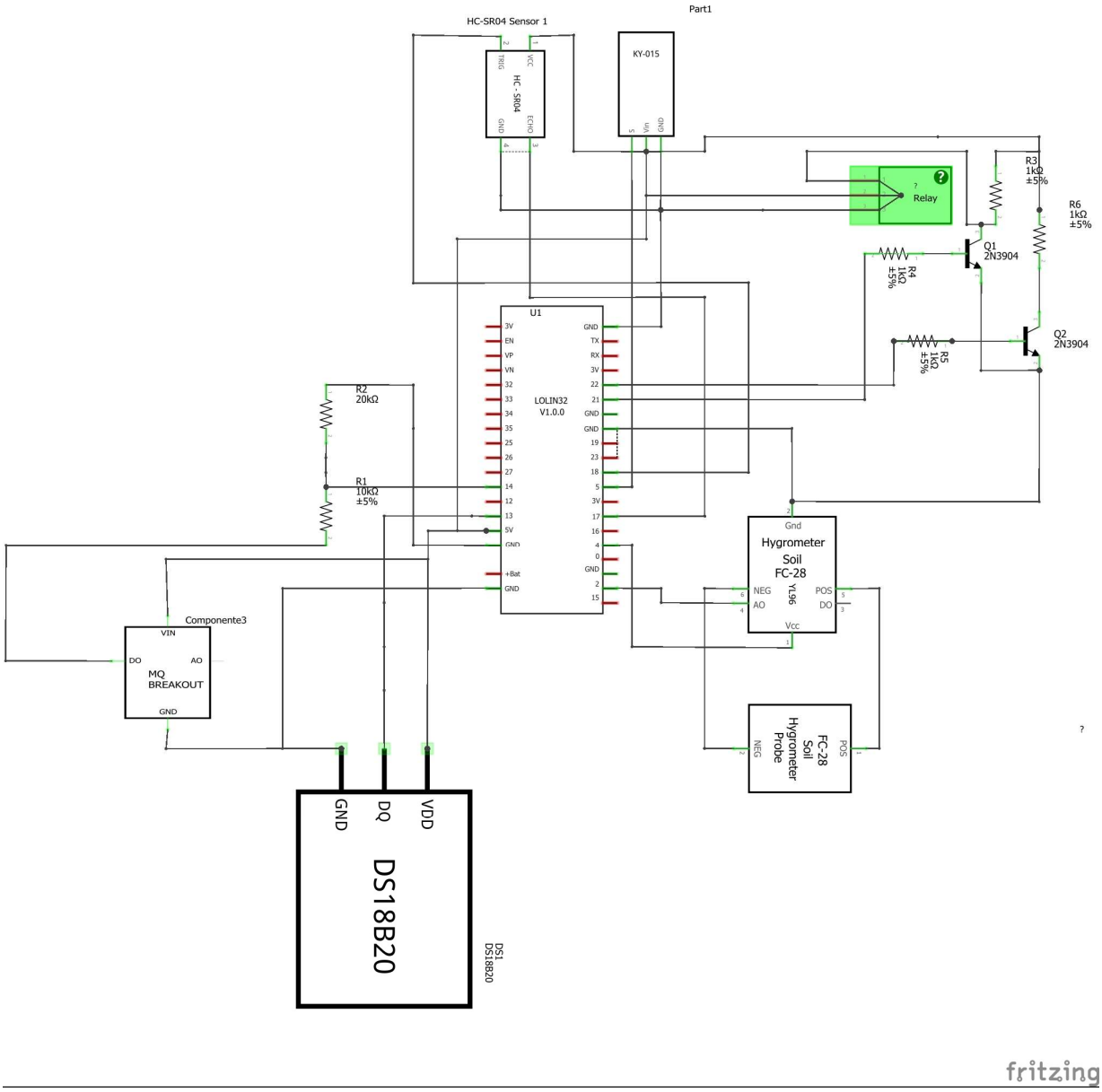
Anexo IV: Código de la API ESP Data

```
<!DOCTYPE html>
<html><body>
<?php
/*
API PARA MOSTRAR LOS DATOS ALMACENADOS EN LA BASE DE DATOS
Basado en el código de Rui Santos con modificaciones de Alberto Garcia
Rui Santos: Complete project details at
https://RandomNerdTutorials.com/esp32-esp8266-mysql-database-php/

  Permission is hereby granted, free of charge, to any person
obtaining a copy of this software and associated documentation files.

  The above copyright notice and this permission notice shall be
included in all copies or substantial portions of the Software.
*/

$servername = "localhost";

// REPLACE with your Database name
$dbname = "esp_data";
// REPLACE with Database user
$username = "admin";
// REPLACE with Database user password
$password = "AlbertoRaspi";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT Cycle, Ambient_Temp, Ambient_Humid, Compost_Temp,
Compost_Moist, Compost_CH4, Deposit_Level,  ReadingTime FROM
CompostData ORDER BY Cycle DESC";

echo '<table cellspacing="5" cellpadding="5">
      <tr>
        <td>Cycle</td>
        <td>Ambient Temp</td>
        <td>Ambient Humid</td>
        <td>Compost Temp</td>
        <td>Compost Moist</td>
        <td>Compost CH4</td>
        <td>Deposit Level</td>
        <td>Reading Time</td>
      </tr>';

if ($result = $conn->query($sql)) {
    while ($row = $result->fetch_assoc()) {
        $row_Cycle = $row["Cycle"];
        $row_Ambient_Temp = $row["Ambient_Temp"];
        $row_Ambient_Humid = $row["Ambient_Humid"];
        $row_Compost_Temp = $row["Compost_Temp"];
        $row_Compost_Moist = $row["Compost_Moist"];
        $row_Compost_CH4 = $row["Compost_CH4"];
        $row_Deposit_Level = $row["Deposit_Level"];
        $row_ReadingTime = $row["ReadingTime"];
```

```
        echo '<tr>
                <td>' . $row_Cycle . '</td>
                <td>' . $row_Ambient_Temp . '</td>
                <td>' . $row_Ambient_Humid . '</td>
                <td>' . $row_Compost_Temp . '</td>
                <td>' . $row_Compost_Moist . '</td>
                <td>' . $row_Compost_CH4 . '</td>
                <td>' . $row_Deposit_Level . '</td>
                <td>' . $row_ReadingTime . '</td>
              </tr>';
    }
    $result->free();
}

$conn->close();
?>
</table>
</body>
</html>
```

Anexo V: Diagrama eléctrico y de conexiones




Part1
HC-SR04 Sensor 1
HC - SR04
TRIG
VCC
ECHO
GND
KY-015
Vin
GND
Relay
R3
1kΩ
±5%
R6
1kΩ
±5%
Q1
2N3904
R4
1kΩ
±5%
Q2
2N3904
R5
1kΩ
±5%
U1
LOLIN32
V1.0.0
R2
20kΩ
R1
10kΩ
±5%
Componente3
VIN
DO
AO
MQ
BREAKOUT
GND
Hygrometer
Soil
FC-28
V1.96
Gnd
NEG
AO
POS
DO
Vcc
FC-28
Soil
Hygrometer
Probe
NEG
POS
GND
DQ
VDD
DS18B20
DS1
DS18B20
fritzing

Documento II: PLIEGO DE CONDICIONES

Objeto del pliego

Este pliego de condiciones presenta los requisitos exigibles al contratista para llevar a cabo el proyecto “Diseño e implementación de un sistema IoT mediante la plataforma ESP32 para la automatización del proceso de compostaje de residuos orgánicos domésticos”

El objeto del documento es el de especificar las condiciones técnicas para el correcto desarrollo del proyecto. No se pretende definir los detalles constructivos del mismo, pues será responsabilidad del contratista que la selección de componentes y detalles constructivos se corresponda con las últimas normas de diseño y técnicas aplicables.

Documentación del proyecto y compatibilidad de documentos

El presente proyecto se compone de los siguientes documentos que son compatibles entre sí y complementarios:

- Documento 1. Memoria y anexos
- Documento 2. Pliego de condiciones
- Documento 3. Presupuesto

Condiciones técnicas

Para poder realizarse el proyecto se pondrá a disposición del contratista toda la documentación antes detallada. Cualquier cambio será consultado con la dirección facultativa del proyecto. Los cambios podrán ser valorados siempre y cuando no modifiquen la funcionalidad del sistema y vayan acorde con los objetivos marcados.

La memoria y sus anexos serán siempre prioritarios al resto de documentos para un correcto desarrollo del proyecto. En este documento se detallan los objetivos, funcionalidades y metodología a seguir para el desarrollo del sistema.

Desde el punto de vista legal y facultativo el documento capital será el presente pliego de condiciones.

El presupuesto quedará en última posición en cuanto a jerarquía, siendo el único de no obligado cumplimiento por suponer únicamente una orientación económica cuantificativa de los costes del proyecto además de una medición de los materiales necesarios para llevarlo a cabo.

Condiciones facultativas

La dirección facultativa será responsabilidad de un ingeniero del ámbito industrial. Este será responsable del correcto desarrollo del proyecto y de alcanzar los objetivos marcados en el Documento I.

En cuanto a la fabricación, el contratista será el máximo responsable de los posibles errores de construcción y deberá responder por ellos ante la dirección facultativa. El contratista deberá también facilitar cualquier certificado necesario en materia de calidad de los materiales, maquinaria y equipo empleado para la fabricación.

Entregables

Al finalizar el proyecto, la dirección facultativa deberá entregar una memoria técnica detallando todos los procesos y metodologías seguidas para su desarrollo. Anexado a esta memoria también se entregará el código de los programas desarrollados y los esquemas eléctricos a seguir para la fabricación.

Como documento aparte, se entregará un presupuesto en el que se detallen los materiales utilizados y sus costes asociados.

Por último, se dispondrá de un pliego de condiciones donde se describan todas las disposiciones técnicas, legales y económicas derivadas del desarrollo del proyecto, así como los certificados y garantías relativas al producto final.

Documento III: PRESUPUESTO

En este documento se expone la relación de costes asociados al proyecto “Diseño e implantación de un sistema IoT basado en ESP32 para la automatización del proceso de compostaje de residuos orgánicos domésticos”. Los precios son únicamente orientativos, suponiendo un punto de referencia para el contratista a la hora de efectuar sus ofertas.

Presupuesto de ejecución material (PEM)

La siguiente tabla detalla los costes asociados a los materiales del proyecto:

DESCRIPCIÓN	UNIDADES	PRECIO (€)	TOTAL (€)
SENSORES			**14,26**
Sensor de temperatura y humedad DHT 11	1	2,67	2,67
Sensor de humedad para suelo YL69	1	1,60	1,60
Módulo ultrasonidos HC-SR04	1	4,49	4,49
Módulo sensor de metano MQ-4	1	4,27	4,27
Sonda de temperatura 18B20, 2 metros	1	1,23	1,23
ACTUADORES			**37,21**
Mini Bomba Sumergible 800 l/h, 5 mca, 12VDC	1	15,50	15,50
Bomba de aire para acuario Ireenuo, 3W, 2x150 l/h	1	19,99	19,99
Módulo Relé 5V KY-019 2 canales con optoacoplador	1	1,72	1,72
COMPOSTERA Y SISTEMA DE RIEGO			**33,52**
Maceta redonda 30 l con asas	1	14,99	14,99
Cubo de polipropileno de 30 l con válvula de salida	1	12,95	12,95
Tubo de riego 16 mm	1	0,24	0,24
Válvula de 16 mm para tubería de riego	1	1,25	1,25
Tapón tubería 16 mm	1	0,32	0,32
Gotero	1	0,27	0,27
Conector rápido macho tipo Cam-lock, DN 20 mm, rosca hembra	2	0,20	0,40

DESCRIPCIÓN	UNIDADES	PRECIO (€)	TOTAL (€)
Conector rápido hembra tipo Cam-lock, DN 20 mm, de espiga	2	1,27	2,54
Abrazadera metálica de 13 a 24 mm diam	4	0,14	0,56
CONTROL			**87,46**
Caja de conexiones estanca IP55 de 170x220x85mm	1	4,29	4,29
Placa de desarrollo AZDelivery ESP32 D1 R32 con CH340G y WiFi + Bluetooth	1	9,99	9,99
Raspberry Pi 4b, 4 Gb en caja con ventilador	1	73,18	73,18
ALIMENTACIÓN			**5,93**
Fuente de alimentación salida USB 5V, 3 A	1	5,93	5,93
COMPONENTES ELECTRÓNICOS			**4,69**
Protoboard	1	3,95	3,95
Transistores 2N3904	2	0,25	0,50
Resistencias 1/4 W diferentes valores	6	0,04	0,24
		TOTAL COSTES MATERIALES (€)	**187,28**

A continuación, se listan los recursos de software requerido para el proyecto:

DESCRIPCIÓN	UNIDADES	PRECIO (€)	TOTAL (€)
SOFTWARE			**77,00**
Arduino IDE	-	-	-
Fritzing	1,00	8,00	8,00
Microsoft Office	1,00	69,00	69,00
		TOTAL SOFTWARE (€)	**77,00**

El coste de software anterior quedará posteriormente incluido en el apartado de amortización de activos.

El siguiente apartado recoge los costes de personal asociado al proyecto

DESCRIPCIÓN	UNIDADES	PRECIO (€)	TOTAL (€)
COSTES POR RECURSOS HUMANOS			**8.750,00**
Ingeniero Junior	350,00	25,00	8.750,00
TOTAL SOFTWARE Y COSTES POR RECURSOS HUMANOS (€)			**8.750,00**

Presupuesto de ejecución por contrato (PEC) y total

El presupuesto de ejecución incluye las partidas anteriores, a las que se han de añadir tanto los gastos generales de empresa como el beneficio industrial así como los gastos de suministros y servicios y la amortización de activos.

DESCRIPCIÓN	COSTE (€)
A. Costes Materiales	**187,28**
B. Costes por Recursos Humanos	**8.750,00**
C. Presupuesto de Ejecución Material (PEM) (A+B)	**8.937,28**
D. Amortización de activos materiales e inmateriales (2%)	**178,75**
E. Suministros y servicios (3%)	**268,12**
F. Gastos generales (13%)	**1161,85**
G. Beneficio Industrial (6%)	**536,24**
H. Presupuesto de Ejecución por Contrato (PEC) (C+D+E+F+G)	**11.182,24**
I. Impuesto sobre el Valor Añadido (IVA) (21% del PEC)	**2.327,27**
COSTE TOTAL DEL PROYECTO (F+G)	***13.509,51***