Sistema de monitoreo y gestión remota del clima en invernaderos

Autor

Lic. Martín Anibal Lacheski

Director del trabajo

Mg. Lic. Leopoldo Alfredo Zimperz (FIUBA)

Este plan de trabajo ha sido realizado en el marco de la asignatura Gestión de Proyectos entre el 20 de agosto de 2024 y el 8 de octubre de 2024

Tabla de contenido

Revisión	Cambios realizados	Fecha
1.0	Creación del documento	28/03/2025
1.1	Actualización de avance	16/04/2025
1.2	Actualización de avance	25/04/2025

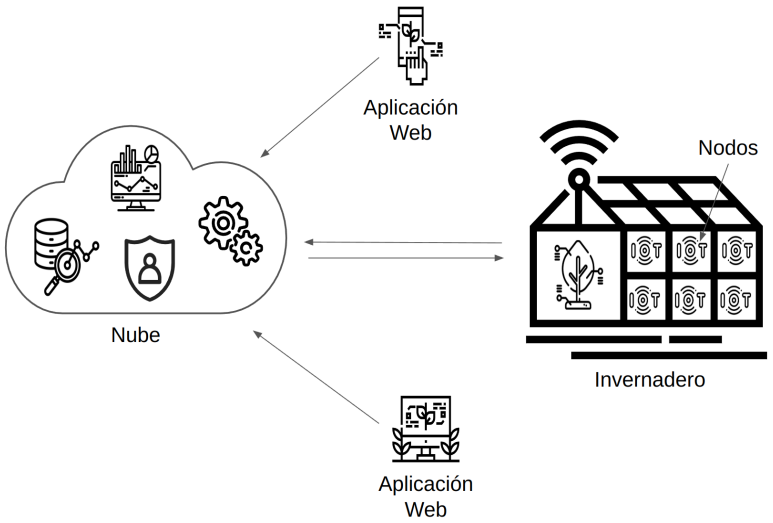
1. Breve resumen del trabajo realizado hasta la fecha

Elabore un detalle del estado del proyecto a la fecha. **Utilicé como mínimo dos páginas completas y como máximo tres páginas**. Explique muy brevemente en qué consiste su Trabajo Final, **aunque esa información esté más detallada en el Plan de Trabajo al cual su Jurado también tiene acceso**. Incluya imágenes y tablas según considere apropiado. **Indique con claridad por qué estima que podrá completar todos los faltantes** (o al menos la gran mayoría) **antes del inicio del Taller de Trabajo Final.**

La agricultura enfrenta desafíos en la optimización de la productividad y eficiencia, especialmente en regiones con condiciones climáticas adversas. Según la FAO, para 2050 se necesitará aumentar la producción de alimentos en un 60% debido al crecimiento de la población mundial. Los cultivos hidropónicos ofrecen una solución eficiente, al reducir el consumo de agua, permitir el cultivo durante todo el año y mejorar la productividad.

En Misiones, la producción hidropónica ha crecido, pero persisten problemas en la gestión de recursos debido a sistemas de control inadecuados que requieren intervenciones manuales y afectan la eficiencia, calidad y sostenibilidad de los cultivos.

Este trabajo desarrolla e implementa un sistema IoT de bajo costo para monitorear y controlar en tiempo real los invernaderos de la Facultad de Ciencias Forestales de la Universidad Nacional de Misiones. El sistema registra variables como temperatura, humedad, CO2, niveles de nutrientes y consumo de agua y energía, con datos disponibles para docentes, estudiantes e investigadores. La figura siguiente muestra el diagrama en bloques del sistema.



El proyecto impacta tanto en la producción como en la formación académica y el avance científico, proporcionando una plataforma para el análisis y el desarrollo de soluciones tecnológicas enfocadas en sostenibilidad ambiental y seguridad alimentaria.

Estado del trabajo

Al día de la fecha, se han realizado las siguientes tareas:

1. Investigación: Se ha realizado una investigación y selección de los microcontroladores, sensores y actuadores necesarios para la implementación del sistema.

Además se investigaron y seleccionaron los siguientes lenguajes de programación, tecnologías y herramientas de software:

- **Firmware**: se seleccionó MicroPython como lenguaje para programar los microcontroladores. Su facilidad de uso, la amplia disponibilidad de bibliotecas y la reducción del tiempo de desarrollo lo convierten en una opción eficiente.

- **Backend**: se decidió implementar el framework FastAPI. Esta herramienta utiliza Python y aprovecha las características de tipado estático, de manera de proporcionar una API auto documentada y la validación de los tipos de datos.

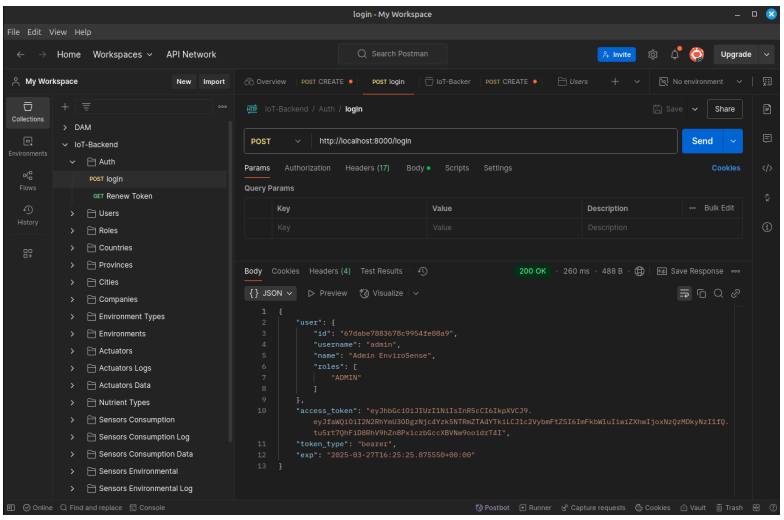
- **Base de datos**: para persistir los datos, se decidió utilizar MongoDB, que es una base de datos de tipo NoSQL orientada a documentos. Su flexibilidad, escalabilidad y capacidad para manejar datos no estructurados lo hacen una elección popular en el mundo del desarrollo de software moderno.

- **Frontend**: se decidió utilizar la biblioteca de JavaScript React, desarrollada por Facebook. Desde su introducción, React se ha convertido en una de las opciones más populares y ampliamente adoptadas para el desarrollo de interfaces de usuario interactivas y dinámicas.

- **Infraestructura**: se decidió utilizar Docker para poder construir, empaquetar y desplegar la aplicación en servicios en la nube.

- **Servicios cloud:** se seleccionaron servicios de AWS para poder desplegar la solución propuesta. Para ello se implementa AWS IoT Core para poder permitir a los dispositivos comunicarse a través del protocolo MQTT de manera segura y el servicio de AWS EC2 en el cual se despliega la base de datos, backend y frontend en contenedores de Docker.

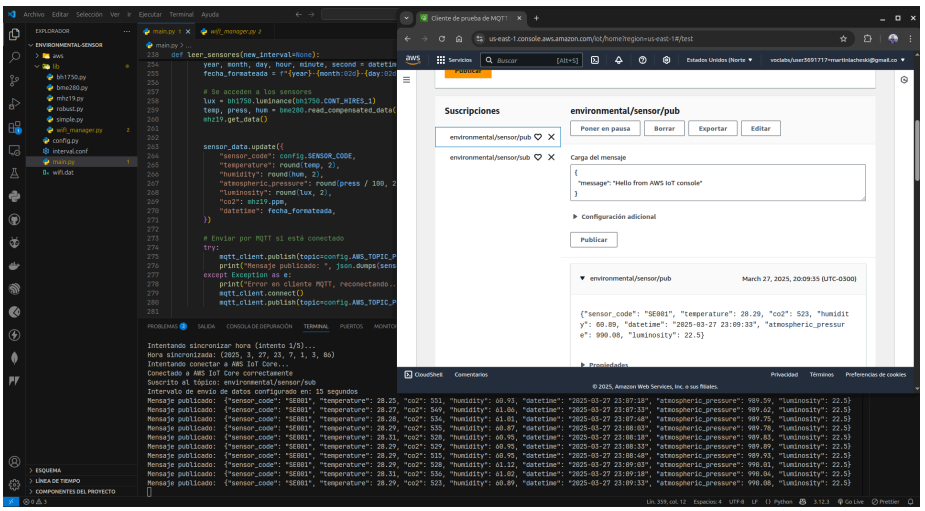
2. Desarrollo del backend: Se desarrollaron todos los Endpoints necesarios para la funcionalidad prevista. También, se implementaron las medidas de autenticación con JWT (*Json Web Token*). Se realizó la integración con la base de datos para poder persistir la información. Resta realizar el contenedor Docker e integrar la funcionalidad de recepción y envío de mensajes con MQTT y WebSocket. En la siguiente imagen puede visualizarse la funcionalidad del endpoint /auth/login que genera el JWT de acceso verificado desde la aplicación Postman.



3. Desarrollo del frontend: Se desarrolló el sistema de login con credenciales, la interfaz intuitiva y accesible desde teléfonos móviles y de escritorio. Se realizaron las operaciones CRUD necesarias. Resta desarrollar la visualización en tiempo real, la funcionalidad para enviar comandos a los sensores y actuadores y los reportes para visualizar los datos históricos de los sensores y actuadores. La siguiente figura permite visualizar el dashboard de la aplicación web.

4. Desarrollo del firmware de los nodos sensores y actuadores: Se desarrollaron diversos firmware para visualizar cómo se reciben los datos de los sensores y actuadores utilizados. Se desarrolló el firmware del sensor ambiental. Esto incluye la implementación de los certificados, la elección y conexión a una red Wi-Fi. La publicación de los datos de los sensores a un tópico dado, mediante el protocolo MQTT. La suscripción a un tópico determinado para poder cambiar el tiempo establecido para el envío de los datos. Resta desarrollar el firmware necesario para implementar el sensor de solución nutritiva, el sensor de consumos y el nodo actuador.

5. Implementación en servicios cloud: Se realizó la configuración del servicio AWS IoT Core, se creó el dispositivo sensor ambiental y se descargaron los certificados para el dispositivo. Se establecieron las políticas para publicación y recepción de mensajes y se comprobó el envío y recepción de mensajes en el tópico establecido. Resta implementar la creación de los otros dispositivos y desplegar el servicio AWS EC2 que contendrá la aplicación IoT. En la siguiente figura puede visualizarse la conexión del sensor ambiental y el envío de los datos a AWS IoT Core.



2. Avance en las tareas

a) Indicar a continuación para cada una de las tareas su estado de situación según su criterio, utilizando verde si considera que es satisfactorio, amarillo si considera que es insatisfactorio por sobrecostos y/o demoras, y rojo si lo considera muy insatisfactorio por sobrecostos y/o demoras.
Si a la fecha de completar este informe no está previsto que la tarea haya comenzado entonces deje la celda correspondiente en blanco, sin pintarla con ningún color.

En subcelda inferior izquierda colocar:

- ** si los recursos u horas utilizadas fueron o están siendo muy inferior a lo planificado.
- * si los recursos u horas utilizadas fueron o están siendo inferior a lo planificado.
- $ si los recursos u horas utilizadas fueron o están siendo de acuerdo a lo planificado.
- $$ si los recursos u horas utilizadas fueron o están siendo superior a lo planificado.
- $$$ si los recursos u horas utilizadas fueron o están siendo muy superior a lo planificado.

En subcelda inferior derecha colocar:

- -- si la tarea se ejecutó o se está ejecutando mucho más rápido de lo previsto
- - si la tarea se ejecutó o se está ejecutando más rápido de lo previsto
- = si la tarea se ejecutó o se está ejecutando en el tiempo previsto.
- + si la tarea se ejecutó o se está ejecutando con demoras.
- ++ si la tarea se ejecutó o se está ejecutando con demoras muy significativas.

IMPORTANTE: Indicar con borde grueso las tareas que forman parte del camino crítico

En el transcurso del desarrollo del trabajo hubo algunas modificaciones de los requerimientos. Las mismas son descritas a continuación:

- Las tareas (4.3 y 4.4) y las tareas (4.5 y 4.6) se unificaron en una sola. Se desarrollan los requerimientos en uno solo.
- La tarea 6.2 se elimina, debido a que no se instalan certificados TLS en el frontend. Inicialmente se planteaba un cliente conectado por MQTT en el frontend, pero se implementó el cliente en el backend y se envían los datos por websocket al frontend.

1.1 Planificación del trabajo		1.2 Diseño arquitectura trabajo		2.1 Investigación de los protocolos MQTT, HTTP y WebSockets		2.2 Investigación de certificado TLS	
$	=			$	=	$	=
2.3 Investigación de los microcontroladores		2.4 Investigación de sensores y actuadores		2.5 Investigación de la base de datos a utilizar		2.6 Investigación del backend a utilizar	
$	=	$	=	$	=	$	=
2.7 Investigación del frontend a utilizar		2.8 Investigación de la plataforma IoT		2.9 Entorno de desarrollo		3.1 Configuración y conexionado	
$	=	$	=	$	=	$	=
3.2 Integración de los componentes		4.1 Configuración microcontroladores		4.2 Implementación Certificados TLS		4.3 Implementación nodo ambiental	
$	=	$	+	$	+	$	+
~~4.4 Implementación sensor CO2~~		4.5 Implementación nodo solución nutritiva		~~4.6 Implementación nivel y temperatura solución nutritiva~~		4.7 Implementación nodo de consumos	
$	+	$	+	$	+	$	+
4.8 Implementación nodo actuador		5.1 Configuración entorno backend		5.2 Implementación Certificados TLS		5.3 Implementación del broker MQTT	
$	+	$	=	$	=	$	=
5.4 Configuración de JWT		5.5 Configuración de la base de datos		5.6 Configuración de WebSocket		5.7 Endpoints de usuarios	
$	=	$	=	$	=	$	=
5.9 Endpoints de ambientes		5.11 Endpoints sensores		5.13 Endpoints actuadores		5.15 Endpoints mediciones sensores	
$	=	$	=	$	=	$	=

5.17 Endpoints estado actuadores		5.19 Parámetros enviados sensores		5.21 Parámetros enviados actuadores		6.1 Configuración entorno frontend	
$	=	$	=	$	=	$	=
~~6.2 Implementación de TLS~~		6.3 Implementación de JWT		6.4 Implementación de WebSocket		6.5 Implementación de login	
$	=	$	=	$	=	$	=
6.7 CRUD de usuarios		6.9 CRUD de ambientes		6.11 CRUD de sensores		6.13 CRUD de actuadores	
$	=	$	=	$	=	$	=
6.15 Histórico de mediciones de sensores		6.17 Histórico de estados de actuadores		6.19 Envío de parámetros a los sensores		6.21 Envío de parámetros a los actuadores	
$	=	$	=	$	=	$	=
6.23 Interfaz principal del sistema		7.1 Pruebas de los microcontroladores		7.2 Pruebas del backend		7.3 Pruebas del frontend	
$	=	$	=	$	=	$	=
8.1 Informe de avance		8.2 Vídeo demostración sistema		8.3 Estructura de la memoria		8.4 Escritura de la memoria	
$	=	$	=	$	=	$	=
8.5 Revisión de la memoria		8.6 Presentación final					
$	=	$	=				

3. Cumplimiento de los requerimientos

a) Indicar a continuación para cada uno de los requerimientos el estado de situación según su criterio, utilizando verde si considera que ya se ha cumplido, amarillo si considera que aún no se a cumplido pero se podrá cumplir, y rojo si considera que aún no se ha cumplido y tiene dudas si se podrá cumplir.

Si considera que es necesario modificar los requerimientos respecto a los indicados en la planificación inicial entonces incluya acá los requerimientos actualizados, **marcando en negrita** aquellos que son nuevos o se han modificado.

Requerimientos asociados a los nodos sensores:

Req #1.1. Cada nodo deberá contar con un microcontrolador basado en ESP32.
Req #1.2. Cada nodo deberá implementar certificados TLS.
Req #1.3. Cada nodo deberá conectarse a una red Wi-Fi.
Req #1.4. Cada nodo deberá tener un identificador único dentro del sistema.
Req #1.5. Cada nodo deberá permitir configurar de manera remota el tiempo para enviar los datos recolectados.
Req #1.6.1 El nodo deberá enviar al servidor IoT la temperatura ambiente, humedad relativa, presión atmosférica, nivel de luminosidad y nivel de dióxido de carbono.
~~Req #1.6.2 El nodo deberá enviar al servidor IoT el nivel de dióxido de carbono.~~
Req #1.6.2 El nodo deberá enviar al servidor IoT el nivel de pH, CE y TDS; el nivel y temperatura de la solución nutritiva.
~~Req #1.6.4 El nodo deberá enviar al servidor IoT el nivel y temperatura de la solución nutritiva.~~
Req #1.6.3 El nodo deberá enviar al servidor IoT el consumo de agua, nutrientes y energía eléctrica.

Requerimientos asociados a los nodos actuadores:

Req #2.1. Cada nodo deberá contar con un microcontrolador basado en ESP32.
Req #2.2. Cada nodo deberá implementar certificados TLS.
Req #2.3. Cada nodo deberá conectarse a una red Wi-Fi.
Req #2.4. Cada nodo deberá tener un identificador único dentro del sistema.
Req #2.5. Cada nodo deberá contar con diez canales.
Req #2.6. Cada nodo deberá permitir configurar de manera remota los parámetros del actuador y cada canal.
Req #2.7. Cada nodo deberá enviar al servidor IoT el estado de cada canal.
Req #2.8. Cada nodo deberá permitir activar de manera remota cada canal.

Requerimientos asociados al broker MQTT:

Req #3.1. Deberá contar con un broker MQTT que soporte conexiones seguras mediante TLS.
Req #3.2. Deberá gestionar las suscripciones y publicaciones de los datos enviados desde los nodos y permitir la comunicación bidireccional para el envío de comandos.
Req #3.3. Deberá implementar QoS (Quality of Service) para garantizar la entrega de los mensajes.

Requerimientos asociados al frontend:

Req #4.1. La interfaz deberá ser intuitiva y accesible desde dispositivos móviles y de escritorio.
Req #4.2. Deberá permitir el acceso al sistema a usuarios debidamente autenticados.
Req #4.3. Deberá permitir realizar el CRUD de usuarios.
Req #4.4. Deberá permitir realizar el CRUD de ambientes.
Req #4.5. Deberá permitir realizar el CRUD de sensores asociados a un determinado ambiente.
Req #4.6. Deberá permitir realizar el CRUD de actuadores asociados a un determinado ambiente.
~~Req #4.7. Deberá permitir visualizar en tiempo real los datos recibidos de los sensores y actuadores.~~
Req #4.7. Deberá permitir enviar configuraciones de tiempo de envío de mensajes a los nodos sensores.

Req #4.8. Deberá permitir enviar configuraciones de tiempo de envío de mensajes a los nodos actuadores.
Req #4.9. Deberá permitir enviar parámetros a los canales de los nodos actuadores.
Req #4.10. Deberá permitir visualizar los datos históricos de los nodos sensores de un ambiente.
Req #4.11. Deberá permitir visualizar los datos históricos de los nodos actuadores de un ambiente.
Req #4.12. Deberá permitir la visualización en tiempo real de los nodos sensores y actuadores de un ambiente a través de un tablero interactivo.
Req #4.13. Generar contenedor del frontend en Docker.
Req #4.14. Desplegar el contenedor del frontend en la instancia EC2 y de AWS.

Requerimientos asociados al backend:

Req #5.1. Deberá permitir conexiones seguras con el broker MQTT mediante TLS.
Req #5.2. Deberá poder implementar JWT (JSON Web Token) para poder propagar entre dos partes, y de forma segura, la identidad de un determinado usuario.
Req #5.3. Deberá soportar los métodos HTTP para realizar operaciones CRUD y visualizar los reportes, WebSockets para la visualización en tiempo real de los datos y MQTT para la interacción con los sensores y actuadores.
Req #5.4. Deberá poder persistir la información de los usuarios.
Req #5.5. Deberá poder persistir la información de los ambientes.
Req #5.6. Deberá poder persistir la información de los sensores.
Req #5.7. Deberá poder persistir la información de los actuadores.
Req #5.8. Deberá poder persistir la información histórica de las mediciones de los sensores.
Req #5.9. Deberá poder persistir la información histórica de los estados de los actuadores.
Req #5.10. Deberá poder persistir la información histórica de los parámetros enviados a los sensores.
Req #5.11. Deberá poder persistir la información histórica de los parámetros enviados a los actuadores.
Req #5.12. Deberá permitir la autenticación y autorización de usuarios autorizados para acceder a los recursos.
Req #5.13. Deberá incluir los endpoints para permitir realizar el CRUD de usuarios.
Req #5.14. Deberá incluir los endpoints para permitir realizar el CRUD de ambientes.
Req #5.15. Deberá incluir los endpoints para permitir realizar el CRUD de sensores asociados a un determinado ambiente, con la configuración de los parámetros específicos.
Req #5.16. Deberá incluir los endpoints para permitir realizar el CRUD de actuadores asociados a un determinado ambiente, con la configuración de los parámetros específicos por canal.
Req #5.17. Deberá incluir el endpoint para registrar las mediciones de los sensores.
Req #5.18. Deberá incluir el endpoint para registrar los estados de los actuadores.
Req #5.19. Deberá incluir el endpoint para obtener el histórico de las mediciones de los sensores.
Req #5.20. Deberá incluir el endpoint para obtener el histórico de los estados de los actuadores.
Req #5.21. Deberá incluir un endpoint para el envío de parámetros a los sensores.
Req #5.22. Deberá incluir un endpoint para el envío de parámetros a los actuadores.
Req #5.23. Generar contenedores del backend y la base de datos.
Req #5.24. Desplegar los contenedores del backend y base de datos en la instancia EC2 y de AWS.

Requerimientos asociados a la documentación:

Req #6.1. Se entregará el código del sistema, que incluye todos los componentes desarrollados (sensores, actuadores, broker MQTT, frontend, backend y API).
Req #6.2. Se entregarán las guías y diagramas de instalación, configuración y operación.
Req #6.3. Se desarrollará un informe de avance al finalizar el taller de trabajo final A.
Req #6.4. Se desarrollará la memoria del proyecto al finalizar el taller de trabajo final B.

4. Gestión de riesgos

a) Indicar a continuación para cada uno de los riesgos el estado de situación según su criterio, utilizando verde si considera que el riesgo ya no se manifestará o es muy improbable que se manifieste, amarillo si considera que es posible que es improbable que el riesgo se manifieste o si se manifiesta estima que será fácilmente controlado, y rojo si considera que es muy probable que el riesgo se manifieste y que no pueda ser controlado fácilmente.

Si considera que es necesario modificar los riesgos respecto a los presentados en la planificación inicial entonces incluya acá los riesgos actualizados, **marcando en negrita** aquellos que son nuevos o se han modificado, e indicando para ellos los valores de S, O y RPN, junto con su respectiva justificación.

Riesgo #1: Imposibilidad de cumplir con los plazos establecidos.
Riesgo #2: Retraso en la disponibilidad de los componentes para desarrollar el prototipo.
Riesgo #3: Selección inadecuada de actuadores, sensores y microcontroladores.
Riesgo #4: Falta de conocimientos adecuados para el desarrollo del trabajo.
Riesgo #5: Cambio de trabajo del responsable del trabajo.