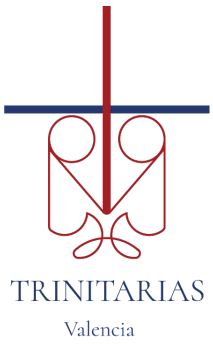



Colegio Virgen del Remedio

Madrid

COLEGIO VIRGEN DEL REMEDIO

PROYECTO TECNOLOGÍA

# Automatización del cuidado de una planta

**Autores**

Samuel Matamoros, Alejandro Clérigo y Manuel Gómez

15/04/2024

# Resumen

En primer lugar, este proyecto tiene un objetivo claro, que es, el ahorro de tiempo en la vida de una persona sin descuidar a las plantas, ya que son seres vivos como nosotros. Además, queremos fomentar el desarrollo de la jardinería con la implementación de nuevas metodologías, debido a que es un sector que está perdiendo paulatinamente relevancia en la sociedad.

En cuanto a la metodología, fabricaremos una maceta inteligente que mediante una serie de sensores y elementos de programación será capaz de aportar de manera automática a la planta lo que necesita para llevar una vida óptima, esta tendrá una pantalla inteligente en el que el propietario podrá ver toda la información de su planta en todo momento y reguladores para poder medir las cantidades.

Mediante el uso de la tecnología, pretendemos realizar un sistema automático capaz de cubrir las necesidades básicas de la planta para el correcto cuidado de esta.

# Abstract

Firstly, this project has a clear objective, which is to save time in a person's life without neglecting the plants, as they are living beings like us. Additionally, we aim to promote the development of gardening through the implementation of new methodologies, as this is a sector that is gradually losing relevance in society.

As for the methodology, we will manufacture a smart pot that, through a series of sensors and programming elements, will be able to automatically provide the plant with what it needs to lead an optimal life. It will have a smart screen on which the owner can view all the information about their plant at any time, as well as regulators to measure the quantities.

Using tecnology, we intend to develop an autonomous system capable of covering the vital needs of a plant.

# Índice de contenidos

# Índice de tablas

# Índice de figuras

# Capítulo 1. Motivación

El progreso tecnológico ofrece una ventana de oportunidad para abordar desafíos cotidianos de una manera más eficiente y sostenible. En un mundo donde la sobrecarga de información y las demandas constantes pueden abrumar, la automatización de tareas diarias emerge como un alivio para la mente moderna, liberando tiempo y energía para dedicar a otras actividades significativas.

Además, el vínculo humano con la naturaleza y las plantas es innegable. Las plantas no solo añaden belleza a nuestro entorno, sino que también proporcionan una conexión tangible con el mundo natural. Su color, aroma y la sensación de frescura que irradian son aspectos que enriquecen nuestras vidas diarias. Para aquellos que apreciamos la presencia de las flores y plantas en nuestros hogares, el cuidado y la observación de su crecimiento son fuentes de satisfacción y tranquilidad.

Sin embargo, mantener el cuidado adecuado de las plantas puede ser un desafío, especialmente cuando se enfrenta a la falta de tiempo o a imprevistos en la rutina diaria. Es en este cruce entre el interés por la tecnología y la pasión por la naturaleza donde surge la motivación para desarrollar este proyecto: *Automatización del cuidado de una planta*. La convergencia de estos dos mundos permite explorar nuevas formas de garantizar el bienestar de nuestras plantas mientras se integran soluciones innovadoras en nuestra vida cotidiana.

# Capítulo 2. Introducción

## 2.1. - Planteamiento

Las plantas, con su belleza y serenidad, añaden una dimensión vital a nuestros espacios, ya sea en el hogar, la oficina o cualquier otro lugar. Sin embargo, mantenerlas en óptimas condiciones puede convertirse en un desafío, especialmente en un mundo donde el tiempo es un recurso escaso.

Este problema puede resultar en la muerte prematura de las plantas, la disminución de su salud y vitalidad, así como en una experiencia de jardinería menos satisfactoria para los aficionados. Además, la falta de cuidado adecuado puede generar sentimientos de frustración y culpa en aquellos que desean mantener un entorno verde y saludable en su hogar o lugar de trabajo.

Ante este panorama, surge la necesidad de encontrar una solución que permita automatizar el cuidado de las plantas, garantizando que reciban la cantidad adecuada de agua, luz y nutrientes en todo momento, incluso en ausencia de la atención directa del propietario. Este enfoque no solo facilitaría el mantenimiento de un entorno óptimo para el crecimiento de las plantas, sino que también liberaría tiempo y energía para que las personas disfruten de la belleza y los beneficios de tener plantas en su vida diaria.

En conclusión, la automatización del cuidado de las plantas representa una solución innovadora, prometedora y eficaz para aquellos que deseen disfrutar de los beneficios y la belleza de las plantas sin dedicar demasiado tiempo y esfuerzo a su cuidado.

## 2.2. - Objetivos

El objetivo principal de nuestro proyecto no es obtener un beneficio económico, sino ganar tiempo para realizar otras actividades y así quitarnos preocupaciones a la hora de mantener en buen estado las plantas de nuestro hogar. MA-Z permite despreocuparte del cuidado de las plantas de tu hogar para que estas estén sanas y en condiciones óptimas y muy vistosas para decorar tu casa.

Por otro lado se pretende con este estudio investigar en la automatización como concepto y el desarrollo de sistemas de recopilación y análisis de datos. El objetivo principal de nuestro proyecto no es obtener un beneficio económico, aunque podría convertirse en un potencial producto si se deseara.

## 2.3. - Estructura

El presente documento se estructura en varios capítulos y secciones, los cuales abordan diferentes aspectos relacionados con el desarrollo de un proyecto de automatización de riego mediante el uso de tecnología IoT. A continuación, se describe brevemente el contenido de cada parte.

En la primera sección, se expone la motivación que impulsó la realización del proyecto, destacando la importancia de mejorar la eficiencia en el uso del agua y optimizar los procesos de riego en la agricultura.

Posteriormente, en el apartado de introducción, se proporciona un contexto general del proyecto, se establecen los objetivos a alcanzar y se esboza la estructura general del documento. Además, se mencionan las distintas fases del proyecto que se tratarán en detalle en las secciones siguientes.

El capítulo de memoria constituye la parte central del documento y se divide en varias secciones. La sección de memoria se fragmenta en distintas partes para abordar de manera detallada los aspectos relevantes del proyecto.

En primer lugar, se realiza un exhaustivo análisis del estado del arte en sistemas de riego automatizado y tecnología IoT aplicada a la agricultura. Se revisan investigaciones previas, avances tecnológicos y tendencias en el campo de la automatización agrícola, destacando los beneficios y desafíos asociados a la implementación de sistemas de riego inteligentes.

Posteriormente, se detallan los materiales y métodos empleados en el desarrollo del proyecto. Se describe el proceso de diseño de la maceta inteligente, incluyendo aspectos como la selección de materiales para el revestimiento y la elección de los componentes electrónicos, con especial énfasis en los sensores de humedad del suelo, temperatura y luminosidad. Además, se presenta un análisis del presupuesto necesario para la adquisición de los materiales y se explica la metodología utilizada en el estudio, desde el diseño del experimento hasta la recopilación y análisis de datos.

En el apartado de desarrollo, se proporciona una visión general del proceso de construcción de la maceta inteligente, incluyendo detalles técnicos sobre la instalación de los sensores, la programación del microcontrolador y la integración de los distintos componentes electrónicos. Se describen los pasos seguidos para ensamblar y configurar la maceta, así como las pruebas realizadas para verificar su funcionamiento correcto.

En el siguiente apartado, se presentan las conclusiones obtenidas a partir del desarrollo del proyecto, así como las posibles líneas de trabajo futuras para mejorar y ampliar la funcionalidad de la maceta inteligente. Se abordan aspectos como la optimización de la maqueta y el código, el registro y procesado atemporal de los datos, la inclusión de nuevos sensores y la adición de funcionalidades extra.

Finalmente, se incluye una lista de las fuentes bibliográficas utilizadas para la investigación y el desarrollo del proyecto, así como información adicional relevante en los anexos, como el código fuente, diagrama de Gantt , esquema electrónicos y los planos de la maceta inteligente.

# Capítulo 3. Memoria

## 3.1. - Estado del arte

Este estudio se enmarca en el contexto de la automatización del cuidado de plantas, un campo emergente que busca integrar tecnología y biología para mejorar la salud y el crecimiento de las plantas en entornos controlados en un ambiente doméstico. A continuación, presentamos una revisión de la literatura existente acerca del tema.

Recientemente, en 2023, se publicó uno de los artículos que mejor ilustra la situación de la automatización de la agricultura es *Design and validation of an open-sourced automation system for vertical farming* (1) escrito publicado en la revista *HardwareX* por la Universidad de Cambridge. En él, se desarrolla un sistema de automatización llamado *MACARONS2*, diseñado para ofrecer una solución inteligente, modular, escalable, personalizable y más importante, de licencia libre. El diseño aportado es de gran calidad y enfocado a un plano industrial para el crecimiento controlado en masa de plantas.

También, otro artículo relacionado con la agricultura vertical es *Vertical farming: The future of agriculture: A review*(2) que aporta información sobre cómo factores como el cambio climático y la sobrepoblación, entre otros, influyen en la agricultura de la actualidad y los problemas que plantea para el futuro. También se explora la idea de la agricultura vertical en entornos urbanos como solución al inminente problema de la escasez de alimentos.

Toda esta literatura anteriormente expuesta únicamente pone el foco de atención en la industrialización y la aplicación a gran escala de esta práctica. Con ello, nuestro proyecto pretende aportar una dimensión personal e individual a este formato de agricultura inteligente, acercando los beneficios de la intersección entre la tecnología y la naturaleza a los hogares.

## 3.2. - Materiales y métodos

En esta sección abordaremos los diferentes materiales empleados para el desarrollo del proyecto junto con otros aspectos relacionados con los componentes como el presupuesto. También describiremos el proceso atravesado a la hora de elaborar las diferentes partes del trabajo.

### 3.2.1. - Diseño

El objetivo principal de este diseño es crear un entorno óptimo para el crecimiento y la salud de las plantas mediante el uso de tecnologías de automatización y sensores ambientales.

El diseño se fundamenta en la integración de sensores que permiten monitorear de manera precisa las condiciones ambientales en el entorno de cultivo de las plantas. Estos sensores están conectados a un sistema de control inteligente que ajusta automáticamente variables como el riego, la iluminación y la ventilación para satisfacer las necesidades específicas de cada planta.

Además, se consideran aspectos ergonómicos y estéticos en el diseño de la maceta inteligente, asegurando que sea funcional y atractiva tanto para los usuarios como para las plantas. Se exploran materiales duraderos y sostenibles para la construcción de la maceta, así como soluciones de diseño que faciliten el acceso y el mantenimiento de las plantas. El resultante diseño quedaría como indican las siguientes figuras.

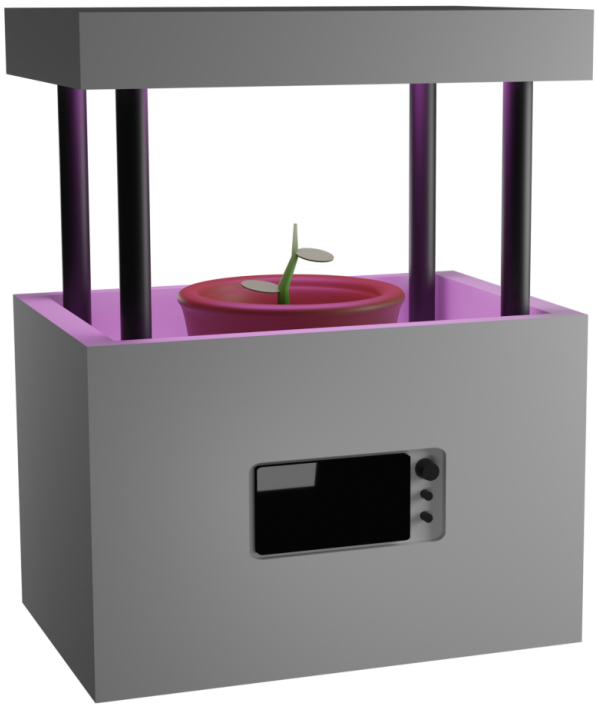


*Figura 1: Renderizado en 3D del modelo de diseño preliminar*

## 3.2.2. - Materiales

En esta sección, exploraremos los materiales necesarios para construir una maceta inteligente que proporcione un entorno óptimo para el crecimiento de las plantas, así como los componentes electrónicos que permitirán su automatización.

### *3.2.2.1. - Materiales del revestimiento*

Para la construcción de la maceta, emplearemos materiales básicos con algún elemento decorativo para mejorar el aspecto visual de la maceta. Para la construcción del cuerpo principal emplearemos madera, pues es un material abundante y con un coste relativamente barato. Construiremos el depósito de agua, situado en la base del cuerpo de la maceta, con planchas de metacrilato. El elemento de transparencia sirve como referencia visual para el rellenado de la maceta así como para mejorar el aspecto exterior. Por último, con tubos de PVC fabricaremos los pilares que sustenten el techo y parte de la estructura.

### *3.2.2.2. - Selección de componentes electrónicos*

Por otra parte, el proyecto requiere de una serie de componentes electrónicos para funcionar. El corazón de nuestro proyecto va a estar por un microcontrolador con el procesador *ESP32-WROOM-32*, un micro controlador de grandes capacidades con funcionalidades Wi-Fi y Bluetooth entre otras. Él se encargará de recibir y manejar los datos generados por los diferentes sensores empleados para monitorizar diferentes métricas. El primero de los sensores va a ser el *DTH22*, un sensor capaz de medir la temperatura y la humedad relativa con un ±0,1% de precisión en sus mediciones. También relacionado con la humedad tenemos el *sensor de humedad de suelo*, este nos proporcionará datos sobre el porcentaje de agua en la tierra, de gran utilidad para determinar si es necesario activar el sistema de riego automático. Para el sistema de riego emplearemos una bomba de agua de 5V genérica. También necesitaremos de un sensor luminoso o *foto resistor* para evaluar la cantidad de luz que recibe nuestra planta y en función de esos datos decidir si encender la luz de crecimiento o no. La luz de crecimiento cuenta con una matriz de LEDs (*Light Emiting Diodes*) que cubren en su totalidad el espectro de luz captada por las hojas. Para mostrar todos estos datos al usuario contaremos con una pantalla LCD (*Liquid Crystal Display*) y un codificador rotatorio (*rotary encoder*) o *knob*. Además, para controlar la cantidad de agua disponible para el sistema de riego contaremos con un sensor de ultrasonidos. La diferencia de voltajes en este

proyecto, 220V por parte de la luz y 5V necesarios para el microcontrolador; será manejado por una fuente de alimentación. La luz de crecimiento presenta un problema adicional, pues esta no puede ser alimentada con 5 voltios y un interruptor de 5V no sería capaz de moverla, por ello, utilizaremos un *Solid State Switch*, un dispositivo muy parecido a los relés pero de mayor fiabilidad y un mejor rendimiento.

Cada material desempeña un papel crucial en la funcionalidad y estética del proyecto. La integración de componentes electrónicos como sensores de humedad del suelo, sistemas de riego automatizados y microcontroladores para la gestión del ciclo de luz garantiza un cuidado óptimo y personalizado para cada planta.

### 3.2.3. - Presupuesto

Debemos tener en cuenta los materiales que necesitamos adquirir para la creación de la maqueta, que pueden ir desde paneles de madera a sensores o distintos tipos de elementos electrónicos. Para ello hemos realizado una tabla en la que aparece representado el coste individual de los materiales empleados así como su categoría.

| Nombre | Tipo | Precio |
|---|---|---|
| Solid State Switch / Relé | Alimentación | 2,47 € |
| fuente de alimentación 5V | Alimentación | 5,43 € |
| ESP32 | controlador | 3,69 € |
| Puerto USB | Sensores | 1,50 € |
| Zumbador | Sensores | 0,23 € |
| Rotary encoder | Sensores | 5,01 € |
| Botones / Botones capacitivos | Sensores | 0,99 € |
| Pantalla | Sensores | 6,25 € |
| Bomba de agua | Sensores | 5,10 € |
| Luz de crecimiento | Sensores | 3,99 € |
| Luminoso | Sensores | 1,00 € |
| Cantidad de agua / ultrasonidos | Sensores | 3,50 € |
| Humedad suelo | Sensores | 0,97 € |
| Temperatura y Humedad | Sensores | 1,67 € |
| Tubos de PVC | Estructura | 0,25 € |
| Tornillos | Estructura | 5,00 € |
| Madera | Estructura | 10,00 € |
| **Total** | | **57,05 €** |

*Tabla 1: Componentes y precios*

Nuestra intención es crear una maqueta con productos que no causasen efectos nocivos en el medioambiente, ya que estamos concienciados con la sostenibilidad de nuestro planeta. Además contamos con un reducido presupuesto, debido a que no disponemos de una capacidad económica alta, por ello fijamos un presupuesto máximo de 50 euros y finalmente sumando todos los componentes, salió un coste final de 40,99 euros, aunque en la tabla aparece un coste superior, eso se debe a que ya disponíamos previamente de algunos productos pero aun así hemos decidido incluirlos en la tabla para que se vea reflejado cual sería el coste total del proyecto a modo de estimación por si alguien quisiera replicarlo.

Como todos los inventos nosotros esperamos obtener una rentabilidad de este trabajo, en este caso más que una rentabilidad económica lo que esperamos es ganar tiempo para otras actividades (ocio, otras labores o descanso) gracias al no emplear nuestro tiempo en el cuidado de las planta.

Para prevenir los futuros beneficios que nos puede traer este proyecto deberíamos utilizar la fórmula de payback, pero como no tenemos intención de obtener beneficios económicos no podemos calcularlo, ya que la fórmula de payback es así.

$$Payback = \frac{Inversión}{Flujos\ de\ caja}$$

*Figura 2: Ecuación payback*

### 3.2.4. - Métodos

La recopilación y análisis de datos juegan un papel fundamental en la metodología, permitiendo una evaluación continua del entorno y la eficacia de las intervenciones automatizadas. Esta sección proporciona una visión general del enfoque metodológico utilizado para desarrollar y evaluar nuestro sistema de automatización del cuidado de plantas.

### *3.2.4.1. - Diseño del estudio*

Este proyecto emplea un enfoque experimental para la automatización del cuidado de plantas utilizando una serie de sensores para monitorear variables ambientales clave. El diseño del estudio incluye la instalación de sensores de humedad y temperatura, sensores de humedad del suelo y sensores de luz en una maceta inteligente.

### *3.2.4.2. - Población y muestra*

Para la recopilación de estos datos emplearemos una planta. Para la realización de este trabajo valdría con cualquier tipo de planta, ya que sus necesidades se verían adecuadamente saciadas por la maceta.

### *3.2.4.3. - Recopilación de datos*

Para adquirir los datos a trabajar se emplearán los sensores mencionados anteriormente en la sección 3.2.2.2: *"Selección de componentes electrónicos"*. Con el sensor DHT22 encargado de recopilar los datos relacionados con la temperatura y humedad relativa, el sensor de humedad de suelo para determinar la cantidad de agua en la tierra y el foto resistor que evaluará la cantidad de luz recibida por la planta.

### *3.2.4.4. - Análisis de datos*

Los datos recopilados por los sensores se registran continuamente y se valorarán instantáneamente por diferentes condicionales para decidir la acción que se ejecutará en función de esa entrada.

### *3.2.4.5. - Limitaciones*

El carácter comercial de los sensores puede influir en su calibración así como la precisión a la hora de recoger los datos. Por añadido contamos con un tiempo limitado para el desarrollo del software y, en general, el desarrollo del proyecto dada su complejidad.

Por otro lado, no disponemos de las herramientas ideales para realizar con precisión los cortes y huecos necesarios en la maqueta, la falta de enrutadoras o una máquina CNC, entre otras, hace que el proceso sea menos preciso.

## 3.3. - Desarrollo

En esta sección, se presenta el proceso de desarrollo del proyecto con un enfoque específico en la programación del sistema y la construcción de la maceta. Esta fase del proyecto representa una etapa crucial en la implementación de la solución propuesta, donde se traducen los conceptos teóricos en soluciones prácticas y tangibles.

El desarrollo del proyecto se lleva a cabo en dos frentes principales: la programación del sistema de automatización y la construcción de la maqueta. La programación implica la codificación de algoritmos y la configuración de dispositivos electrónicos para controlar y monitorear las condiciones ambientales en la maqueta. Por otro lado, la construcción de la maqueta implica la selección de materiales adecuados y la creación de un entorno de cultivo realista que pueda albergar plantas y simular condiciones de luz, humedad y temperatura.

### 3.3.1. - Código

El proceso de programación se basa en la integración de sensores y actuadores para recopilar datos ambientales y realizar acciones automáticas en respuesta a las necesidades de las plantas. Se utiliza un enfoque iterativo de desarrollo de software, donde se realizan pruebas y ajustes continuos para optimizar el rendimiento del sistema y garantizar su eficacia en la gestión del cuidado de las plantas.

El objetivo principal del código es crear un dispositivo que pueda medir las diferentes métricas como la temperatura, la humedad relativa y la humedad del suelo entre otras y mostrar estas lecturas en una pantalla TFT. Se comienza con un diseño funcional de las tareas que se quiere que el código lleve a cabo para posteriormente concretar con la sintaxis apropiada del lenguaje de programación que se emplee, en este caso C++.

Comenzar a construir el programa final desde cero es una práctica muy compleja e induce normalmente error, por ello comenzamos con pruebas de componentes separados para, en un entorno controlado, experimentar con ellos y evaluar su comportamiento y la interacción con ellos.

Una vez todos los componentes han sido probados por separado y se posee un entendimiento de lo que se debe hacer, entonces procedemos a realizar el código que correrá sobre el hardware de nuestra maceta. El desarrollo del código se divide en varias etapas, cada una de las cuales se centra en aspectos específicos del funcionamiento del dispositivo.

Primeramente, se procede con la inicialización de las librerías que se van a emplear. En esta etapa, se importan las bibliotecas necesarias para controlar el sensor DHT, la pantalla TFT y establecer la conexión WiFi. Además, se definen constantes para configurar el pin del sensor DHT, el tipo de sensor, el intervalo de actualización de la pantalla y los detalles de la red WiFi a la que se conectará el dispositivo.

```
#include <Arduino.h>
#include <SPI.h>
#include <TFT_eSPI.h>
#include <Arduino.h>
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>
#include <WiFi.h>
#include "time.h"
```

Seguidamente, incluimos las matrices vectoriales que contienen la información de los iconos que van a ser mostrados en la pantalla. Estas cuentan 64 filas y 64 columnas, en representación de cada píxel del icono. Un ejemplo es el siguiente, se reduce el número de lineas por la cantidad tan grande de estas (el código completo se muestra en el anexo 6.1 :

```
const unsigned char icon__Clock [] PROGMEM = {
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xf8, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x01, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xff, 0xff, 0xe0, 0x00, 0x00,
    0x00, 0x00, 0x1f, 0xff, 0xff, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xe0, 0x0f, 0xfc, 0x00, 0x00,
    0x00, 0x00, 0xff, 0x00, 0x00, 0xfe, 0x00, 0x00, 0x00, 0x01, 0xf8, 0x00, 0x00, 0x3f, 0x80, 0x00,
    0x00, 0x03, 0xf0, 0x00, 0x00, 0x1f, 0xc0, 0x00, 0x00, 0x07, 0xc0, 0x00, 0x00, 0x07, 0xe0, 0x00,
    0x00, 0x0f, 0x80, 0x00, 0x00, 0x03, 0xe0, 0x00, 0x00, 0x1f, 0x00, 0x00, 0x00, 0x01, 0xf0, 0x00,
    0x00, 0x3e, 0x00, 0x00, 0x00, 0x00, 0xf8, 0x00, 0x00, 0x3c, 0x00, 0x03, 0x80, 0x00, 0x7c, 0x00,
    0x00, 0x7c, 0x00, 0x03, 0xc0, 0x00, 0x3c, 0x00, 0x00, 0x78, 0x00, 0x03, 0xc0, 0x00, 0x3e, 0x00,
    0x00, 0xf0, 0x00, 0x03, 0xc0, 0x00, 0x1e, 0x00, 0x00, 0xf0, 0x00, 0x03, 0xc0, 0x00, 0x1f, 0x00,
    0x01, 0xf0, 0x00, 0x03, 0xc0, 0x00, 0x0f, 0x00, 0x01, 0xe0, 0x00, 0x03, 0xc0, 0x00, 0x0f, 0x00,
    0x01, 0xe0, 0x00, 0x03, 0xc0, 0x00, 0x0f, 0x00, 0x03, 0xe0, 0x00, 0x03, 0xc0, 0x00, 0x07, 0x80,
    ...};
```

Después, se declaran las variables que se van a utilizar a lo largo del código. En esta etapa, se inicializan los objetos que representan el sensor DHT y la pantalla TFT.

```
208  TFT_eSPI tft = TFT_eSPI();
209  TFT_eSprite img = TFT_eSprite(&tft);
```

Una vez hecho esto, se explicitan las funciones creadas para poder utilizarlas a lo largo del código.

```
348  float getSoilMoisture() {
349      value = analogRead(SenPin);
350      if (value > 1000) { if (value > highest) { highest = value; } }
351      else { if (value < lowest) { lowest = value; } }
352      int mapped = map(value, lowest, highest, 100, 0);
353     return mapped;
354  }
```

A continuación, se realiza la configuración del entorno. También se establece la comunicación serial para la depuración (*debugging)* y se configuran algunos parámetros como la rotación de la pantalla TFT según sea necesario.

```
589  void setup() {
590    Serial.begin(9600);
591
592      //DHT22 setup
593      dht.begin();
594      sensor_t sensor;
595      dht.temperature().getSensor(&sensor);
596      dht.humidity().getSensor(&sensor);
597      dhtDelayMS = 2000;
648      ...}
```

Por último se inicializa la sección *loop()* la cual se ejecutará continuamente una vez todo lo mencionado anteriormente se haya ejecutado. En esta Se evalúan los parámetros de los sensores así como la posición del *rotary encoder* para dibujar el

menú y las diferentes pantallas. Esta parte del código es la más corta, ya que el mayor peso del código recae sobre las funciones creadas para interceptar y procesar los diferentes eventos.

```
654  void loop() {
655
656    NewStateCLK = digitalRead(CLKPin);
657
658   if (NewStateCLK != OldStateCLK) {
659   if (itemPage == 0) {
660
661            if (digitalRead(DTPin) == NewStateCLK) {counter++;}
662            else {counter--;}
663            if (counter < 0) {counter = iconArray_LEN - 1;}
664            if (counter > iconArray_LEN - 1) {counter = 0;}
665
666            current_icon = counter;
667
668            previous_icon = current_icon - 1;
669            if (previous_icon < 0) {previous_icon = iconArray_LEN - 1;}
670            next_icon = current_icon + 1;
671            if (next_icon > iconArray_LEN - 1) {next_icon = 0;}
672            drawSelectionMenu();
673            }
674   }
675
676   OldStateCLK = NewStateCLK;
677
678   if (digitalRead(button) != 1) {
679   handleButtonPress();
680   printSelectionMenuLogic();
681   printSelectedItemLogic();
682    }
683  }
```

Finalmente, todo el código se encuentra en el anexo 6.1, y también se puede encontrar en el repositorio de Github del proyecto.

### 3.3.2. - Construcción de la maceta

La creación de la maqueta requirió un enfoque meticuloso que abarcó desde el diseño inicial en AutoCAD hasta el ensamblaje final de todos los componentes. En esta sección, se detallará el proceso paso a paso, desde la concepción de los planos hasta las pruebas y ajustes realizados para garantizar el funcionamiento óptimo del dispositivo. Se explorará el diseño de los planos en AutoCAD, la selección de materiales y componentes, el corte y preparación de los materiales, el ensamblaje de la estructura y las pruebas exhaustivas llevadas a cabo para validar el rendimiento de la maqueta. Este análisis proporcionará una visión completa del desarrollo del proyecto y destacará los desafíos y decisiones clave que se enfrentaron en el camino hacia la creación de la maqueta funcional y estéticamente satisfactoria.

Primeramente, el proceso de desarrollo de la maqueta comenzó con el diseño detallado de los planos en AutoCAD. Se elaboraron planos que incluían las dimensiones precisas de cada componente, así como las ubicaciones de los orificios de montaje y las rutas de los cables. Se prestó especial atención a la disposición de los componentes para garantizar una estructura compacta y funcional. Aunque estos han sido modificado en varias ocasiones para resolver conflictos que surgieron durante el proceso.

Una vez preparados todos los materiales, se procedió al ensamblaje de la estructura de la maqueta. Se unieron los tableros de madera cortados mediante tornillos, asegurando una unión sólida y duradera. Se montaron los componentes electrónicos en sus ubicaciones designadas y se conectaron entre sí siguiendo las rutas de cableado previamente planificadas según el esquema electrónico en el anexo 6.3.

Finalmente, tras corregir muchos errores en el prototipo y buscar soluciones para lograr el ensamblaje preciso de la estructura y la realización de pruebas exhaustivas para garantizar un funcionamiento óptimo del dispositivo, el resultado final es una maqueta funcional y estéticamente atractiva que cumple con los objetivos establecidos.

## 3.4. - Resultados

Hasta el momento, lamentablemente, no se ha dispuesto del tiempo necesario para llevar a cabo una comparación exhaustiva entre los resultados obtenidos mediante la automatización implementada y el método tradicional de ejecución de tareas. La complejidad del proyecto y los recursos limitados han impedido realizar pruebas completas que permitan evaluar con precisión el rendimiento y la eficiencia del sistema automatizado en comparación con los métodos convencionales.

Sin embargo, a pesar de esta limitación, el diseño y la implementación del sistema automatizado han generado expectativas prometedoras. La optimización de los procesos mediante la automatización tiene el potencial de mejorar significativamente la velocidad, la precisión y la consistencia en la ejecución de tareas, lo que podría traducirse en un aumento de la productividad y la eficiencia en el futuro.

Se espera que, una vez se disponga del tiempo necesario para realizar pruebas y comparaciones rigurosas, se puedan obtener resultados más concretos y cuantificables que validen la eficacia y el valor agregado de la automatización frente al método tradicional. Esta fase de evaluación y análisis será fundamental para identificar áreas de mejora y optimización continua del sistema automatizado, con miras a maximizar sus beneficios y su impacto en el contexto específico de aplicación.

# Capítulo 4. Conclusiones y futuras líneas de trabajo

El proyecto ha demostrado la viabilidad y el potencial de la automatización para mejorar la eficiencia y la comodidad en la vida diaria. La integración de dispositivos conectados y sistemas inteligentes ha permitido optimizar procesos y simplificar tareas, brindando una solución efectiva a un problema concreto.

Además, se ha destacado la importancia del diseño y la planificación cuidadosa en el desarrollo de sistemas automatizados. Desde la concepción de la idea hasta la implementación final, cada paso ha requerido un análisis detallado y una consideración minuciosa de los requisitos y las limitaciones del proyecto.

Lejos de explotar al máximo las posibilidades de este proyecto y llegar hasta el final, con todo lo que ello implica, este trabajo ha sido muy satisfactorio de realizar, pues, a parte de realizar su función y solucionar un problema de la vida cotidiana, ha servido como una gran vía de aprendizaje y de ahondamiento en temas como el *Internet of Things*, la automatización y la electrónica en general.

Por tanto, en vistas a todos los posibles caminos aún por explorar y las amplias posibilidades en el futuro desarrollo del proyecto, he de mencionar algunas de las futuras líneas de trabajo.

## 4.1. - Optimización de la maqueta

El formato del prototipo es únicamente un indicativo lejos del modelo refinado diseñado para la producción en masa. Las limitaciones técnicas y de materias primas reducen las posibilidades de diseño drásticamente, teniendo que optar por adoptar un diseño más simple y fácil de ejecutar con herramientas cotidianas, evitando, por tanto, elementos que pudieran elevar la complejidad del diseño.

Por otro lado, la innegable existencia de plantas de diferentes tamaños implica a su vez la necesidad de incorporar nuevos diseños en relación a las dimensiones de la planta que se desea alojar dentro del sistema.

## 4.2. - Optimización del código

La inexperiencia como programador, especialmente al enfrentarse por primera vez al lenguaje C++, ha resultado en fallos en la estructura del programa. Estos errores no solo afectan la eficiencia y la legibilidad del código, sino que también dificultan su expansión y modificación futuras. La falta de comprensión de los principios fundamentales de la programación y las mejores prácticas ha contribuido a esta situación.

Se espera una futura mejora en el código que mejore significativamente la calidad y la capacidad de mantener el código actualizado, facilitando su expansión y modificación futuras. Al mejorar la estructura y la organización del código, se reducirá la complejidad y el riesgo de errores, lo que permitirá un desarrollo más eficiente y una mayor flexibilidad en la implementación de nuevas funcionalidades.

## 4.3. - Registro y procesado atemporal de los datos

Los datos generados por los sensores, una vez procesados, son desechados por el programa una vez se ejecuta de nuevo. Esto podría solucionarse Mediante la implementación de una pequeña unidad de almacenamiento local, como una pequeña tarjeta SD, que periódicamente enviara todos estos datos a una base de datos remota alojada en la nube mediante plataformas como ThingSpeak o mediante alojamiento de un servidor local conectado a internet.

Esto nos llevaría al siguiente nivel, el procesado posterior de los datos. Con la inmensa cantidad de datos generados por el dispositivo, una vez guardados y correctamente tratados, estos podrían atravesar una segunda fase de evaluación, mucho menos apresurada y más refinada. La inclusión de un algoritmo de aprendizaje(3), mediante el uso de *machine learning,* aplicado a los datos obtenidos podría potencialmente añadir una capa extra de rendimiento y optimización, posibilitando un mejor cuidado de la planta a través de la predicción de factores como la hora ideal para comenzar a iluminar artificialmente la planta, detectar cuándo la va a necesitar ser regada u otros factores de gran importancia.

## 4.4. - Inclusión de nuevos sensores

La cantidad de sensores en el proyecto es mínima, para realizar una maceta inteligente capaz de ser un producto comercial completo esta debe de contar con sensores no incluidos que aporten información crucial como la cantidad de agua en el depósito.

## 4.5. - Adición de funcionalidades extra

Del mismo modo, se presenta un número reducido de capacidades con las que cuenta la maceta. El menú principal puede ser ampliado con nuevas pantallas, como una página en la que se mostrara una cara simple formada por un par de puntos por ojos y una línea por boca que reaccionara en función de los datos recogidos por los sensores, mostrando calor si la temperatura es muy alta, tiritando si hace frío o cerrando los ojos si está recibiendo mucha luz.

También cabría la posibilidad de añadir nuevas características como la regulación de la altura del techo por medio de motores, con el objetivo de poder introducir un mayor número de variedad de plantas así como para acompañarlas durante su crecimiento, u otras posibilidades.

En última instancia, este proyecto es solo el comienzo de un viaje continuo hacia la exploración y la innovación en el campo de la automatización y el IoT. Con un enfoque en la mejora continua y la adaptación a las tendencias tecnológicas emergentes, se pueden lograr avances significativos en la creación de soluciones inteligentes y eficientes para mejorar la calidad de vida y la productividad en diversos contextos.

# Capítulo 5. Bibliografía

1: Vijja Wichitwechkarn, William Rohde, Ruchi Choudhary, Design and validation of an open-sourced automation system for vertical farming, 2023

2: Mohd Salim Mir, Nasir Bashir Naikoo, Raihana Habib Kanth, FA Bahar, M Anwar Bhat, Aijaz Nazir, S Sheraz Mahdi, Zakir Amin, Lal Singh, Waseem Raja, AA Saad, Tauseef A Bhat, Tsultim Palmo and Tanveer A Ahngar, Vertical farming: The future of agriculture: A review, 2022

3: Pedro Domingos, The Master Algorithm, 2015

# Capítulo 6. Anexos

# 6.1. - Código

```
#include <Arduino.h>
#include <SPI.h>
#include <TFT_eSPI.h>
#include <Arduino.h>
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>
#include <WiFi.h>
#include "time.h"

// ' Clock', 64x64px
const unsigned char icon__Clock [] PROGMEM = {
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xf8, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x01, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xff, 0xff, 0xe0, 0x00, 0x00,
    0x00, 0x00, 0x1f, 0xff, 0xff, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xe0, 0x0f, 0xfc, 0x00, 0x00,
    0x00, 0x00, 0xff, 0x00, 0x00, 0xfe, 0x00, 0x00, 0x00, 0x01, 0xf8, 0x00, 0x00, 0x3f, 0x80, 0x00,
    0x00, 0x03, 0xf0, 0x00, 0x00, 0x1f, 0xc0, 0x00, 0x00, 0x07, 0xc0, 0x00, 0x00, 0x07, 0xe0, 0x00,
    0x00, 0x0f, 0x80, 0x00, 0x00, 0x03, 0xe0, 0x00, 0x00, 0x1f, 0x00, 0x00, 0x00, 0x01, 0xf0, 0x00,
    0x00, 0x3e, 0x00, 0x00, 0x00, 0x00, 0xf8, 0x00, 0x00, 0x3c, 0x00, 0x03, 0x80, 0x00, 0x7c, 0x00,
    0x00, 0x7c, 0x00, 0x03, 0xc0, 0x00, 0x3c, 0x00, 0x00, 0x78, 0x00, 0x03, 0xc0, 0x00, 0x3e, 0x00,
    0x00, 0xf0, 0x00, 0x03, 0xc0, 0x00, 0x1e, 0x00, 0x00, 0xf0, 0x00, 0x03, 0xc0, 0x00, 0x1f, 0x00,
    0x01, 0xf0, 0x00, 0x03, 0xc0, 0x00, 0x0f, 0x00, 0x01, 0xe0, 0x00, 0x03, 0xc0, 0x00, 0x0f, 0x00,
    0x01, 0xe0, 0x00, 0x03, 0xc0, 0x00, 0x0f, 0x00, 0x03, 0xe0, 0x00, 0x03, 0xc0, 0x00, 0x07, 0x80,
    0x03, 0xc0, 0x00, 0x03, 0xc0, 0x00, 0x07, 0x80, 0x03, 0xc0, 0x00, 0x03, 0xc0, 0x00, 0x07, 0x80,
    0x03, 0xc0, 0x00, 0x03, 0xc0, 0x00, 0x07, 0x80, 0x03, 0xc0, 0x00, 0x03, 0xc0, 0x00, 0x07, 0x80,
    0x03, 0xc0, 0x00, 0x03, 0xc0, 0x00, 0x07, 0x80, 0x03, 0xc0, 0x00, 0x03, 0xf0, 0x00, 0x07, 0x80,
    0x03, 0xc0, 0x00, 0x01, 0xf8, 0x00, 0x07, 0x80, 0x03, 0xc0, 0x00, 0x00, 0xfe, 0x00, 0x07, 0x80,
    0x03, 0xc0, 0x00, 0x00, 0x7f, 0x00, 0x07, 0x80, 0x03, 0xe0, 0x00, 0x00, 0x1f, 0x80, 0x07, 0x80,
    0x01, 0xe0, 0x00, 0x00, 0x0f, 0x80, 0x0f, 0x00, 0x01, 0xe0, 0x00, 0x00, 0x07, 0x80, 0x0f, 0x00,
    0x01, 0xe0, 0x00, 0x00, 0x01, 0x00, 0x0f, 0x00, 0x00, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x1f, 0x00,
    0x00, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x1e, 0x00, 0x00, 0xf8, 0x00, 0x00, 0x00, 0x00, 0x3e, 0x00,
    0x00, 0x7c, 0x00, 0x00, 0x00, 0x00, 0x3c, 0x00, 0x00, 0x3c, 0x00, 0x00, 0x00, 0x00, 0x7c, 0x00,
    0x00, 0x3e, 0x00, 0x00, 0x00, 0x00, 0xf8, 0x00, 0x00, 0x1f, 0x00, 0x00, 0x00, 0x01, 0xf0, 0x00,
    0x00, 0x0f, 0x80, 0x00, 0x00, 0x03, 0xf0, 0x00, 0x00, 0x0f, 0xc0, 0x00, 0x00, 0x07, 0xe0, 0x00,
    0x00, 0x07, 0xe0, 0x00, 0x00, 0x0f, 0xc0, 0x00, 0x00, 0x03, 0xf8, 0x00, 0x00, 0x3f, 0x80, 0x00,
    0x00, 0x00, 0xfe, 0x00, 0x00, 0xff, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xc0, 0x07, 0xfc, 0x00, 0x00,
    0x00, 0x00, 0x3f, 0xff, 0xff, 0xf8, 0x00, 0x00, 0x00, 0x00, 0x0f, 0xff, 0xff, 0xe0, 0x00, 0x00,
    0x00, 0x00, 0x03, 0xff, 0xff, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xf8, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};
// ' Face', 64x64px
const unsigned char icon__Face [] PROGMEM = {⋯
// ' Luz', 64x64px
const unsigned char icon__Luz [] PROGMEM = {⋯
// ' Menu', 64x64px
const unsigned char icon__Menu [] PROGMEM = {⋯
// ' Riego', 64x64px
const unsigned char icon__Riego [] PROGMEM = {⋯

// Array of all bitmaps for convenience. (Total bytes used to store images in PROGMEM = 2640)
const int iconArray_LEN = 5;
const unsigned char* iconArray[5] = {
    icon__Clock,
```

```
    icon_Face,
    icon_Luz,
    icon_Menu,
    icon_Riego
};


char iconNameArray[iconArray_LEN] [20]= {
    {"CLOCK"},
    {"LIGHT"},
    {"LUZ"},
    {"MENU"},
    {"WATERING"}
};

//Screen selection variables

TFT_eSPI tft = TFT_eSPI();
TFT_eSprite img = TFT_eSprite(&tft);

int current_icon = 0;
int previous_icon = iconArray_LEN - 1;
int next_icon = current_icon + 1;
bool itemPage = 0;

//ItemPage variables
#define lightPin 12
bool on_off = 0;

//Rotary encoder inclusion variables
#define DTPin 21
#define CLKPin 19
#define button 22

int OldStateCLK;
int NewStateCLK;
int counter = 0;

unsigned long startTime = 0;
unsigned long pressedTime = 0;
int lowPressedThreshold = 300;
int topPressedThreshold = 2000;

//PWM variables
int pwmPin = 2;
int pwmChannel = 0;
int pwmRes = 8;
int pwdFrequency = 1000;
int dt = 5;

//DHT22 variables
#define DHTPIN 32
#define DHTTYPE DHT22
DHT_Unified dht(DHTPIN, DHTTYPE);
int dhtDelayMS;

//SoilMoisture variables
```

```
#define SenPin 34
int value;
int highest = 0;
int lowest = 1000;

// Riego variables
int wateringTimeStart = 0;
int wateringTimePressed = 0;
int wateringProgressBar = 6;
int conversionVal;

//wifi variables
// const char* ssid       = "      ";
// const char* password   = "          ";
const char* ssid       = "      ";
const char* password   = "       ";
// const char* ssid       = "       ";
// const char* password   = "              ";
const char* ntpServer = "pool.ntp.org";
const long  gmtOffset_sec = 3600;
const int   daylightOffset_sec = 3600;
int tryCount = 0;
int tryQuantity = 50;

//////////////////////
//   ButtonPress    //
//////////////////////

void handleButtonPress() {
    startTime = millis();
    while (digitalRead(button) == 0) {}
    delay(10);
    pressedTime = millis() - startTime;
}

//////////////////////
// WatertingProgress //
//////////////////////

int MillisToProgress(int millisProgressTime) {
    conversionVal = 256*millisProgressTime/5000;
    return conversionVal;
}


//////////////////////
//      Wi-Fi       //
//////////////////////

void imgPrintLocalTime()
{
  struct tm timeinfo;
  if(!getLocalTime(&timeinfo)){
    img.setTextSize(1);
    img.setCursor(0,0);
    img.print("Failed to obtain time");
    return;
```

```
  }
  img.setTextSize(5);
  img.setCursor(40,90);
  img.println(&timeinfo, "%H:%M:%S");
  img.setTextSize(2);
  img.setCursor(25,140);
  img.println(&timeinfo, "%A, %B %d %Y");
}


//////////////////////
//      DHT22       //
//////////////////////
float getTemperature() {
    sensors_event_t event;
    dht.temperature().getEvent(&event);
    if (isnan(event.temperature)) {
        return 0;
    }
    else {
        return int(event.temperature);
    }
}

int temperatureToProgress(int temperature) {
    int progress = 138*temperature/50;
    return progress;
}

float getHumidity() {
    sensors_event_t event;
    dht.humidity().getEvent(&event);
    if (isnan(event.relative_humidity)) {
        return 0;
    }
    else {
        return int(event.relative_humidity);
    }
}
//////////////////////
//  SoilMoisture    //
//////////////////////

float getSoilMoisture() {
    value = analogRead(SenPin);
    if (value > 1000) { if (value > highest) { highest = value; } }
    else { if (value < lowest) { lowest = value; } }
    int mapped = map(value, lowest, highest, 100, 0);
    return mapped;
}

////////////////////////////
//Startup animation function//
////////////////////////////
void createImgSprite() {
    img.setColorDepth(8);
    img.createSprite(320,240);
```

```
    img.createSprite(320,240);
    img.fillSprite(TFT_BLACK);
    img.setTextColor(TFT_WHITE);
}

void startupAnimation(){

    tft.fillScreen(TFT_BLACK);
    tft.setTextColor(TFT_WHITE);
    tft.setTextSize(6);
    tft.drawCentreString("MA-z",160,100,1);

    //TFT initiation animation// (turn into a function and call it out in the setup a
    for (int k = 0; k < 3; k++){

    for (int i = 0; i < 255; i++){
        ledcWrite(pwmChannel, i);
        delay(dt);
    }
    for (int j = 255; j >= 0; j--){
        ledcWrite(pwmChannel, j);
        delay(dt);
    }
    delay(100);
    }

    delay(1000);
    tft.fillScreen(TFT_BLACK);

    ledcWrite(pwmChannel, 255);
}

////////////////////////////
//Drawing the selection menu//
////////////////////////////

void drawSelectionMenu() {

    img.setColorDepth(8);
    img.createSprite(320,240);
    img.fillSprite(TFT_BLACK);
    img.setTextColor(TFT_WHITE);
    img.setTextSize(4);

    //Previous icon
    img.drawBitmap(16,8,iconArray[previous_icon],64,64,TFT_WHITE, TFT_BLACK);
    img.drawString(iconNameArray[previous_icon],100,22);

    //Current icon
    img.drawBitmap(16,88,iconArray[current_icon],64,64,TFT_WHITE, TFT_BLACK);
    img.drawString(iconNameArray[current_icon],100,102);

    //Next icon
    img.drawBitmap(16,168,iconArray[next_icon],64,64,TFT_WHITE, TFT_BLACK);
    img.drawString(iconNameArray[next_icon],100,184);

    //Selection square
```

```
418     // img.drawBitmap(0,80,icon_Select,64,64,TFT_WHITE);
419     img.drawRoundRect(0,80,320,80,5,TFT_WHITE);
420
421     img.pushSprite(0,0, TFT_TRANSPARENT);
422     img.deleteSprite();
423
424 }
425
426 void printSelectionMenuLogic() {
427     if (itemPage == 1 && pressedTime > lowPressedThreshold && pressedTime < topPresse
428         itemPage = 0;
429         drawSelectionMenu();
430     }
431 }
432
433 ////////////////////////////
434 //Drawing the specified item//
435 ////////////////////////////
436
437 void drawSelectedItem(int selectedItem) {
438
439     img.setTextSize(4);
440
441     //clock
442     while (itemPage == 1 && selectedItem == 0) {
443         createImgSprite();
444         imgPrintLocalTime();
445         img.pushSprite(0,0, TFT_TRANSPARENT);
446         img.deleteSprite();
447         handleButtonPress();
448         printSelectionMenuLogic();
449     }
450
451     //face
452     while (selectedItem == 1 && itemPage == 1) {
453         createImgSprite();
454         img.drawCentreString("Ups...", 160, 110, 1);
455         handleButtonPress();
456         printSelectionMenuLogic();
457         img.pushSprite(0,0, TFT_TRANSPARENT);
458         img.deleteSprite();
459     }
460
461     //luz
462     while (selectedItem == 2 && itemPage == 1) {
463         createImgSprite();
464         img.drawRoundRect(80,56,160,80,40,TFT_WHITE);
465         NewStateCLK = digitalRead(CLKPin);
466         if(NewStateCLK != OldStateCLK) {
467             if (current_icon == 2 && itemPage == 1) {
468                 if (on_off == 0) {
469                     on_off = 1;
470                 }
471                 else {
472                     on_off = 0;
473                 }
474             }
```

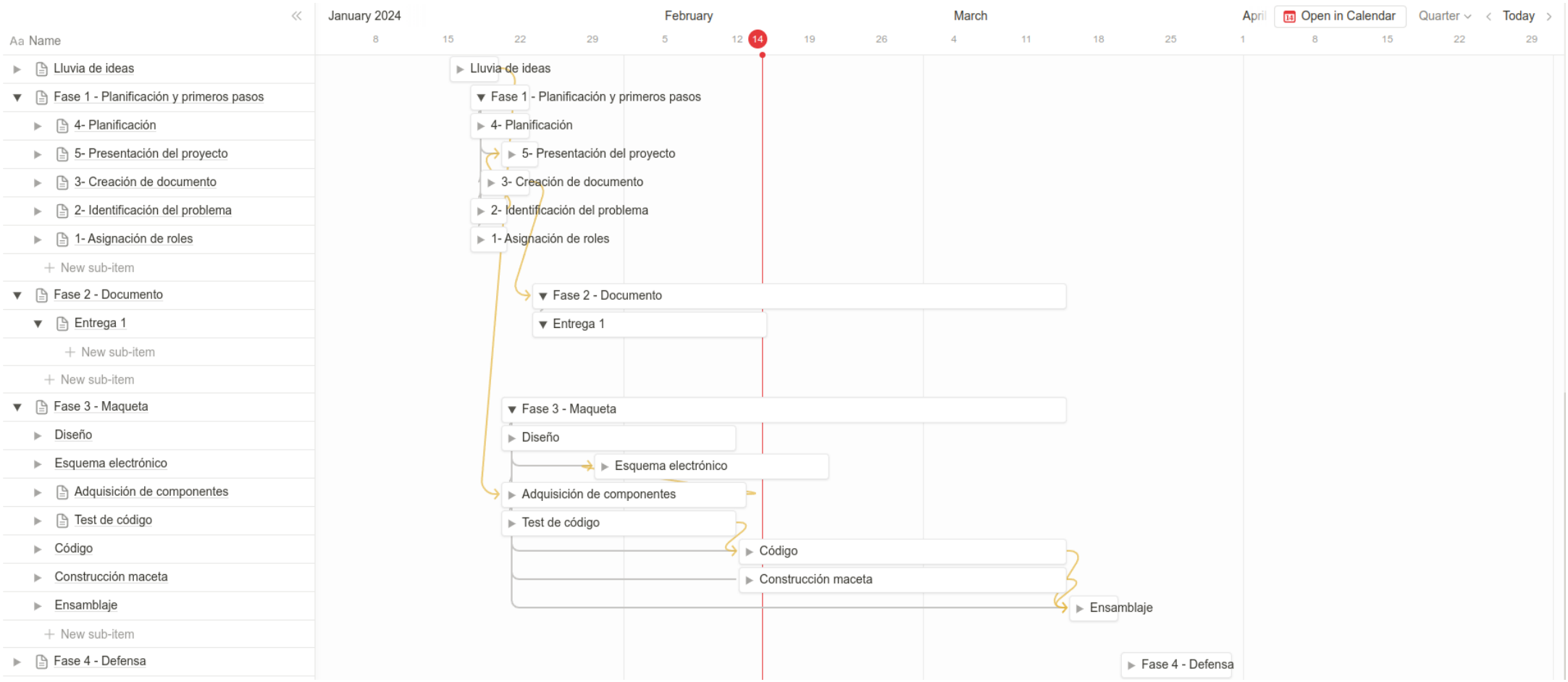
```
475             Serial.println(on_off);
476         }
477         OldStateCLK = NewStateCLK;
478         if (on_off == 0) {
479             img.fillCircle(84+35,61+35,35,TFT_WHITE);
480             img.setTextSize(5);
481             img.drawCentreString("OFF",160,148+18,1);
482         }
483         if (on_off == 1) {
484             img.fillCircle(166+35,61+35,35,TFT_WHITE);
485             img.setTextSize(5);
486             img.drawCentreString("ON",160,148+18,1);
487         }
488         digitalWrite(lightPin,on_off);
489         img.pushSprite(0,0, TFT_TRANSPARENT);
490         img.deleteSprite();
491         handleButtonPress();
492         printSelectionMenuLogic();
493     }
494     //menu
495     while (selectedItem == 3 && itemPage == 1) {
496         createImgSprite();
497 
498         char temperature = getTemperature();
499 
500         //Temperatura (arriba largo)
501         img.drawRoundRect(15,10,290,100,20,TFT_WHITE);
502         img.setTextSize(2);
503         img.drawString("TEMPERATURA",32,28,2);
504         img.drawCircle(48,76,16,TFT_WHITE);
505         img.drawRoundRect(56,69,140,14,7,TFT_WHITE);
506         img.fillCircle(48,76,15,TFT_BLACK);
507         img.fillRoundRect(57,70,138,12,6,TFT_BLACK);
508         img.setTextSize(5);
509         // img.drawString("25",218,42);
510         img.drawFloat(getTemperature(),0,220,42);
511         img.setTextSize(3);
512         img.drawString("C",278,42);
513         //--------Thermometer-----------
514         img.fillCircle(48,76,14,TFT_RED);
515         img.fillRoundRect(58,71,temperatureToProgress(getTemperature()),10,5,TFT_RED)
516         //--------------------------------
517 
518         // Humedad relativa (abajo izquierda)
519         img.drawRoundRect(15,120,140,100,20,TFT_WHITE);
520         img.drawRoundRect(165,120,140,100,20,TFT_WHITE);
521         img.setTextSize(2);
522         img.drawCentreString("H.RELATIVA",85,140,1);
523         img.drawCentreString("H. SUELO",235,140,1);
524         img.setTextSize(5);
525         img.drawFloat(getHumidity(),0,40,170);
526         img.drawCentreString("%",120,170,1);
527         img.drawFloat(getSoilMoisture(),0,194,170);
528         img.drawString("%",254,170,1);
529         handleButtonPress();
530         printSelectionMenuLogic();
531         img.pushSprite(0,0, TFT_TRANSPARENT);
```

```
532         img.deleteSprite();
533         delay(500);
534 
535     }
536     //riego
537     while (selectedItem == 4 && itemPage == 1) {
538         createImgSprite();
539         img.drawRoundRect(30,94,260,20,10,TFT_WHITE);
540         img.setTextSize(2);
541         img.drawCentreString("Preiona 5s para regar",160,138,1);
542         startTime = millis();
543         wateringTimeStart = millis();
544         while (digitalRead(button) == 0) {
545             // x [0,256]
546             createImgSprite();
547             img.drawRoundRect(30,94,260,20,10,TFT_WHITE);
548             img.setTextSize(2);
549             img.drawCentreString("Preiona 5s para regar",160,138,1);
550 
551             wateringTimePressed = millis() - wateringTimeStart;
552             wateringProgressBar = MillisToProgress(wateringTimePressed);
553             // Serial.print("progrestime: ");
554             // Serial.println(wateringTimePressed);
555             wateringProgressBar = MillisToProgress(wateringTimePressed);
556             img.fillRoundRect(32,96,wateringProgressBar,16,8,TFT_SKYBLUE);
557             if (wateringProgressBar >= 256) {
558                 digitalWrite(lightPin,HIGH);
559                 delay(1000);
560                 digitalWrite(lightPin,LOW);
561                 delay(1000);
562                 break;
563             }
564             img.pushSprite(0,0, TFT_TRANSPARENT);
565             img.deleteSprite();
566         }
567         wateringProgressBar = 0;
568         pressedTime = millis() - startTime;
569         printSelectionMenuLogic();
570         img.pushSprite(0,0, TFT_TRANSPARENT);
571         img.deleteSprite();
572     }
573 
574 
575 }
576 
577 void printSelectedItemLogic() {
578     if (itemPage == 0 && pressedTime < lowPressedThreshold) {
579         itemPage = 1;
580         drawSelectedItem(current_icon);
581     }
582 }
583 
584 
585 ////////////////////
586 //      Setup       //
587 ////////////////////
588 
```

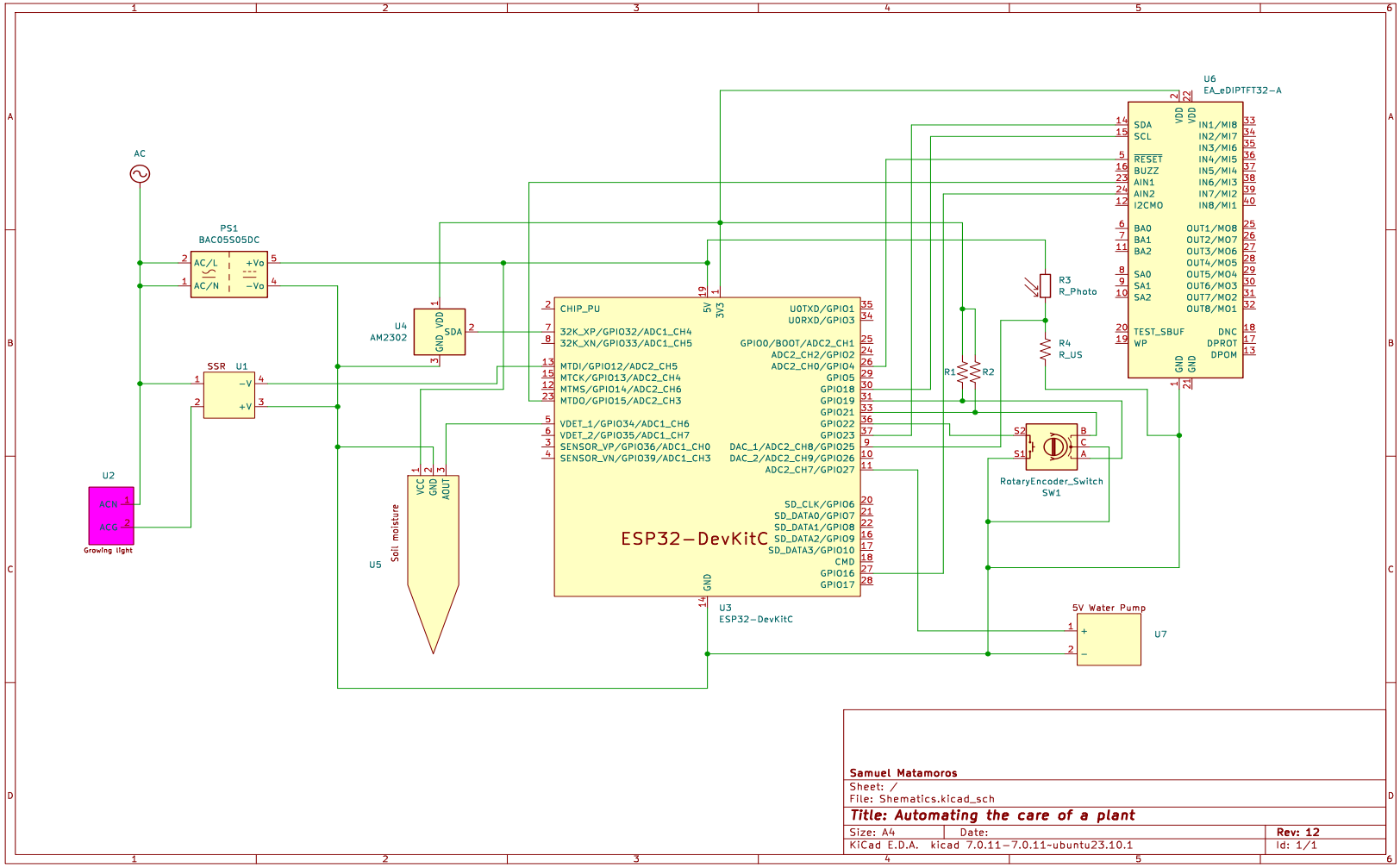
```
590  {
591      Serial.begin(9600);
592
593      //DHT22 setup
594      dht.begin();
595      sensor_t sensor;
596      dht.temperature().getSensor(&sensor);
597      dht.humidity().getSensor(&sensor);
598      dhtDelayMS = 2000;
599
600      //SoilMoisture setup
601      pinMode(SenPin, INPUT);
602
603      //knob setup//
604      pinMode(DTPin, INPUT);
605      pinMode(CLKPin, INPUT);
606      pinMode(button, INPUT);
607
608      OldStateCLK = digitalRead(CLKPin);
609
610      //PWM setup//
611      ledcSetup(pwmChannel, pwdFrequency, pwmRes);
612      ledcAttachPin(pwmPin, pwmChannel);
613
614      //Growing light setup//
615      pinMode(lightPin,OUTPUT);
616
617      //TFT setup//
618      tft.init();
619      tft.setRotation(3);
620      tft.fillScreen(TFT_BLACK);
621      ledcWrite(pwmChannel,255);
622
623      // Connect to NTP
624      tft.setTextSize(1);
625      tft.setTextColor(TFT_WHITE);
626      tft.setCursor(0,0);
627      tft.printf("Connecting to %s ", ssid);
628      WiFi.mode(WIFI_STA);
629      WiFi.begin(ssid, password);
630      while (WiFi.status() != WL_CONNECTED) {
631          delay(500);
632          tft.print(".");
633      }
634      //init and get the time
635      tft.println(" CONNECTED");
636      tft.println("Conecting to NTP server");
637      tft.println("Configuring time");
638      configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
639      tft.println("Time configured!");
640      imgPrintLocalTime();
641      WiFi.disconnect(true);
642      WiFi.mode(WIFI_OFF);
643      tft.fillScreen(TFT_BLACK);
644
645
646      startupAnimation();
```

```
647     drawSelectionMenu();
648 }
649
650 //////////////////////
651 //      Loop        //
652 //////////////////////
653
654 void loop()
655 {
656
657     NewStateCLK = digitalRead(CLKPin);
658
659     if (NewStateCLK != OldStateCLK) {
660         if (itemPage == 0) {
661
662             if (digitalRead(DTPin) == NewStateCLK) {counter++;}
663             else {counter--;}
664             if (counter < 0) {counter = iconArray_LEN - 1;}
665             if (counter > iconArray_LEN - 1) {counter = 0;}
666
667             current_icon = counter;
668
669             previous_icon = current_icon - 1;
670             if (previous_icon < 0) {previous_icon = iconArray_LEN - 1;}
671             next_icon = current_icon + 1;
672             if (next_icon > iconArray_LEN - 1) {next_icon = 0;}
673             drawSelectionMenu();
674
675             }
676
677     }
678
679     OldStateCLK = NewStateCLK;
680
681     if (digitalRead(button) != 1) {
682         handleButtonPress();
683         printSelectionMenuLogic();
684         printSelectedItemLogic();
685     }
686
687 }
```

# 6.2. - Diagrama de Gannt / Organigrama



# 6.3. - Esquema electrónico



# 6.4. - Planos