**NORMAS BÁSICAS DE ESTILO Y FORMATO**

El texto del trabajo se escribirá preferentemente en español. Se admitirá el idioma inglés, siempre que el trabajo esté avalado por un convenio internacional (Erasmus u otros) o si la Comisión delegada de Trabajos Fin de Grado o la Comisión Académica del Máster, según el caso, informe favorablemente a partir de una solicitud previa del alumno o alumna.

**PROYECTOS DE INGENIERÍA**

Se seguirán las consideraciones indicadas en la norma UNE 157001:2004 en cuanto a requisitos formales de carácter general con qué deben redactarse los proyectos.

Se utilizará la carátula incluida en este anexo.

Serán elementos de obligada aparición los siguientes:

**Resumen.** Entre 300 y 500 palabras como referencia.

**Documento Nº 1. Índice**

**Documento Nº 2. Memoria.** Tiene como misión justificar las soluciones adoptadas, su adecuación a la normativa legal aplicable y, conjuntamente con los planos y el pliego de condiciones, debe describir de forma unívoca el objeto del Proyecto.

**Documento Nº 3. Anexos**

Este documento debe contener los anejos necesarios (según proceda en cada caso) correspondientes a:

* **Documentación de partida.** Este Anexo debe incluir aquellos documentos que se han tenido en cuenta para establecer los requisitos de diseño.
* **Cálculos**. Este Anexo o Anexos tienen como misión justificar las fórmulas aplicadas, las soluciones adoptadas y, conjuntamente con los documentos planos y el pliego de condiciones, debe describir de forma unívoca el objeto del Proyecto.

Debe contener las hipótesis de partida, los criterios y procedimientos de cálculo, así como los resultados finales base del dimensionado o comprobación de los distintos elementos que constituyen el objeto del Proyecto.

* Anexos de aplicación en función del ámbito del Proyecto, son por ejemplo:
  + Seguridad (prevención de incendios, sanidad, radiaciones, pública concurrencia, etc.).
  + Medio ambiente (acústica, residuos, emisiones, etc.).
  + Eficiencia energética.
  + Emplazamiento del proyecto, Geotécnicos, Hidráulicos, Hidrológicos, Pluviométricos, etc.
  + Gestión de residuos.
  + Otros.
* **Estudios con entidad propia.** Este documento debe contener todos aquellos estudios que deban incluirse en el Proyecto por exigencias legales. Debe comprender, entre otros y sin carácter limitativo, los relativos a:
  + Estudio Básico de Seguridad y Salud o Estudio de Seguridad y Salud, según corresponda.
  + Estudio de Impacto Ambiental. Se realizará según la tipología prevista en la legislación vigente que aplique.

Cada anexo debe contener la justificación del cumplimiento de la normativa legal vigente aplicable y, si procede, de las fórmulas aplicadas para el cálculo.

* **Otros documentos que justifiquen y aclaren conceptos expresados en el Proyecto.** Se pueden incluir:
  + Catálogos de los elementos constitutivos del objeto del Proyecto.
  + Listados.
  + Información en soportes lógicos, magnéticos, ópticos o cualquier otro.
  + Maquetas o modelos.
  + Otros documentos que se juzguen necesarios.

**Documento Nº 4. Planos.** Se deberá tener en cuenta lo indicado en las normas referencias en el apartado 8.2 de la norma UNE 157001:2014.

**Documento Nº 5. Pliego de Condiciones.** El pliego de condiciones es uno de los documentos que constituyen el Proyecto y tiene como misión establecer las condiciones técnicas, económicas, administrativas, facultativas y legales para que el objeto del Proyecto pueda materializarse en las condiciones especificadas, evitando posibles interpretaciones diferentes de las deseadas.

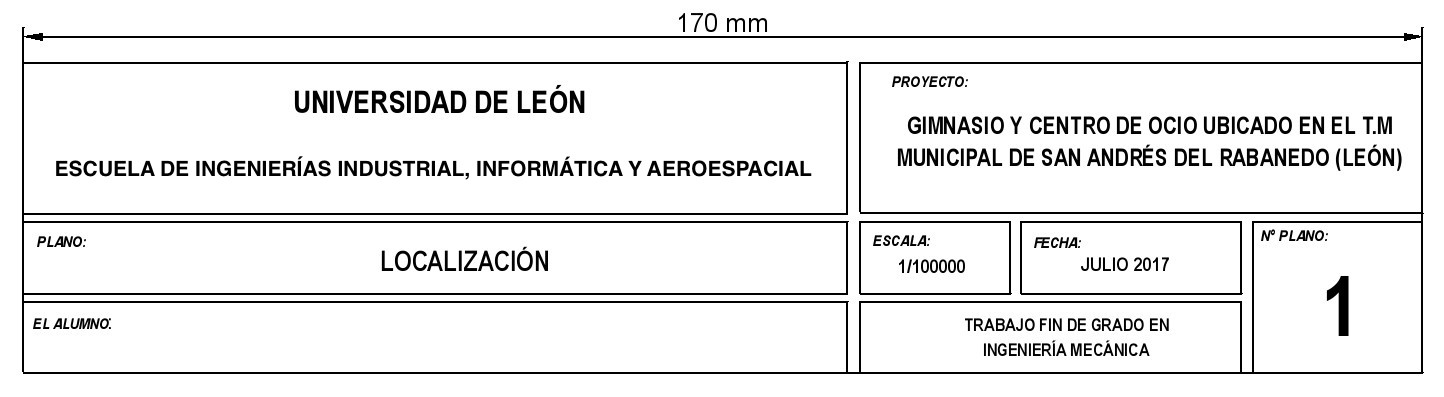
**Documento Nº 6. Mediciones.** Debe contener un listado completo de las partidas de obra que configuran la totalidad del proyecto. Debe servir de base para la realización del Presupuesto.

**Documento Nº 7. Presupuesto.**

MODELO DE CAJETÍN PARA PLANOS

Anchura máxima 170 mm, y altura variable. No habría límites si el plano no se doblase.

Situación en ángulo inferior derecho del área de dibujo del plano, con el mismo grosor que el recuadro que marca la zona efectiva de dibujo.



**ESTUDIOS TÉCNICOS, ORGANIZATIVOS Y ECONÓMICOS, Y TRABAJOS TEÓRICO-EXPERIMENTALES**

Se utilizará la carátula incluida en este anexo. El tamaño de papel será A4 y los márgenes del documento serán 3 cm en la parte superior y a la izquierda y 2.5 cm en la parte inferior y a la derecha.

Se utilizarán encabezados y pies de página. En el pie de página irá el nombre del alumno-autor del trabajo y en el encabezado irá la numeración de las hojas, indicándose el total.

Para la numeración de los capítulos, apartados y anexos se seguirán las recomendaciones de la norma UNE 50132.

Cada capítulo iniciará una página nueva. Los títulos de los capítulos utilizarán una fuente de tamaño 28 puntos en negrita y llevarán la primera letra en mayúscula, mientras que el resto serán minúsculas. El interlineado (si el título ocupa más de una línea) será sencillo. Después del título irá una línea en blanco de tamaño 12 puntos.

Para los títulos de los apartados, se utilizará un tamaño de 14 puntos, letra negrita y mayúscula. Los títulos de los apartados y sub-apartados irán precedidos del número del apartado. Debajo del título de los apartados se dejará una línea en blanco.

Para el cuerpo del texto se utilizará un tamaño de 12 puntos, justificado a izquierda y derecha y con interlineado igual a 1.5 líneas, sin espacio adicional entre párrafos.

Las figuras y tablas se intercalarán en el texto, centradas y con un pie de figura o encabezado de tabla. Las figuras y tablas llevarán una línea antes y otra después, y la forma de nombrarlas en el texto será “figura #.#” o “tabla #.#”, donde el primer número es el del capítulo y el segundo es el número de orden de la figura o de la tabla dentro de ese capítulo. El texto al pie de la figura o en el encabezado de la tabla estará en escrito con un tamaño de 12 puntos, y deberá señalar lo que se representa de una forma clara. Ejemplo:

Figura 4.1. Relación entre … (Fuente: ….)

Tabla 4.1. Tabla con resultados de…(Fuente: ….)

Si el texto requiere la inclusión de ecuaciones, el formato de las mismas es libre. Se numerarán según el formato (#.#) alineado al margen derecho junto a la ecuación y se citarán en el cuerpo del texto como “ecuación #.#”, donde el primer número es el capítulo y el segundo el número de la ecuación. Ejemplo:

… (4.1)

Al final del cuerpo del trabajo debe incluirse una lista de todas las fuentes bibliográficas en las que se basa éste, y debe hacerse referencia a cada uno de los elementos de esta lista en todos los lugares adecuados dentro del texto. A pesar de que la norma UNE 50135 incluye cuestiones relativas al formato de las referencias bibliográficas, se usarán prioritariamente los estilos de referenciación:

* HARVARD (Autor-Fecha)
* IEEE (Numérico)

En todo caso, se podrá usar cualquier otro formato siempre que esté normalizado.

Serán elementos de obligada aparición los siguientes:

**Resumen.** Entre 300 y 500 palabras, en español.

**Abstract.** Entre 300 y 500 palabras, en inglés.

**Índice de contenidos**

**Índice de figuras**

**Índice de cuadros y tablas**

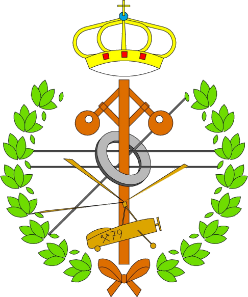
**Glosario de signos, símbolos, unidades, abreviaturas, acrónimos o términos** (si hubiera)

**Cuerpo del trabajo (recomendación):**

* Introducción
* Estado del arte
* Núcleo del trabajo con ilustraciones esenciales y tablas (objetivos, materiales y métodos,…)
* Resultados
* Conclusiones y recomendaciones
* Agradecimientos, si los hubiere

**Lista de referencias bibliográficas**

**Anexos**

****

Escuela de Ingenierías

Industrial, Informática y Aeroespacial

**GRADO EN INGENIERÍA INFORMÁTICA**

Trabajo de Fin de Grado

Aplicación web para la gestión de cultivos hidropónicos

Web application for hydroponic crop management

Autor: José Antonio López Pérez

Tutores: Rubén Rodríguez, Jesús Fernández

(Julio, 2025)

|  |  |
| --- | --- |
| UNIVERSIDAD DE LEÓN  Escuela de Ingenierías Industrial, Informática y Aeroespacial  GRADO EN INGENIERÍA INFORMÁTICA  Trabajo de Fin de Grado | |
| **ALUMNO:** José Antonio López Pérez | |
| **TUTOR:** Rubén Rodríguez, Jesús Fernández | |
| **TÍTULO:** Aplicación web de gestión de cultivos hidropónicos | |
| **TITLE:** Web application for hydroponic crop management | |
| **CONVOCATORIA:** Julio, 2025 | |
| **RESUMEN:**  En un contexto de creciente interés por los sistemas de cultivo sostenible y la optimización de recursos, la hidroponía se presenta como una alternativa eficiente para la producción de alimentos en espacios reducidos y con un menor consumo de agua. Sin embargo, uno de los principales retos de estos sistemas es la necesidad de un monitoreo constante y una gestión precisa de variables como la temperatura, la humedad o los niveles de nutrientes. Automatizar y facilitar este control resulta fundamental para asegurar la viabilidad del cultivo y mejorar su rendimiento.  Este Trabajo de Fin de Grado tiene como objetivo principal el diseño y desarrollo de una aplicación web dedicada a la gestión de módulos hidropónicos inteligentes. La aplicación permite a los usuarios monitorizar en tiempo real los datos capturados por distintos sensores instalados en el sistema, además de programar acciones automáticas basadas en condiciones específicas o preferencias del usuario. De este modo, es posible optimizar el mantenimiento del cultivo mediante la automatización de tareas como el riego, la ventilación o el ajuste de la iluminación. La plataforma está organizada en diferentes secciones que ofrecen funcionalidades como la visualización gráfica de los datos registrados, el estado actual del sistema, recomendaciones personalizadas para mejorar el cultivo y paneles de configuración para adaptar el funcionamiento del módulo a las necesidades concretas del usuario.  Desde el punto de vista técnico, el sistema sigue una arquitectura basada en microservicios virtualizados mediante contenedores Docker, lo que garantiza su escalabilidad, portabilidad y facilidad de despliegue. Además, el proyecto incluye el diseño y la construcción de un módulo hidropónico físico, equipado con una serie de sensores y componentes electrónicos gestionados a través de una estructura fabricada mediante impresión 3D. Gracias a esta combinación de hardware y software, se ofrece una solución integral para el control inteligente de cultivos hidropónicos. | |
| **ABSTRACT:**  In a context of growing interest in sustainable farming systems and resource optimization, hydroponics emerges as an efficient alternative for food production in reduced spaces and with lower water consumption. However, one of the main challenges of these systems is the need for constant monitoring and precise management of variables such as temperature, humidity, and nutrient levels. Automating and facilitating this control isessential to ensure the viability of the crop and improve its performance.  This Final Degree Project aims to design and develop a web application dedicated to the management of intelligent hydroponic modules. The application allows users to monitor in real time the data captured by different sensors installed in the system, as well as to schedule automatic actions based on specific conditions or user preferences. In this way, it is possible to optimize crop maintenance through the automation of tasks such as irrigation, ventilation, or lighting adjustment. The platform is organized into different sections that offer functionalities such as graphical visualization of recorded data, the status of the system, personalized recommendations to improve the crop, and configuration panels to adapt the module's operation to the user's specific needs.  From a technical point of view, the system follows an architecture based on microservices virtualized through Docker containers, which ensures scalability, portability, and ease of deployment. Additionally, the project includes the design and construction of a physical hydroponic module, equipped with a series of sensors and electronic components managed through a structure manufactured by 3D printing. Thanks to this combination of hardware and software, a comprehensive solution is offered for the intelligent control of hydroponic crops. | |
| **Palabras clave:** Hidroponía, Automatización, Monitoreo, Sensores, Aplicación web, Microservicios, Docker, Impresión 3D, Optimización de recursos, Control inteligente | |
| **Firma del alumno:** | **VºBº Tutor/es:** |

Tabla de contenido

[Índice de figuras 9](#_Toc192241343)

[Índice de tablas 10](#_Toc192241344)

[Glosario de términos 10](#_Toc192241345)

[Introducción 11](#_Toc192241346)

[Estudio del problema 14](#_Toc192241347)

[El contexto del problema 14](#_Toc192241348)

[El estado de la cuestión 15](#_Toc192241349)

[La definición del problema 15](#_Toc192241350)

[Gestión de proyecto software 16](#_Toc192241351)

[2.1 Alcance del proyecto 16](#_Toc192241352)

[2.1.1 Definición del proyecto 16](#_Toc192241353)

[2.1.2 estimación de tareas y recursos 17](#_Toc192241354)

[2.1.3 Presupuesto 18](#_Toc192241355)

[2.2 Plan de trabajo 19](#_Toc192241356)

[2.2.1 Identificación de tareas 19](#_Toc192241357)

[2.2.2. Asignación de tareas y estimación 20](#_Toc192241358)

[2.2.3. Planificación de tareas 2.3. Gestión de recursos 20](#_Toc192241359)

[2.3.1. Especificación de recursos 20](#_Toc192241360)

[2.3.2. Asignación de recursos 20](#_Toc192241361)

[2.4. Gestión de riesgos 20](#_Toc192241362)

[2.4.1. Identificación de riesgos 20](#_Toc192241363)

[Solución 21](#_Toc192241364)

[3.1. Descripción de la solución 3.2. El proceso de desarrollo 21](#_Toc192241365)

[3.2.1. Análisis 3.2.2. Diseño 21](#_Toc192241366)

[3.3. El producto del desarrollo 21](#_Toc192241367)

[Evaluación 21](#_Toc192241368)

[4.1. Proceso de evaluación 21](#_Toc192241369)

[4.1.1. Forma de evaluación 21](#_Toc192241370)

[4.1.2. Casos de prueba 4.2. Análisis de resultados Conclusión 21](#_Toc192241371)

[Lista de referencias 21](#_Toc192241372)

[Anexo 1 - Seguimiento de Proyecto 21](#_Toc192241373)

[Anexo 2 - Manual de usuario 21](#_Toc192241374)

# Índice de figuras

[Ilustración 1 8](#_Toc190859315)

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.

Ilustración 1

# Índice de tablas

# Glosario de términos

**React**: Framework utilizado para la creación de aplicaciones web.

**API**: Conjunto de definiciones y protocolos que permiten que dos aplicaciones se comuniquen entre sí.

**Backend**: Parte de una aplicación que gestiona la lógica, la base de datos y el procesamiento de datos en el servidor.

**MariaDB**: Sistema de gestión de bases de datos relacional, derivado de MySQL, utilizado para almacenar y gestionar datos.

**DaisyUI**: Plugin de Tailwind CSS que proporciona componentes predefinidos y estilos personalizables para interfaces web.

**Docker**: Plataforma que permite crear, distribuir y ejecutar aplicaciones dentro de contenedores ligeros y portables.

**CSS**: Lenguaje de hojas de estilo usado para describir la presentación y diseño de documentos HTML.

**Arduino**: Plataforma de hardware libre basada en una placa con microcontrolador y un entorno de desarrollo para crear proyectos electrónicos.

**Software**: Conjunto de programas, instrucciones y reglas informáticas que permiten ejecutar tareas en un dispositivo.

**Hardware**: Conjunto de componentes físicos y tangibles de un sistema informático o electrónico

# Introducción

Este proyecto plantea una solución enfocada en el desarrollo de un producto software completo destinado a la gestión y automatización de módulos hidropónicos inteligentes. El objetivo principal es permitir a los usuarios visualizar la información recogida por los sensores instalados en el sistema y facilitar la programación de acciones automáticas que optimicen el mantenimiento del cultivo. A través del procesamiento y presentación adecuada de los datos capturados (como temperatura, humedad o electro conductividad del agua), se busca garantizar un control preciso que permita mejorar el rendimiento y la viabilidad de los cultivos hidropónicos.

Planteamiento del problema

Con el desarrollo del proyecto, se pretende analizar y aplicar metodologías ágiles durante la gestión del proceso, empleando un enfoque iterativo e incremental. Gracias a este modelo, el producto podrá ir incorporando nuevas funcionalidades de manera progresiva a lo largo de diferentes iteraciones, permitiendo que sea operativo desde las primeras fases y favoreciendo la obtención de feedback por parte de los usuarios. Esto facilita la detección temprana de errores y reduce los costes asociados a posibles cambios, además de asegurar una evolución continua del sistema a medida que se desarrollan nuevas versiones.

En cuanto a la arquitectura, se ha optado por un diseño basado en microservicios virtualizados mediante contenedores Docker, lo que permite dividir el sistema en diferentes componentes independientes, cada uno encargado de una funcionalidad específica, garantizando flexibilidad, escalabilidad y facilidad de mantenimiento. La comunicación entre estos microservicios es fundamental y se realiza a través de peticiones bien definidas que permiten integrar de manera eficiente todas las partes del sistema.

El producto final integra tanto el desarrollo de la aplicación web, que proporciona al usuario acceso a los datos y configuración del módulo hidropónico, como el diseño físico del propio módulo, fabricado mediante impresión 3D y equipado con sensores y componentes electrónicos. Gracias a esta combinación de hardware y software, se ofrece una solución completa que facilita el control inteligente y automatizado de cultivos hidropónicos, proporcionando una alternativa eficiente y sostenible para la producción de alimentos.

Objetivos

El objetivo principal del proyecto es desarrollar una aplicación web que permita la gestión y automatización de módulos hidropónicos inteligentes, facilitando a los usuarios la monitorización de datos en tiempo real y la programación de acciones automáticas para optimizar el mantenimiento del cultivo.

Además del objetivo principal, hay una serie de objetivos específicos:

* Comprender qué es un módulo hidropónico inteligente y cuáles son sus principales componentes.
* Comprender la arquitectura basada en microservicios y su aplicación en sistemas de gestión de datos IoT.
* Conocer los procedimientos y métodos para desarrollar una arquitectura basada en microservicios mediante Docker.
* Comprender el funcionamiento y la integración de sensores en sistemas de monitoreo ambiental (temperatura, humedad, nutrientes).
* Conocer los medios de comunicación entre los microservicios del sistema y cómo garantizar su correcta interacción.
* Evaluar la experiencia del usuario en la aplicación web, analizando la usabilidad y accesibilidad de las funcionalidades ofrecidas.
* Integrar hardware y software de forma eficiente para ofrecer una solución completa de gestión de cultivos hidropónicos.
* Conocer las ventajas que aporta el uso de un bus de datos a la hora de evitar pérdidas de información.
* Desarrollar un módulo hidropónico diseñado específicamente para la aplicación.
* Analizar los datos obtenidos y realizar recomendaciones para el usuario.

Metodología

Para el desarrollo de este proyecto se ha seguido una metodología ágil e incremental, gestionando el ciclo de vida del software a través de la plataforma GitHub como herramienta principal para el control de versiones y seguimiento del progreso. El trabajo se ha dividido en dos grandes etapas de aproximadamente tres meses cada una.

Durante la primera etapa, el objetivo principal fue obtener una primera versión del prototipo funcional, enfocada en desarrollar un Producto Mínimo Viable (MVP) que incluyera las características básicas necesarias para validar el funcionamiento del sistema. En esta fase inicial se priorizó la integración de los componentes esenciales del sistema, como la captura de datos desde sensores, la visualización básica de información en la aplicación web y la ejecución de acciones automáticas simples.

La planificación del desarrollo se organizó en iteraciones de dos semanas, al final de las cuales se incorporaba al proyecto al menos una nueva funcionalidad que permitiera avanzar de forma continua y medible. Además, esta estructura permitió realizar pruebas parciales y corregir posibles errores antes de continuar con nuevas implementaciones.

Una vez validado el prototipo inicial, se llevó a cabo una segunda etapa, también de tres meses, centrada en el perfeccionamiento del sistema y el desarrollo de la versión final. En esta fase se añadieron mejoras tanto a nivel funcional como visual, incluyendo la optimización de la interfaz de usuario, la ampliación de las opciones de configuración y la integración completa de todas las funcionalidades planificadas para el producto final.

El uso de esta metodología ha permitido mantener un flujo de trabajo flexible y adaptativo, facilitando la incorporación de mejoras continuas y la resolución temprana de incidencias, lo que ha contribuido a obtener un sistema estable y operativo al finalizar cada ciclo de desarrollo.

Estructura del trabajo

Este trabajo se organiza en varios apartados que permiten guiar al lector a través de todas las fases del desarrollo del proyecto, desde su concepción hasta su evaluación final:

**Capítulo 1: Análisis del problema**

En esta sección se contextualiza la problemática que da origen al proyecto, detallando las necesidades detectadas y los objetivos que se persiguen. También se analizan las alternativas existentes y se justifican las tecnologías seleccionadas para abordar la solución.

**Capítulo 2: Planificación y gestión del proyecto**

Aquí se describe cómo se ha organizado el desarrollo del sistema, incluyendo la definición del alcance del trabajo, la distribución y calendarización de tareas, así como los recursos empleados a lo largo del proceso. Se detallan las estrategias seguidas para garantizar una gestión eficiente y controlada del proyecto.

**Capítulo 3: Desarrollo de la solución**

En este apartado se profundiza en la solución implementada, explicando de manera detallada el diseño del sistema, sus principales componentes y la arquitectura adoptada. Se justifica cada decisión técnica tomada y se describe cómo estas decisiones permiten cumplir con los objetivos planteados.

**Capítulo 4: Validación y resultados**

Esta sección recoge el proceso llevado a cabo para verificar el correcto funcionamiento del sistema, así como las pruebas realizadas. Además, se presentan los resultados obtenidos y se analiza hasta qué punto se han alcanzado los objetivos iniciales.

**Capítulo 5: Conclusiones y futuras mejoras**

Finalmente, se realiza una reflexión global sobre el trabajo desarrollado. Se exponen los principales retos y dificultades encontrados durante el proyecto, se valoran los resultados alcanzados y se proponen posibles líneas de mejora y evolución futura del sistema.

**Anexo 1: Registro del desarrollo**

Este anexo recoge de forma cronológica la evolución del proyecto, mostrando cómo ha progresado a lo largo del tiempo y los hitos más relevantes alcanzados.

**Anexo 2: Guía de usuario**

Incluye instrucciones claras y detalladas para facilitar al usuario la utilización de la aplicación, desde su puesta en marcha hasta el manejo de sus principales funcionalidades.

Capítulo 1

# Estudio del problema

En esta sección se realiza el estudio del contexto del problema, su estado en la actualidad y se define sus aspectos clave.

## **1.1 El contexto del problema**

El Trabajo de Fin de Grado tiene como objetivo el desarrollo de GreenhouseIOT, una aplicación web orientada a la gestión y supervisión de datos generados por un módulo hidropónico. La idea principal surge como una oportunidad para consolidar conocimientos en tecnologías actuales del ámbito del desarrollo web, además de obtener experiencia práctica en la creación de soluciones reales con proyección profesional.

El proyecto propone una herramienta capaz de monitorizar y almacenar los datos obtenidos a partir de sensores instalados en sistemas hidropónicos, permitiendo así un control preciso de variables críticas como temperatura, humedad o nivel de nutrientes. Todo ello está pensado para facilitar el seguimiento de estos cultivos en tiempo real desde una plataforma web intuitiva y accesible.

Gracias al uso de tecnologías modernas como React para el frontend, FastAPI para la lógica del backend, MariaDB como sistema de almacenamiento, junto con Kafka para el envío de datos y Docker para la contenerización de servicios, la aplicación busca ofrecer una solución completa para el seguimiento local de instalaciones hidropónicas.

## **1.2 El estado de la cuestión**

El auge de la agricultura inteligente y, en particular, de los sistemas hidropónicos, ha impulsado el desarrollo de soluciones tecnológicas orientadas al monitoreo y gestión de cultivos mediante sensores IoT. A nivel industrial, existen plataformas robustas que ofrecen servicios avanzados de análisis de datos, control remoto y optimización de recursos, aunque muchas de estas soluciones están orientadas a explotaciones a gran escala y suponen altos costes de implementación.

En cambio, GreenhouseIOT surge como una alternativa pensada para instalaciones locales o de pequeña escala, proporcionando una plataforma sencilla, adaptable y sin grandes requerimientos técnicos para el usuario final. Mientras que otras herramientas del mercado priorizan la integración de sistemas complejos y centralizados, este proyecto apuesta por una arquitectura distribuida mediante microservicios, facilitando así su mantenimiento, escalabilidad y despliegue en diferentes entornos.

## **1.3 La definición del problema**

En base al análisis previo, el problema principal que aborda este proyecto es la falta de soluciones accesibles y adaptadas a pequeñas instalaciones hidropónicas que permitan gestionar y visualizar de manera eficiente los datos recogidos por sensores IoT.

Con GreenhouseIOT se pretende cubrir las siguientes necesidades:

* Monitorización continua de parámetros ambientales relevantes (temperatura, humedad, nivel de agua, etc.).
* Almacenamiento ordenado y seguro de los datos generados por los sensores.
* Visualización clara e intuitiva de los datos históricos y en tiempo real desde una interfaz web.
* Comunicación eficiente entre los diferentes módulos del sistema.
* Facilitar la escalabilidad del sistema mediante una arquitectura basada en contenedores (Docker) y microservicios.
* Además, uno de los objetivos clave de la aplicación es ofrecer una experiencia de usuario sencilla, que permita tanto la instalación como la gestión del sistema sin necesidad de conocimientos avanzados en tecnología o programación. A partir de una plataforma web accesible desde cualquier dispositivo, el usuario puede supervisar su módulo hidropónico, realizar consultas de los registros históricos y recibir recomendaciones basadas en los datos almacenados.

Con todo ello, GreenhouseIOT pretende ser una herramienta útil y práctica para impulsar el uso de la agricultura inteligente en proyectos locales y domésticos.

Capítulo 2

# Gestión de proyecto software

## 2.1 Alcance del proyecto

En este apartado se describen las tareas necesarias para llevar a cabo el desarrollo de GreenhouseIOT, así como los recursos requeridos, la estimación de costes y los posibles riesgos asociados. La finalidad del proyecto es crear una aplicación web que permita monitorizar y gestionar los datos de sensores instalados en un módulo hidropónico, facilitando su visualización, análisis y almacenamiento de manera sencilla y eficiente para el usuario final.

La aplicación busca ofrecer una herramienta de apoyo a pequeñas instalaciones hidropónicas, proporcionando información en tiempo real y registros históricos que permitan optimizar el cultivo y anticipar posibles problemas en el entorno controlado.

### 2.1.1 Definición del proyecto

El proyecto consiste en el desarrollo de una plataforma web que permita a los usuarios supervisar y gestionar los datos de su módulo hidropónico. La aplicación incluirá funciones como:

* Visualización en tiempo real de datos ambientales (temperatura, humedad, nivel de nutrientes, etc.).
* Consulta de registros históricos y gráficas comparativas.
* Alertas ante valores críticos o anómalos detectados.
* Gestión del almacenamiento de datos mediante una base de datos relacional.
* Panel de administración básico para supervisar el sistema y verificar el estado de los distintos módulos.

La arquitectura estará dividida en 3 capas: origen de datos, persistencia y explotación, desarrolladas de manera paralela y coordinada, asegurando una integración continua y modular para facilitar su mantenimiento y escalabilidad.

La tecnología elegida para la capa de explotación es React, que permitirá diseñar una interfaz interactiva y dinámica para el usuario, en la capa de persistencia encontramos la base de datos y un contenedor desarrollado con Fastapi que permite interactuar con ella. Por último, en la capa de origen de datos encontramos un servicio de Python que se encarga de preguntar a los módulos hidropónicos los datos de forma periódica y prepara esos datos para ser transportados por Kafka hacia la capa de persistencia.

Además, todo el entorno se desplegará utilizando contenedores Docker, asegurando así la portabilidad y estabilidad de la aplicación, independientemente del entorno de ejecución.

El sistema está diseñado para su funcionamiento a nivel local, pensado para pequeñas instalaciones hidropónicas que requieran una solución asequible, funcional y sencilla de implementar.

### 2.1.2 estimación de tareas y recursos

**Tareas**

El desarrollo del proyecto se organizará en tareas semanales, dentro de una metodología incremental que permitirá ir incorporando nuevas funcionalidades progresivamente. A lo largo del desarrollo se realizarán reuniones periódicas para revisar los avances, valorar posibles mejoras y ajustar las prioridades.

Entre las tareas principales previstas se encuentran:

* Diseño de la arquitectura de la aplicación y configuración inicial del entorno.
* Desarrollo del backend con FastAPI.
* Desarrollo del frontend con React.
* Integración de la base de datos MariaDB.
* Configuración y manejo de envió de datos mediante Kafka.
* Creación de pruebas funcionales para asegurar el correcto comportamiento de cada módulo.
* Despliegue de los servicios con Docker.
* Elaboración de documentación técnica y manual de usuario.
* Pruebas finales y validación del sistema.

Cada entrega de una nueva funcionalidad tendrá una duración estimada mínima de dos semanas, dependiendo de la complejidad y los recursos disponibles.

**Recursos**

Para el desarrollo de GreenhouseIOT se contemplan tanto recursos humanos como físicos:

**Humanos**

El equipo ideal del proyecto estaría compuesto por cuatro perfiles clave, cada uno enfocado en áreas específicas del desarrollo:

Jefe de proyecto: encargado de planificar, supervisar y coordinar todas las fases del desarrollo.

Backend developer: encargado de la implementación de la lógica de negocio, API y conexiones con la base de datos. Se encargaría del mantenimiento y configuración de la base de datos y de las tareas relacionadas con Docker. Desarrollaría prácticas de ci/cd para asegurar el correcto funcionamiento y entrega de las actualizaciones del sistema.

Frontend developer: responsable del desarrollo de la parte visual e interactiva de la plataforma. Se encargaría de conseguir una interfaz amigable para el usuario haciendo uso de librerías de componentes UI.

Encargado hardware: responsable del diseño de circuitos electrónicos, diseño de los componentes 3D, procesos de impresión 3D, montaje de módulos hidropónicos, desarrollo del código de microcontroladores esp32 y configuraciones derivadas.

**Físicos**

Los recursos físicos necesarios para llevar a cabo el proyecto son:

* Hardware: equipos informáticos para cada miembro del equipo, capaces de ejecutar entornos de desarrollo y contenedores de Docker de manera eficiente. El coste de la impresión 3D y materiales asociados a componentes del módulo.
* Software: no se prevén costes adicionales, ya que todas las tecnologías utilizadas son open source (React, FastAPI, MariaDB, Kafka y Docker).

### 2.1.3 Presupuesto

El presupuesto para el desarrollo de GreenhouseIoT se ha estimado considerando los costes asociados tanto a los recursos humanos como a los recursos físicos e infraestructurales necesarios para garantizar el correcto desarrollo del proyecto.

**Recursos Humanos**

El equipo de trabajo estará formado por los siguientes perfiles profesionales, junto con su coste mensual aproximado y la duración estimada del proyecto (3 meses):

|  |  |  |  |
| --- | --- | --- | --- |
| Puesto | Coste por hora(€/h) | Horas de trabajo | Coste total (€) |
| Jefe de proyecto | 30 | 216 | 6.480 |
| Backend developer | 25 | 352 | 8.800 |
| Frontend developer | 25 | 248 | 6.200 |
| Encargado hardware | 25 | 280 | 7.000 |

Total, RRHH 28.480€

Nota: Se ha estimado un equipo reducido enfocado en cubrir las áreas críticas del proyecto, prescindiendo de perfiles secundarios para optimizar los costes. Otra opción más económica sería contratar a un desarrollador fullstack y alargar ligeramente la duración del desarrollo.

**Recursos Físicos**

Hardware:

Aquí se observa el listado de materiales necesarios para la realización de 1 módulo.

|  |  |  |
| --- | --- | --- |
| Nombre del recurso | Trabajo | Costo |
| Impresora 3D | 1 und | 319,00 € |
| Plástico PLA | 3 bobina 850g | 63,00 € |
| Portatiles | 4 und | 1.716,00 € |
| Cableado | 2 metros | 22,98 € |
| DHT22 | 1 und | 4,63 € |
| Sensor temperatura del agua | 1 und | 1,39 € |
| Sensor TDS | 1 und | 3,49 € |
| Fuente de alimentación | 1 und | 7,24 € |
| Regulador de voltaje | 1 und | 2,97 € |
| Foco led | 1 und | 5,79 € |
| Tornillería | 1 caja | 4,79 € |
| LDR | 1 und | 1,42 € |
| Resistencias | 1 caja | 2,92 € |
| Modulo 4x Relé | 1 und | 2,42 € |
| Conexiones | 1 caja | 9,80 € |
| ESP32 S3 | 1 und | 6,03 € |
| ESP32 Cam | 1 und | 7,49 € |
| Contenedores | 1 und | 4,50 € |
| Oxigenador | 1 und | 2,49 € |
| Ventilador | 1 und | 1,54 € |
| Pegamento | 1 caja | 3,50 € |
| Tuercas inyección | 1 caja | 3,99 € |
| pcb | 1 caja | 2,84 € |
| lente 77mm | 1 und | 6,36 € |
| Nivel de agua | 1 und | 2,79 € |
| Soldador | 1 kit | 21,00 € |
|  | **Coste Total** | 2.230.37€ |

En el presupuesto anterior vienen incluidos costes materiales asociados a la compra de equipos de montaje o producción. A continuación, se muestra el presupuesto mínimo que requiere la construcción de un segundo módulo, no se incluyen costes como la compra de la impresora 3D o de nueva tornillería ya que el gasto viene por caja y no siempre es necesario comprar más unidades.

|  |  |  |
| --- | --- | --- |
| Nombre del recurso | Trabajo | Costo |
| Plástico PLA | 3 bobina 850g | 63,00 € |
| Cableado | 2 metros | 22,98 € |
| DHT22 | 1 und | 4,63 € |
| Sensor temperatura del agua | 1 und | 1,39 € |
| Sensor TDS | 1 und | 3,49 € |
| Fuente de alimentación | 1 und | 7,24 € |
| Regulador de voltaje | 1 und | 2,97 € |
| Foco led | 1 und | 5,79 € |
| LDR | 1 und | 1,42 € |
| Modulo 4x Relé | 1 und | 2,42 € |
| ESP32 S3 | 1 und | 6,03 € |
| ESP32 Cam | 1 und | 7,49 € |
| Contenedores | 1 und | 4,50 € |
| Oxigenador | 1 und | 2,49 € |
| Ventilador | 1 und | 1,54 € |
| lente 77mm | 1 und | 6,36 € |
| Nivel de agua | 1 und | 2,79 € |
|  | **Coste total** | 146,53 € |

Software: No se contemplan costes adicionales debido al uso exclusivo de herramientas y tecnologías open source como FastAPI, React, MariaDB, Kafka y Docker.

**Costes Indirectos**

Oficina: Se optará por el trabajo remoto por lo que no se aplican costes debidos a oficinas físicas.

Para el desarrollo y testeo de la aplicación será necesaria la construcción del por lo menos 1 módulo hidropónico inteligente.

En este caso se considerará que se construye solo el primer módulo ya que tras el desarrollo el cliente será responsable y propietario de la compra y mantenimiento del módulo físico.

**Presupuesto Total**

|  |  |
| --- | --- |
| Concepto | Coste (€) |
| Recursos Humanos | 28.480 € |
| Hardware | 2.230.37€ |
| **Total** | **30.710,37€** |

Este presupuesto está sujeto a posibles ajustes según la evolución del proyecto, cambios en los plazos o incorporación de nuevos requerimientos.

**Coste total**

|  |  |
| --- | --- |
| Concepto | Coste (€) |
| Recursos Humanos  Hardware | 28.480 €  2.230 € |
| Beneficio industrial (15 %) | 4.607 € |
| Impuestos (I.V.A. 21 %) | 7.417 € |
| **Total del proyecto** | **42.733 €** |

## 2.2 Plan de trabajo

En esta sección se detalla cómo se han establecido las diferentes tareas del desarrollo de la aplicación GreenhouseIoT, así como el tiempo aproximado de dedicación a cada una. El plan de trabajo sigue una metodología incremental, permitiendo validar cada funcionalidad antes de avanzar a la siguiente, asegurando un desarrollo controlado y eficiente.

### 2.2.1 Identificación de tareas

**Tareas de investigación del sector**

|  |
| --- |
| Nombre de tarea |
| * 1. Investigación sobre el sector |
| * 1. Investigación sobre las tecnologías |
| * 1. Planificación de objetivos esperados |
| * 1. Estudio de viabilidad del proyecto |
| * 1. Creación de la identidad del producto |
| * 1. Diseño del primer boceto relativo a la interfaz de la aplicación y estructura |
| * 1. Planificación de metodología de desarrollo. |

1. **Investigación sobre el sector**

Se llevó a cabo un estudio exploratorio para entender el contexto actual de la agricultura inteligente y el uso de módulos hidropónicos en pequeñas instalaciones. Este análisis permitió definir con precisión el alcance del sistema GreenhouseIOT.

1. **Investigación sobre las tecnologías**

Se investigaron herramientas modernas como React, FastAPI, Docker o Kafka, seleccionando las más adecuadas para la arquitectura distribuida del sistema. Esta fase resultó fundamental para garantizar una integración tecnológica eficiente.

1. **Planificación de objetivos esperados**

Se definieron los objetivos principales y secundarios del proyecto, delimitando funcionalidades clave a implementar. Esta etapa permitió estructurar las tareas de desarrollo con metas alcanzables en cada iteración.

1. **Estudio de viabilidad del proyecto**

Se evaluaron aspectos técnicos, económicos y de tiempo para confirmar que el proyecto era ejecutable en el plazo y con los recursos disponibles. Se ajustaron prioridades para optimizar el proceso de desarrollo.

1. **Creación de la identidad del producto**

Se estableció la imagen de la plataforma, incluyendo nombre, paleta de colores y estilo visual. Esta identidad facilitará su futura expansión y diferenciación frente a soluciones similares.

1. **Diseño del primer boceto relativo a la interfaz de la aplicación y estructura**

Se elaboraron prototipos iniciales de la interfaz de usuario, priorizando la claridad, accesibilidad y facilidad de uso. Este diseño sirvió de base para el posterior desarrollo del frontend.

1. **Planificación de metodología de desarrollo.**

Se definió una estrategia basada en metodologías ágiles, con entregas iterativas, control de versiones mediante GitHub y pruebas continuas. Esta elección permitió adaptar el desarrollo a los cambios surgidos durante el proyecto.

**Tareas de desarrollo**

|  |
| --- |
| Nombre de tarea |
| **GreenhouseIOT** |
| **Planificación inicial** |
| * 1. Definición de la idea y requisitos |
| * 1. Diseño de la arquitectura base |
| * 1. Configuraciones del repositorio de código |
| * 1. Creación de la arquitectura del proyecto |
| **Creación del módulo** |
| * 1. Diseño del circuito eléctrico |
| * 1. Montaje del circuito |
| * 1. Diseño de la carcasa 3D |
| * 1. Impresión de la carcasa |
| * 1. Montaje del módulo |
| * 1. Programación del módulo |
| **Origen de datos** |
| * 1. Creación de la api de origen de datos |
| * 1. Programación del productor kafka |
| **Persistencia de datos** |
| * 1. Diseño de la base de datos |
| * 1. Creación de la base de datos |
| * 1. Creación de api para la base de datos |
| * 1. Programación del consumidor Kafka |
| **Explotación de datos** |
| * 1. Boceto diseño gráfico de la aplicación web |
| * 1. Desarrollo estructura básica de la web |
| * 1. Visualización de datos de ultima lectura |
| * 1. Visualización de datos haciendo uso de graficas |
| * 1. Paneles de configuración para automatizaciones |
| * 1. Lógica de gestión de múltiples módulos |
| **Sistema visión artificial** |
| * 1. Pagina dedicada para imágenes de los módulos |
| * 1. Recolección de dataset de imágenes |
| * 1. Entrenamiento de sistema de visión artificial |
| * 1. Creación de recomendaciones personalizadas para el usuario |
| **Control de calidad** |
| * 1. Creación test automáticos para Apis |
| * 1. Creación test automáticos para la web |

**Planificación inicial**

1. **Definición de la idea y requisitos**

Se establecieron los objetivos funcionales y técnicos de la plataforma, así como los requisitos mínimos que debía cumplir cada uno de los módulos. Esta fase permitió fijar las expectativas del proyecto y orientar el diseño del sistema hacia la resolución de necesidades reales en el control de cultivos hidropónicos.

1. **Diseño de la arquitectura base**

Se optó por una arquitectura de microservicios desplegada con Docker, separando las capas de origen de datos, persistencia y explotación. Este enfoque garantiza la escalabilidad y permite mantener cada componente de forma independiente, favoreciendo el mantenimiento y evolución del sistema.

1. **Configuraciones del repositorio de código**

Se configuró un repositorio Git para almacenar y gestionar el código fuente, estableciendo ramas de desarrollo, revisión y producción. También se añadieron flujos básicos de integración continua para automatizar pruebas y asegurar la calidad del software a lo largo del proyecto.

1. **Creación de la arquitectura del proyecto**

Se definieron las rutas internas de comunicación entre los servicios y se organizaron las carpetas y módulos del código según buenas prácticas. Esto facilitó el trabajo colaborativo y la modularización del sistema, permitiendo realizar cambios sin afectar otras partes del código.

**Creación del módulo**

1. **Diseño del circuito eléctrico**

Se diseñó el esquema electrónico del módulo hidropónico, determinando los sensores y actuadores necesarios para medir variables clave como temperatura, humedad, TDS y nivel de agua. Este diseño sirvió de referencia para el montaje físico y programación posterior del sistema.

1. **Montaje del circuito**

Tras definir el esquema eléctrico, se procedió a la integración física de los componentes en una placa base. Se soldaron y conectaron sensores, relés y microcontroladores, verificando la correcta comunicación entre todos los elementos del sistema.

1. **Diseño de la carcasa 3D**

Se creó un diseño 3D personalizado que alojara todos los componentes del sistema de forma compacta y funcional. Se emplearon herramientas de modelado para asegurar el acoplamiento entre piezas, facilitando tanto el montaje como el mantenimiento futuro del hardware.

1. **Impresión de la carcasa**

Utilizando una impresora 3D y filamento PLA, se imprimieron las distintas piezas del módulo hidropónico. Durante este proceso se ajustaron parámetros de impresión y tolerancias para garantizar la estabilidad estructural y la correcta colocación de los sensores y cableado.

1. **Montaje del módulo**

Una vez disponibles todas las piezas, se ensambló el módulo completo, incluyendo el encaje de sensores, conexión del circuito y pruebas preliminares de funcionamiento. Esta etapa permitió validar la parte física del sistema antes de comenzar con su programación.

1. **Programación del módulo**

Se desarrolló el código que permite al ESP32 recoger datos de los sensores, formatearlos y enviarlos al sistema mediante Kafka. También se configuraron rutinas automáticas que permiten ejecutar acciones como activar ventilación o iluminación en función de valores críticos.

**Origen de datos**

1. **Creación de la api de origen de datos**

Se desarrolló una API ligera en FastAPI que actúa como puente entre el módulo físico y el sistema. Esta API permite recibir solicitudes periódicas desde el microcontrolador y estructurar los datos antes de su envío, garantizando que la información fluya correctamente hacia las capas superiores del sistema.

1. **Programación del productor Kafka**

Se implementó un productor Kafka que toma los datos provenientes de la API y los envía a un topic específico en el sistema de mensajería. Este productor se encarga de asegurar que los datos lleguen de forma rápida y confiable al consumidor, permitiendo desacoplar la adquisición de datos de su posterior almacenamiento.

**Persistencia de datos**

1. **Diseño de la base de datos**

Se diseñó un esquema relacional en MariaDB para almacenar de forma estructurada todos los datos generados por los sensores. El diseño contempla la escalabilidad del sistema y permite la consulta eficiente de los registros históricos, incluyendo información por módulo y por fecha.

1. **Creación de la base de datos**

Se llevó a cabo la implementación práctica del modelo de datos definido, creando tablas, relaciones y restricciones necesarias. Además, se realizaron pruebas de integridad y rendimiento para asegurar un almacenamiento fiable y sin pérdidas de información.

1. **Creación de api para la base de datos**

Se desarrolló una API en FastAPI que actúa como intermediaria entre el frontend y la base de datos. Esta API permite acceder a los datos, aplicar filtros por fecha o tipo de módulo y realizar operaciones CRUD, todo ello respetando las reglas de seguridad y acceso definidas.

1. **Programación del consumidor Kafka**

El consumidor Kafka recibe los mensajes generados por el productor y los interpreta para insertarlos automáticamente en la base de datos. Se implementó un sistema de validación para asegurar que los datos sean coherentes y se almacenen correctamente sin duplicados ni errores.

**Explotación de datos**

1. **Boceto diseño gráfico de la aplicación web**

Se creó un prototipo visual de la interfaz gráfica utilizando principios de usabilidad y diseño centrado en el usuario. Este boceto incluyó la disposición de gráficos, menús y paneles interactivos, sirviendo como guía para el desarrollo del frontend en React.

1. **Desarrollo estructura básica de la web**

Se implementaron las rutas principales, componentes base y lógica de navegación de la plataforma. Esta estructura permite que el usuario acceda fácilmente a las diferentes funcionalidades, como la visualización de datos, configuración de módulos y consultas históricas.

1. **Visualización de datos de ultima lectura**

Se implementó un panel principal que muestra en tiempo real los valores más recientes de cada sensor del módulo hidropónico. Esta vista proporciona una lectura rápida del estado actual del cultivo, permitiendo al usuario actuar de inmediato si se detectan valores críticos.

1. **Visualización de datos haciendo uso de graficas**

Se integraron gráficas dinámicas que permiten al usuario visualizar la evolución de los datos registrados a lo largo del tiempo. Esta funcionalidad facilita el análisis del rendimiento del cultivo y ayuda a identificar patrones o anomalías en el entorno.

1. **Paneles de configuración para automatizaciones**

Se desarrollaron herramientas para que el usuario pueda definir reglas personalizadas de automatización, como activar el riego o la ventilación si se superan ciertos umbrales. Estos paneles convierten la plataforma en una solución inteligente, adaptable a diferentes tipos de cultivo.

1. **Lógica de gestión de múltiples módulos**

La aplicación fue diseñada para soportar varios módulos hidropónicos simultáneamente. Se creó una lógica de gestión que permite registrar, supervisar y configurar individualmente cada módulo desde la misma interfaz web, facilitando la expansión del sistema en el futuro.

**Sistema visión artificial**

1. **Pagina dedicada para imágenes de los módulos**

Se incorporó una sección en la aplicación donde se muestran imágenes capturadas por las cámaras instaladas en los módulos. Esta funcionalidad permite al usuario observar el estado visual del cultivo y tener un control adicional mediante visión remota.

1. **Recolección de dataset de imágenes**

Se llevó a cabo la captura y almacenamiento de imágenes representativas de los cultivos en distintos estados. Este dataset sirvió como base para entrenar modelos de visión artificial, orientados a detectar situaciones anómalas como falta de luz, marchitamiento o sobrecrecimiento.

1. **Entrenamiento de sistema de visión artificial**

Utilizando aprendizaje automático, se entrenó un modelo que puede interpretar imágenes del módulo hidropónico y generar alertas automáticas ante condiciones desfavorables. Este sistema complementa la monitorización por sensores, añadiendo una capa adicional de supervisión.

1. **Creación de recomendaciones personalizadas para el usuario**

Basado en los datos recogidos y procesados (tanto numéricos como visuales), se generó un sistema de recomendaciones que sugiere acciones al usuario. Estas sugerencias ayudan a optimizar el cultivo y a prevenir errores comunes, mejorando el rendimiento general del sistema.

**Control de calidad**

1. **Creación test automáticos para Apis**

Se diseñaron e implementaron pruebas automatizadas que verifican el correcto funcionamiento de los endpoints de las APIs. Estas pruebas aseguran que las funcionalidades principales de la plataforma se mantengan operativas tras cualquier actualización o modificación del sistema.

1. **Creación test automáticos para la web**

Se desarrollaron pruebas para comprobar que la interfaz web responde correctamente ante diferentes interacciones del usuario. Se validó la carga de datos, la navegación entre secciones y el correcto despliegue de los gráficos, minimizando así errores en producción.

**Tareas finales**

|  |
| --- |
| Nombre de tarea |
| * 1. Documentación de producto |
| * 1. Creación de la memoria de desarrollo y producto |
| * 1. Corrección de errores |
| * 1. Tareas de mantenimiento |

1. **Documentación de producto**

Se elaboró una documentación técnica que detalla la arquitectura del sistema, los componentes utilizados y la lógica de funcionamiento de cada módulo. Esta documentación está orientada a facilitar el mantenimiento del proyecto y su posible evolución por parte de futuros desarrolladores.

1. **Creación de la memoria de desarrollo y producto**

Se redactó la memoria oficial del Trabajo de Fin de Grado, recogiendo todo el proceso de desarrollo desde la concepción de la idea hasta la validación final. En ella se incluyen los fundamentos teóricos, decisiones técnicas, metodología utilizada y resultados obtenidos.

1. **Corrección de errores**

Durante esta fase se revisaron todas las funcionalidades del sistema, corrigiendo errores detectados en pruebas anteriores. Se depuró el código tanto del backend como del frontend, y se ajustaron pequeños detalles visuales y de rendimiento para mejorar la experiencia del usuario.

1. **Tareas de mantenimiento**

Se implementaron acciones preventivas y de mantenimiento para garantizar la estabilidad del sistema en el tiempo. Esto incluye la actualización de librerías, refactorización de código redundante y mejora de logs, así como una revisión general del entorno Dockerizado para asegurar compatibilidad futura.

### **2.2.2. Asignación de tareas y estimación**

En esta sección se muestra la asignación de tareas y estimación de duraciones organizado por grupo de desarrollo asignado.

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Desarrollador backend** | **288 horas** | **39d** |
| Creacion de la api de origen de datos | 16 horas | 2 días |
| *Desarrollador backend* | *16 horas* |  |
| Programación del productor kafka | 8 horas | 1 día |
| *Desarrollador backend* | *8 horas* |  |
| Diseño de la base de datos | 8 horas | 1 día |
| *Desarrollador backend* | *8 horas* |  |
| Creación de la base de datos | 8 horas | 1 día |
| *Desarrollador backend* | *8 horas* |  |
| Creación de api para la base de datos | 16 horas | 2 días |
| *Desarrollador backend* | *16 horas* |  |
| Programación del consumidor Kafka | 8 horas | 1 día |
| *Desarrollador backend* | *8 horas* |  |
| Recolección de dataset de imágenes | 112 horas | 14 días |
| *Desarrollador backend* | *112 horas* |  |
| Entrenamiento de sistema de vision artificial | 56 horas | 7 días |
| *Desarrollador backend* | *56 horas* |  |
| Creación test automaticos para apis | 56 horas | 7 días |
| *Desarrollador backend* | *56 horas* |  |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Desarrollador backend;Desarrollador frontend** | **72 horas** | **33d** |
| Lógica de gestión de multiples módulos | 24 horas | 3 días |
| *Desarrollador backend* | *8 horas* |  |
| *Desarrollador frontend* | *16 horas* |  |
| Creación de recomendaciones personalizadas para el usuario | 48 horas | 3 días |
| *Desarrollador backend* | *24 horas* |  |
| *Desarrollador frontend* | *24 horas* |  |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Desarrollador frontend** | **152 horas** | **52d** |
| Desarrollo estructura basica de la web | 56 horas | 7 días |
| *Desarrollador frontend* | *56 horas* |  |
| Visualización de datos de ultima lectura | 8 horas | 1 día |
| *Desarrollador frontend* | *8 horas* |  |
| Visualización de datos haciendo uso de graficas | 8 horas | 1 día |
| *Desarrollador frontend* | *8 horas* |  |
| Paneles de configuración para automatizaciones | 16 horas | 2 días |
| *Desarrollador frontend* | *16 horas* |  |
| Pagina dedicada para imágenes de los modulos | 8 horas | 1 día |
| *Desarrollador frontend* | *8 horas* |  |
| Creación test automaticos para la web | 56 horas | 7 días |
| *Desarrollador frontend* | *56 horas* |  |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Desarrollador frontend;Jefe de proyecto** | **112 horas** | **7d** |
| Boceto diseño gráfico de la aplicación web | 112 horas | 7 días |
| *Desarrollador frontend* | *56 horas* |  |
| *Jefe de proyecto* | *56 horas* |  |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Encargado hardware** | **168 horas** | **21d** |
| Diseño del circuito eléctrico | 112 horas | 14 días |
| *Encargado hardware* | *112 horas* |  |
| Programación del módulo | 56 horas | 7 días |
| *Encargado hardware* | *56 horas* |  |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Encargado hardware** | **24 horas** | **3d** |
| Montaje del circuito | 24 horas | 3 días |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Encargado hardware;Jefe de proyecto** | **64 horas** | **4d** |
| Diseño de la carcasa 3D | 64 horas | 4 días |
| *Encargado hardware* | *32 horas* |  |
| *Jefe de proyecto* | *32 horas* |  |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Encargado hardware** | **40 horas** | **5d** |
| Impresión de la carcasa | 40 horas | 5 días |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Encargado hardware** | **16 horas** | **2d** |
| Montaje del módulo | 16 horas | 2 días |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Jefe de proyecto** | **56 horas** | **12d** |
| Definición de la idea | 40 horas | 5 días |
| *Jefe de proyecto* | *40 horas* |  |
| Configuraciones del repositorio de codigo | 16 horas | 2 días |
| *Jefe de proyecto* | *16 horas* |  |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Jefe de proyecto;Desarrollador backend** | **64 horas** | **4d** |
| Creación de la arquitectura del proyecto | 64 horas | 4 días |
| *Desarrollador backend* | *32 horas* |  |
| *Jefe de proyecto* | *32 horas* |  |

|  |  |  |
| --- | --- | --- |
| Nombre de tarea | Trabajo | Duración |
| **Nombres de los recursos: Jefe de proyecto** | **40 horas** | **5d** |
| Diseño de la arquitectura base | 40 horas | 5 días |

### 2.2.3. Planificación de tareas

En la siguiente figura se observa un diagrama de Gantt con una vista general de la planificación completa del proyectoInterfaz de usuario gráfica, Gráfico

El contenido generado por IA puede ser incorrecto.

En la siguiente tabla se muestra un ejemplo de planificación concreta de desarrollo comprendida entre las fechas 11/11/24 y 21/02/25

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre | Duración | Comienzo | Fin |
| **GreenhouseIOT** | **75 días** | **lun 11/11/24** | **vie 21/02/25** |
| **Planificación inicial** | **16 días** | **lun 11/11/24** | **lun 02/12/24** |
| Definición de la idea | 5 días | lun 11/11/24 | vie 15/11/24 |
| Diseño de la arquitectura base | 5 días | lun 18/11/24 | vie 22/11/24 |
| Configuraciones del repositorio de codigo | 2 días | lun 25/11/24 | mar 26/11/24 |
| Creación de la arquitectura del proyecto | 4 días | mié 27/11/24 | lun 02/12/24 |
| **Creación del módulo** | **37 días** | **mar 03/12/24** | **mié 22/01/25** |
| Diseño del circuito eléctrico | 14 días | mar 03/12/24 | vie 20/12/24 |
| Montaje del circuito | 3 días | vie 03/01/25 | mar 07/01/25 |
| Diseño de la carcasa 3D | 4 días | mié 08/01/25 | lun 13/01/25 |
| Impresión de la carcasa | 5 días | jue 16/01/25 | mié 22/01/25 |
| Montaje del módulo | 2 días | mar 14/01/25 | mié 15/01/25 |
| Programación del módulo | 7 días | lun 23/12/24 | mar 31/12/24 |
| **Origen de datos** | **8 días** | **mar 04/02/25** | **jue 13/02/25** |
| Creacion de la api de origen de datos | 2 días | mar 04/02/25 | mié 05/02/25 |
| Programación del productor kafka | 1 día | jue 13/02/25 | jue 13/02/25 |
| **Persistencia de datos** | **5 días** | **jue 06/02/25** | **mié 12/02/25** |
| Diseño de la base de datos | 1 día | jue 06/02/25 | jue 06/02/25 |
| Creación de la base de datos | 1 día | vie 07/02/25 | vie 07/02/25 |
| Creación de api para la base de datos | 2 días | lun 10/02/25 | mar 11/02/25 |
| Programación del consumidor Kafka | 1 día | mié 12/02/25 | mié 12/02/25 |
| **Explotación de datos** | **54 días** | **mar 03/12/24** | **vie 14/02/25** |
| Boceto diseño gráfico de la aplicación web | 7 días | mar 03/12/24 | mié 11/12/24 |
| Desarrollo estructura basica de la web | 7 días | jue 12/12/24 | vie 20/12/24 |
| Visualización de datos de ultima lectura | 1 día | jue 26/12/24 | jue 26/12/24 |
| Visualización de datos haciendo uso de graficas | 1 día | lun 23/12/24 | lun 23/12/24 |
| Paneles de configuración para automatizaciones | 2 días | vie 27/12/24 | lun 30/12/24 |
| Lógica de gestión de multiples módulos | 3 días | mié 01/01/25 | vie 14/02/25 |
| **Sistema visión artificial** | **25 días** | **mar 31/12/24** | **lun 03/02/25** |
| Pagina dedicada para imágenes de los modulos | 1 día | mar 31/12/24 | mar 31/12/24 |
| Recolección de dataset de imágenes | 14 días | mié 01/01/25 | lun 20/01/25 |
| Entrenamiento de sistema de vision artificial | 7 días | mar 21/01/25 | mié 29/01/25 |
| Creación de recomendaciones personalizadas para el usuario | 3 días | jue 30/01/25 | lun 03/02/25 |
| **Control de calidad** | **45 días** | **lun 23/12/24** | **vie 21/02/25** |
| Creación test automaticos para apis | 7 días | lun 23/12/24 | mar 31/12/24 |
| Creación test automaticos para la web | 7 días | jue 13/02/25 | vie 21/02/25 |
| HITO: Entrega del proyecto | 0 días | vie 21/02/25 | vie 21/02/25 |

### 2.3. Gestión de recursos

En este apartado se van a especificar cuáles han sido los diferentes recursos utilizados en el desarrollo. En la siguiente imagen se muestra el listado completo de recursos. Muchos de ellos cuentan con características particulares especificadas en el siguiente apartado.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nombre del recurso | Tipo | Etiqueta de material | Iniciales | Grupo | Capacidad máxima | Tasa estándar | Tasa horas extra | Costo/Uso | Acumular |
| Jefe de proyecto | Trabajo |  | J |  | 100% | 30,00 €/hora | 0,00 €/hora | 0,00 € | Prorrateo |
| Desarrollador frontend | Trabajo |  | D |  | 100% | 25,00 €/hora | 0,00 €/hora | 0,00 € | Prorrateo |
| Desarrollador backend | Trabajo |  | D |  | 100% | 25,00 €/hora | 0,00 €/hora | 0,00 € | Prorrateo |
| Encargado hardware | Trabajo |  | E |  | 100% | 25,00 €/hora | 0,00 €/hora | 0,00 € | Prorrateo |
| Impresora 3D | Material | und | I |  |  | 319,00 € |  | 0,00 € | Comienzo |
| Plástico PLA | Material | bobina 850g | P |  |  | 21,00 € |  | 0,00 € | Comienzo |
| Portatiles | Material | und | P |  |  | 429,00 € |  | 0,00 € | Comienzo |
| Cableado | Material | metros | C |  |  | 11,49 € |  | 0,00 € | Comienzo |
| DHT22 | Material | und | D |  |  | 4,63 € |  | 0,00 € | Comienzo |
| Sensor temperatura del agua | Material | und | S |  |  | 1,39 € |  | 0,00 € | Comienzo |
| Sensor TDS | Material | und | S |  |  | 3,49 € |  | 0,00 € | Comienzo |
| Fuente de alimentación | Material | und | F |  |  | 7,24 € |  | 0,00 € | Comienzo |
| Regulador de voltaje | Material | und | R |  |  | 2,97 € |  | 0,00 € | Comienzo |
| Foco led | Material | und | F |  |  | 5,79 € |  | 0,00 € | Comienzo |
| Tornillería | Material | caja | T |  |  | 4,79 € |  | 0,00 € | Comienzo |
| LDR | Material | und | L |  |  | 1,42 € |  | 0,00 € | Comienzo |
| Resistencias | Material | caja | R |  |  | 2,92 € |  | 0,00 € | Comienzo |
| Modulo 4x Relé | Material | und | M |  |  | 2,42 € |  | 0,00 € | Comienzo |
| Conexiones | Material | caja | C |  |  | 9,80 € |  | 0,00 € | Comienzo |
| ESP32 S3 | Material | und | E |  |  | 6,03 € |  | 0,00 € | Comienzo |
| ESP32 Cam | Material | und | E |  |  | 7,49 € |  | 0,00 € | Comienzo |
| Contenedores | Material | und | C |  |  | 4,50 € |  | 0,00 € | Comienzo |
| Oxigenador | Material | und | O |  |  | 2,49 € |  | 0,00 € | Comienzo |
| Ventilador | Material | und | V |  |  | 1,54 € |  | 0,00 € | Comienzo |
| Pegamento | Material | caja | P |  |  | 3,50 € |  | 0,00 € | Comienzo |
| Tuercas inyección | Material | caja | T |  |  | 3,99 € |  | 0,00 € | Comienzo |
| pcb | Material | caja | p |  |  | 2,84 € |  | 0,00 € | Comienzo |
| lente 77mm | Material | und | l |  |  | 6,36 € |  | 0,00 € | Comienzo |
| Nivel de agua | Material | und | N |  |  | 2,79 € |  | 0,00 € | Comienzo |
| Soldador | Material | kit | S |  |  | 21,00 € |  | 0,00 € | Comienzo |

### 2.3.1. Especificación de recursos

* Impresora 3D

Imagen que contiene tabla, cámara, hombre, sostener

El contenido generado por IA puede ser incorrecto.La impresora 3D utilizada ha sido la Ender-3 V3 KE de Creality.

Características principales:

* Velocidad de hasta 500 mm/s con aceleración de 8000 mm/s²
* El extrusor directo "Sprite" permite una impresión a 300 ℃ y una alimentación suave de varios filamentos
* Riel lineal de eje X
* 3 formas de impresión, control por teléfono, WiFi y unidad USB
* Ensamblaje rápido | Autoprueba con un toque | Carga automática de filamentos
* Admite filamentos Hyper PLA, PETG, ABS, TPU(95A) y ASA.
* Filamento PLA+

Imagen en blanco y negro

El contenido generado por IA puede ser incorrecto.El filamento empleado ha sido bobinas de 850g de PLA+ de la empresa LEON3D

* Tamaño de bobina: 850g
* Diámetro estándar de 1.75 mm
* Temperatura de la cama recomendada: 50 ºC
* Temperatura de impresión: 212-230 ºC
* Portatiles



Portátil HP 255 G9

* AMD Ryzen 5 5625U
* 8GB de memoria
* 512GB SSD de disco
* 15.6" de pantalla
* Cableado

Cableado rígido

* 22AWG
* 2m
* Núcleo de cobre sólido
* DHT22

Imagen en blanco y negro

El contenido generado por IA puede ser incorrecto.Sensor de humedad y temperatura DHT 22

Tamaño: 28,2 mm (largo) \* 13,1 mm (ancho) \* 10 mm (alto)

- Voltaje de funcionamiento: 3 V - 5,5 V

- Forma de salida de señal: señal digital

- Rango de medición de temperatura: - 40°c a 80°c

- Precisión de medición: 0,5°c

- El rango de medición de humedad: 0-100% RH

- Precisión de medición: 2% RH

* Sensor temperatura del agua

Un dibujo de un cable

El contenido generado por IA puede ser incorrecto.

Sensor de temperatura DS18B20

* Tensión de alimentación del sensor de temperatura: 3,0 V ~ 5,5 V
* Resolución del sensor de temperatura: resolución ajustable de 9 a 12
* Rango de temperatura: -55 ~ +125°
* Uso compatible en agua
* Sensor TDS

Cable de color blanco

El contenido generado por IA puede ser incorrecto.

Sensor TDS (Total de sólidos disueltos)

* Funcionamiento de amplio voltaje: 3,3 ~ 5,5 V
* Salida de señal analógica de 0 ~ 2,3 V, compatible con sistemas de control de 5 V y 3,3 V
* La fuente de excitación es una señal de CA, que previene eficazmente la polarización de la sonda.
* Sonda resistente al agua, se puede sumergir en agua para mediciones a largo plazo.
* Fuente de alimentación



Fuente de alimentación 72W

* Voltaje: 12V
* Amperaje: 6A
* Regulador de voltaje

Imagen de la pantalla de un video juego

El contenido generado por IA puede ser incorrecto.

Regulador 300W XL4016 DC-DC

* Max 9A
* Convertidor reductor 5-40V a 1,2-35V
* Módulo de fuente de alimentación ajustable
* Foco led

Lámpara de mesa

El contenido generado por IA puede ser incorrecto.

Luz de crecimiento LED

* Voltaje: 5v
* Ajustable
* Cuerpo de aluminio
* Tornillería

Imagen que contiene foto, diferente, llenado, caja

El contenido generado por IA puede ser incorrecto.

Tornillería M2 y M3

* 100 tornillos M2
* 100 tornillos M3
* 100 tuercas M2
* 100 tuercas M3
* LDR

Imagen que contiene esquiando, nieve, vistiendo, montar a caballo

El contenido generado por IA puede ser incorrecto.

Resistencia fotoeléctrica 5528

* Resistencia sensible a la luz
* Resistencias

Un grupo de folletos sobre una superficie blanca

El contenido generado por IA puede ser incorrecto.

Set de resistencias

* Set de 600 und
* Múltiples valores de resistencias
* Modulo 4x Relé

Un circuito electrónico

El contenido generado por IA puede ser incorrecto.Módulo de relés HW-316A 5V

* Funcionamiento a 5V
* Led de comprobación rápida
* Alimentación unificada
* Conexiones

Imagen que contiene luz

El contenido generado por IA puede ser incorrecto.

Conectores macho hembra DC 12V

* Pack de 5 pares.
* Compatible con 5V
* Imagen que contiene circuito, electrónica

  El contenido generado por IA puede ser incorrecto.ESP32 S3

S3-N16R8

* Wifi
* Bluetooth
* N16R8
* 44Pin
* PSRAM
* ESP32 Cam

Un circuito electrónico

El contenido generado por IA puede ser incorrecto.

ESP32-S3 WROOM N16R8

* Modulo preparado para cámara
* Cámara OV2640
* Oxigenador

Imagen en blanco y negro

El contenido generado por IA puede ser incorrecto.Oxigenador G613A

* Voltaje: 5V
* Potencia: 1W
* Ventilador

Logotipo

El contenido generado por IA puede ser incorrecto.

Ventilador 6010

* 5V
* 60mm x 60mm x 10mm
* 2 pines
* 3500 rpm
* Imagen que contiene Tabla

  El contenido generado por IA puede ser incorrecto.Tuercas inserción

Tuercas de inserción térmica de latón

* 220 unidades
* Compatible con PLA
* Distintos diámetros
* Pcb

Imagen que contiene Patrón de fondo

El contenido generado por IA puede ser incorrecto.

Placa PCB

* 10 unidades
* 3x7 a doble cara
* lente 75mm

Flecha

El contenido generado por IA puede ser incorrecto.

* Nivel de agua
* Soldador

## 2.4. Gestión de riesgos

2.4.1. Identificación de riesgos2.5. Legislación y normativa

Capítulo 3

# Solución

## 3.1. Descripción de la solución3.2. El proceso de desarrollo

### 3.2.1. Análisis3.2.2. Diseño

## 3.3. El producto del desarrollo

Capítulo 4

# Evaluación

## 4.1. Proceso de evaluación

## 4.1.1. Forma de evaluación

## 4.1.2. Casos de prueba4.2. Análisis de resultadosConclusión

# Lista de referencias

# Anexo 1 - Seguimiento de Proyecto

# Anexo 2 - Manual de usuario