



DEPARTAMENTO DE INGENIERÍA ELÉCTRICA  
FACULTAD DE INGENIERÍA  
UNIVERSIDAD DE CONCEPCIÓN  
CONCEPCIÓN, CHILE.



## Proyecto TIC 2: "Home Connected"

*Profesor: Vincenzo Caro*

*Integrantes: Rodrigo Hernández A.*

*Javier Herrera I.*

*7 de julio de 2024*

# Índice

<b>1. Introducción</b>	<b>1</b>
<b>2. Desarrollo</b>	<b>2</b>
2.1. Detalles del proyecto . . . . .	2
2.2. Implementación . . . . .	2
2.2.1. Luces . . . . .	5
2.2.2. Diagrama . . . . .	6
2.2.3. Bibliotecas y codigos . . . . .	7
2.3. Resultados . . . . .	7
<b>3. Conclusión</b>	<b>9</b>

## 1. Introducción

Las aplicaciones creadas con Arduino son diversas. Con Arduino, es posible crear desde simples proyectos educativos hasta complejos sistemas de automatización industrial. Algunas de las aplicaciones más comunes incluyen el desarrollo de sistemas de monitoreo y control remoto, dispositivos de Internet de las Cosas (IoT), prototipos de dispositivos electrónicos, proyectos de robótica, sistemas de domótica, y aplicaciones en el campo de la salud y el bienestar. La capacidad de Arduino para integrarse con diversas tecnologías y su extensa comunidad de usuarios hacen de esta plataforma una opción ideal para innovar y experimentar. Pero, ¿Qué es Arduino?, Arduino es una plataforma de código abierto. Consiste en una placa de desarrollo físico, que puede ser programada para interactuar con el entorno a través de sensores, actuadores y otros dispositivos electrónicos. Su accesibilidad y flexibilidad la han convertido en una herramienta popular tanto para principiantes como para profesionales en el campo de la electrónica y la programación.

Para este proyecto se realizó un sistema al cual llamamos "Home Connected". Este sistema está compuesto por una interfaz gráfica de usuario intuitiva y fácil de comprender, sensores de temperatura y humedad que monitorean continuamente las condiciones ambientales del hogar, un Bot de Telegram para la fácil lectura de datos y un sistema de manejo de luces a través de comandos de voz utilizando la tecnología de Amazon Alexa. Además, el sistema incluye la capacidad de integrarse con otros dispositivos inteligentes, permitiendo una gestión centralizada de múltiples aspectos del hogar.

## 2. Desarrollo

### 2.1. Detalles del proyecto

Este proyecto de domótica consiste en crear una casa inteligente utilizando dos microcontroladores ESP32 y una Raspberry Pi. El primer ESP32 se encarga de recolectar datos de temperatura y humedad de varios sensores distribuidos por la casa y enviar estos datos a una Raspberry Pi. La Raspberry Pi corre una interfaz gráfica creada con PyQt5 que muestra un plano de la casa y la ubicación de los sensores junto con sus niveles de temperatura y humedad. Además, la Raspberry está conectada con un Bot de Telegram al cual se puede consultar los datos de humedad y temperatura de las habitaciones. El segundo ESP32 está conectado a varias luces y un ventilador, los cuales se controlan mediante comandos de voz a través de Alexa utilizando la biblioteca 'Espalexa'.

En el desarrollo del proyecto se realizó el análisis de varios componentes, entre ellos distintos tipos de sensores de temperatura u otros sistemas de comandos de voz como Google Home o Apple HomePod, finalmente se decidió por lo más cómodo de trabajar, siendo estos:

- **Sensores de temperatura y humedad DHT11.**
- **2 x ESP32.**
- **6 x Luces LED.**
- **RaspberryPi 3B+.**
- **Conectores y cables.**
- **Fuente de alimentación para ESP32 y Raspberry Pi.**
- **Amazon Alexa.**
- **Modem/Router WiFi.**
- **Interfaz Gráfica.**
- **Pantalla OLED 128x64 I2C Monochrome.**
- **Módulo Relay KY-019.**
- **Mini ventilador de 12 V.**
- **Jack 12V.**
- **5 x Key Switch Module.**
- **Power Bank.**

### 2.2. Implementación

Para implementar el proyecto, se desarrolló una interfaz gráfica con PyQt5 y *Designer* que permite visualizar la temperatura de cada sección del hogar de manera eficiente. Se optó por un diseño simple y entendible, donde la temperatura se muestra visualmente en tiempo real. Los sensores de temperatura están conectados a un microcontrolador ESP32, que se comunica por Wi-Fi con una Raspberry Pi 3B+ por protocolo HTTP. Esta última realiza la lectura de los datos y los muestra en la interfaz gráfica, facilitando el trabajo de traslado de la información. La elección del ESP32 se debe a su versatilidad y

capacidad de comunicación inalámbrica, lo que entrega la posibilidad de añadir otros tipos de sensores en el futuro, como sensores de gas, sensores de movimiento, etc.

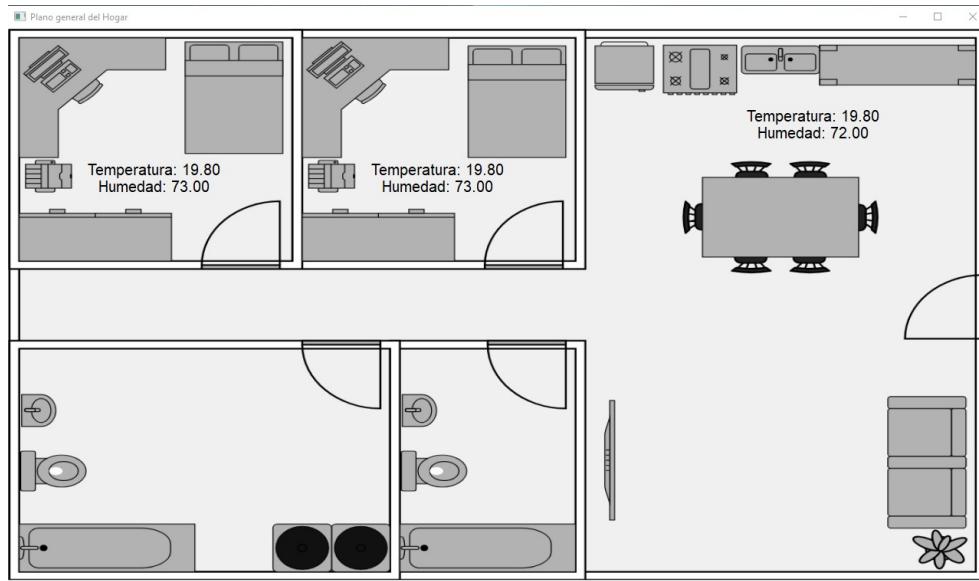


Fig. 1: Plano del Hogar.

El diseño en *Designer* consta de la simulación de una casa real con un plano diseñado en [draw.io](https://draw.io) el cual cuenta con 2 piezas, 2 baños y un living comedor cocina.

Para imprimir los valores de los sensores en las distintas habitaciones, se usaron labels (5 en específico) para todas las ubicaciones de la casa. Este diseño se guardó en formato .ui para posteriormente a través de **Anaconda** transformarlo a .py y correrlo en la Raspberry Pi.

En la Raspi, utilizando la biblioteca **Flask** la cual permite la captura y manejo de datos enviados desde formularios web y utilizando el método HTTP, se capturaron los datos de los distintos sensores conectados a la ESP32 para así poder procesarlos y mostrarlos en la interfaz gráfica.

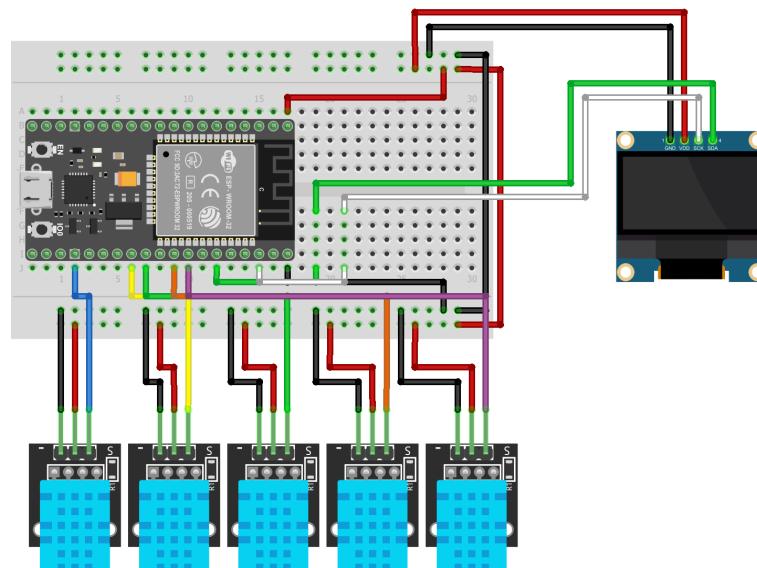
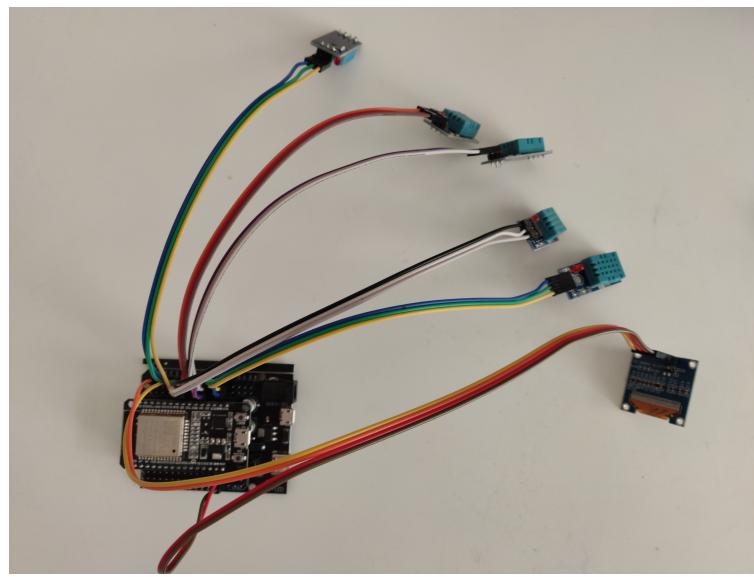


Fig. 2: Diagrama de conexión de los sensores DHT11 a la ESP32.

Adicional a la interfaz gráfica, se instaló una pantalla OLED 128x64 I2C Monochrome la cual permite monitorear los datos de temperatura y humedad de los sensores de manera directa.



**Fig. 3:** Pantalla OLED que muestra los datos de temperatura y humedad.



**Fig. 4:** Conexión Física de los sensores a la ESP.

Para poder consultar los datos de temperatura y humedad cuando no estas en tu hogar, se creó un Bot de Telegram con el cual a través de comandos simples puedes consultar el estado de tu vivienda.

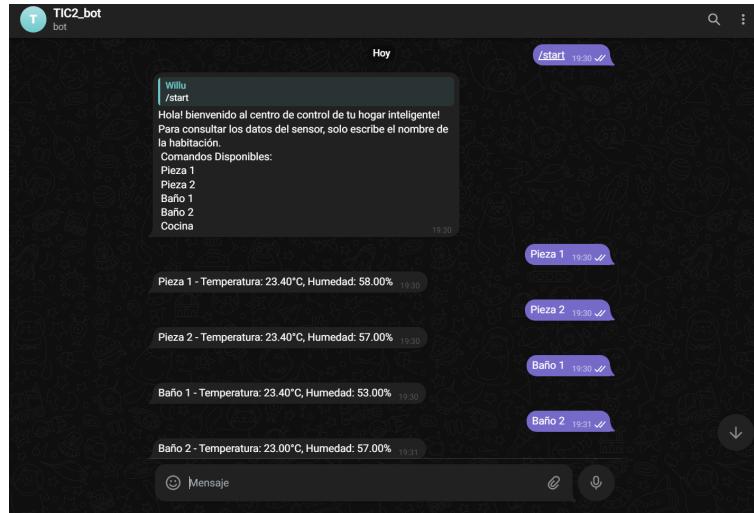


Fig. 5: Chat con el Bot de Telegram.

Para la implementación del Bot de Telegram en el sistema, se modificó el programa de la Raspi para que, al momento de ingresar un mensaje, este muestre la temperatura y humedad de la habitación seleccionada.

Para poder ejecutar los 2 códigos en simultaneo (Extraer los datos HTTP y el Bot de Telegram) se utilizaron 2 hilos (Thread). Esto se aplicó porque al momento de ejecutar el código solo se iniciaba el Bot de Telegram pero no se ejecutaba la interfaz gráfica.

### 2.2.1. Luces

Para la conexión de las luces se utilizó otro ESP32 que actúa como intermediario entre el sistema de Alexa y el sistema de iluminación. Este microcontrolador maneja las órdenes enviadas desde la Alexa y las transmite a las luces, permitiendo tanto el control manual como el control por voz a través de Amazon Alexa. La integración con Alexa añade una capa adicional de comodidad, permitiendo a los usuarios controlar las luces mediante comandos de voz.

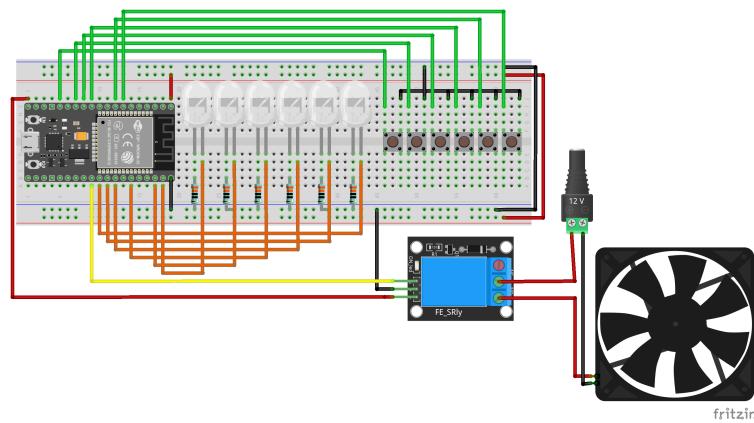


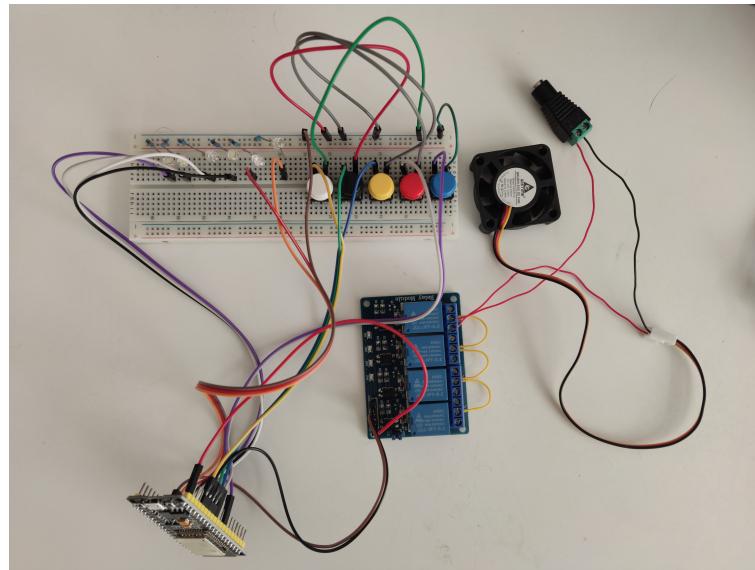
Fig. 6: Diagrama de conexión de las luces LED a la ESP32.

Como se observa en la figura 6, el circuito contempla botones con los cuales puedes encender y apagar los distintos dispositivos de manera manual sin perjudicar la conexión wifi con Alexa. Caso contrario ocurre con las ampolletas convencionales, las cuales al apretar el interruptor, cortan el suministro eléctrico desconectando el dispositivo del wifi, lo que impide que se vuelva a encender con

comandos de voz.

Para conectar la ESP32 con Alexa se utiliza una biblioteca llamada **Espalexa**, la cual, mediante wifi, crea dispositivos tipo On/Off con nombre personalizado. Alexa detecta estos dispositivos y los agrega a su sistema. Para el correcto funcionamiento del sistema, todos los dispositivos tienen que estar conectados a la misma red Wi-Fi.

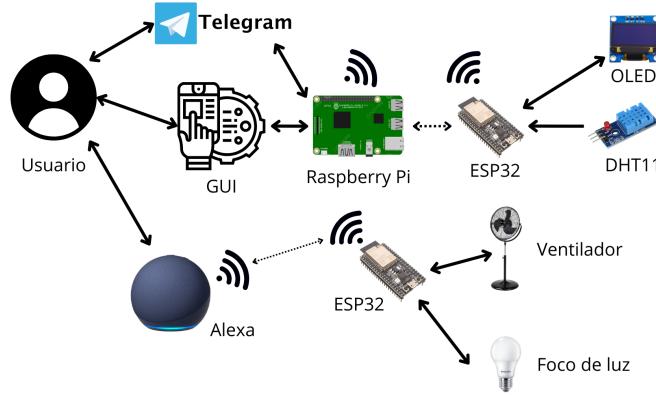
Como adicional, para demostrar la versatilidad del sistema, se agregó un módulo relé para poder conectar otros dispositivos (en este caso, un ventilador de 12 V) el cual sigue la misma lógica de las luces.



**Fig. 7:** Conexión Física de los LEDs a la ESP.

### 2.2.2. Diagrama

El siguiente diagrama representa la conexión del sistema, donde se puede ver claramente cómo los sensores de temperatura están conectados al ESP32, que a su vez se comunica con la Raspberry Pi. También se muestra cómo el segundo ESP32 gestiona la comunicación entre la Alexa y las luces para el control por voz. Este diseño modular no solo facilita la implementación y el mantenimiento del sistema, sino que también permite futuras expansiones y mejoras, haciendo de 'Home Connected' una solución flexible y adaptable a las necesidades cambiantes del hogar inteligente.



**Fig. 8:** Diagrama funcional del proyecto.

### 2.2.3. Bibliotecas y códigos

Para cada ESP32 se utilizaron distintas bibliotecas para cumplir con el correcto funcionamiento de nuestro programa, para la conexión entre la ESP32, la raspberry a través de internet, se utilizó

1. Biblioteca [Flask](#): Flask nos permite crear de fácil manera un framework en el cual poder trabajar independiente de la complejidad del programa
2. Biblioteca [HTTPClient](#): HTTPClient nos permite poder hacer el envío de datos a través de redes WI-FI
3. Biblioteca [Telebot](#): Telebot nos permite programar nuestro Bot en Telegram.

### 2.3. Resultados

En la figura 9 se observa la interfaz gráfica de la Raspi que muestra en tiempo real la temperatura de las distintas zonas del hogar.

En la figura 10 se muestran los dispositivos que detecta Alexa y que puede controlar con la función On/Off.

En un principio se propuso utilizar MQTT para la transmisión de datos de los sensores y los LEDs, pero se optó por HTTP y Espalexia por la facilidad de implementación. Además, se integró el Bot de Telegram para poder visualizar los datos de los sensores en cualquier lugar a través de teléfono (Fig. 5).

En cuanto a dificultades, en un principio se intentó automatizar el encendido del ventilador a base de la temperatura de una habitación, se buscaba que, cuando la temperatura supera un umbral ( $27^{\circ}\text{C}$ ) automáticamente se encendiera el ventilador, pero además poder controlar el ventilador manualmente. Finalmente, esto no se logró, ya que el umbral tomaba control por sobre lo manual y no se logró configurar adecuadamente. Es por esto que se optó por agregar el control del ventilador solamente por comandos de voz con Alexa. Como dificultad adicional, se tuvo que soldar LEDs y resistencias para la implementación de las luces.

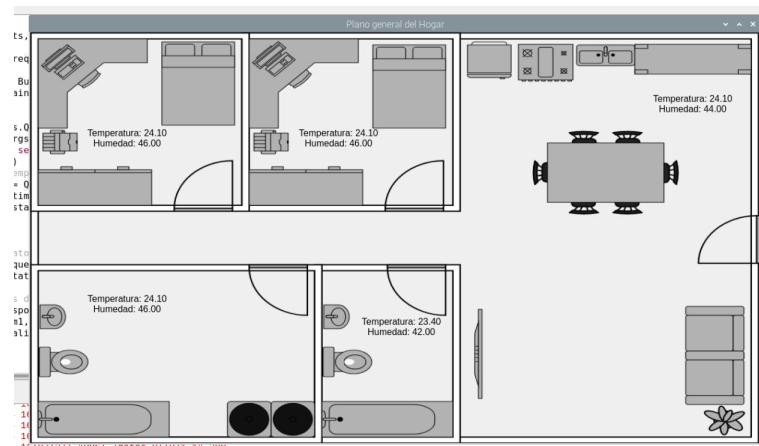


Fig. 9: Interfaz gráfica funcional.

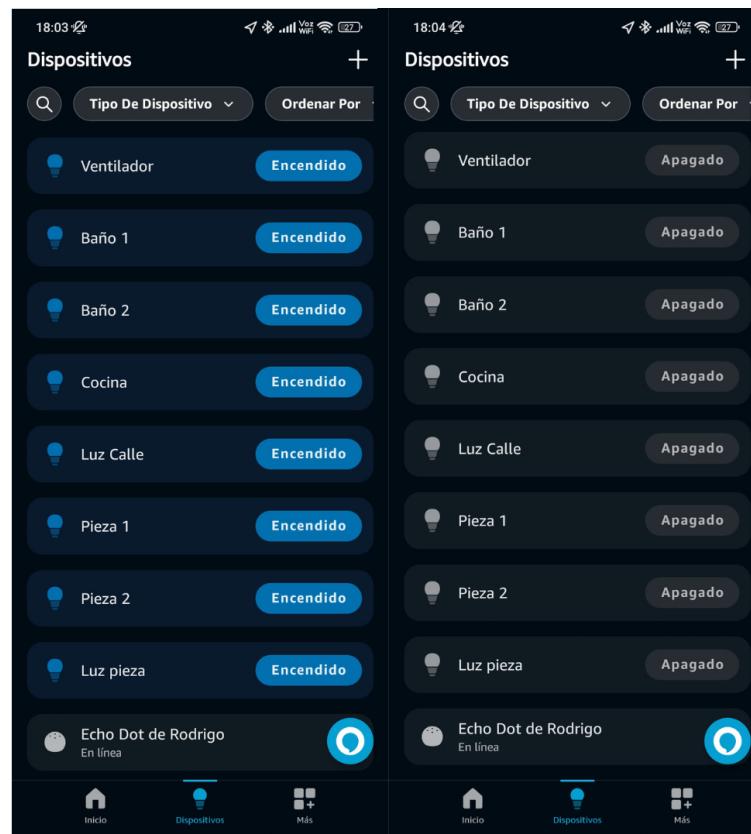


Fig. 10: Dispositivos conectados a Alexa.

Para añadir realismo al proyecto, se buscó armar una maqueta a escala de una casa para implementar todo el sistema, pero por falta de tiempo y presupuesto del equipo finalmente se optó por dejar el sistema conectado al aire, así se logran ver las conexiones en grupo y se entiende de mejor forma cómo está conectado el sistema.

### 3. Conclusión

El proyecto 'Home Connected' ha demostrado ser una solución eficiente y versátil para la gestión de un hogar inteligente. La implementación de sensores de temperatura y humedad DHT11, junto con microcontroladores ESP32 y una Raspberry Pi 3B+, ha permitido desarrollar un sistema capaz de monitorear y controlar diversos aspectos del hogar de manera inalámbrica y en tiempo real. La integración de Amazon Alexa ha añadido una capa adicional de comodidad, permitiendo el control de las luces mediante comandos de voz, lo cual mejora significativamente la experiencia del usuario.

La interfaz gráfica desarrollada facilita la visualización de la temperatura en diferentes secciones del hogar, presentando la información de manera clara y accesible. La elección de componentes y bibliotecas, como Flask y HTTPClient, ha sido crucial para garantizar una comunicación eficiente entre los dispositivos y la correcta funcionalidad del sistema.

El diseño modular del sistema no solo simplifica su implementación y mantenimiento, sino que también permite futuras expansiones y mejoras. Esto hace que 'Home Connected' sea una solución flexible y adaptable a las necesidades cambiantes del hogar inteligente, abriendo la posibilidad de integrar nuevos sensores y dispositivos en el futuro.

## Referencias

- [1] Material de la clase. Caro. V.