

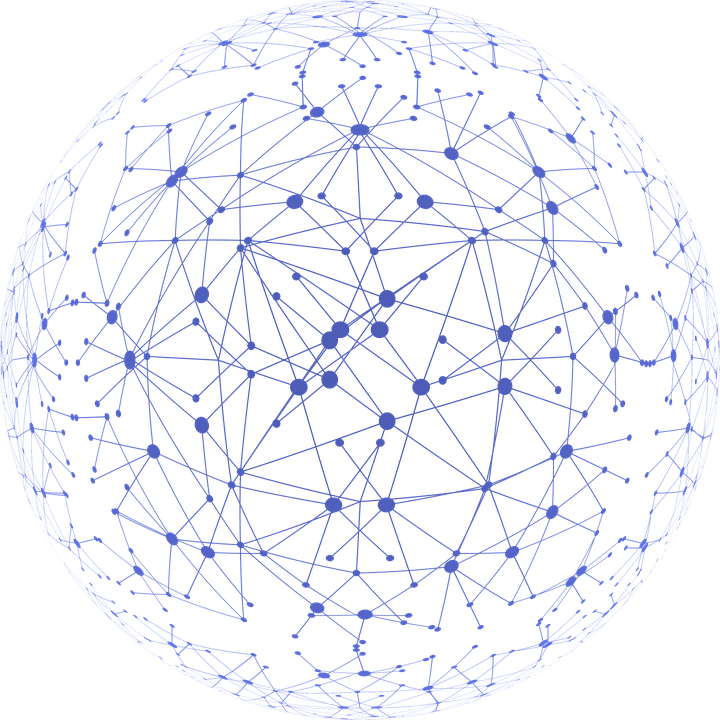
**Personal**

Asistente

Facultad Estadística e Informatica

Tecnologías Web

Jeffrey Romero Del Val



Contenido

[Planteamiento 3](#_Toc73214247)

[Plan de desarrollo económico 3](#_Toc73214248)

[Diagramas Funcionales 4](#_Toc73214249)

[Hardware Utilizado 4](#_Toc73214250)

[Diagrama De Despliegue 5](#_Toc73214251)

[Protocolos 6](#_Toc73214252)

[¿Qué es el protocolo MQTT? 6](#_Toc73214253)

[¿Qué es el protocolo TCP? 6](#_Toc73214254)

[Diagrama Esquemático 7](#_Toc73214255)

[Desarrollo de la aplicación 8](#_Toc73214256)

[Python 8](#_Toc73214257)

[Android Studio 8](#_Toc73214258)

[Docker 8](#_Toc73214259)

[Arduino 9](#_Toc73214260)

[Aspectos de conexión de la aplicación 9](#_Toc73214261)

[Creación del Broker MQTT en Docker 9](#_Toc73214262)

[Conexión Python con MQTT 9](#_Toc73214263)

[Conexión Android Con MQTT 10](#_Toc73214264)

[Conexión Arduino con MQTT 11](#_Toc73214265)

[Imagen del dispositivo final 12](#_Toc73214266)

[ESP32 con Sensor DHT11 12](#_Toc73214267)

[Aplicación de Android Corriendo en Dispositivo 13](#_Toc73214268)

[Programa de Python Corriendo 13](#_Toc73214269)

[Bibliografía 14](#_Toc73214270)

# Planteamiento

Sí en algún momento se le presento la idea de poder controlar de manera remota un ordenador entonces este proyecto es un acercamiento a esta idea, el asistente personal, siendo conocido mayormente por sus ejemplos mas famosos como lo son Cortana ,Google ,Siri o Alexa trabajan individualmente en dispositivos ya establecidos ,este proyecto no es mas que un acercamiento a esta nueva tecnología ,con la particularidad de que se tratara de controlar otros dispositivos de manera remota utilizando solo comandos de voz, debido al poco tiempo de que se empezó este proyecto solo se pudieron implementar algunas funcionalidades básicas ,sin embargo este tendrá un enfoque a largo plazo ,incrementando las funcionalidades y agregando mas interacciones con el usuario.

En cuanto a las funcionalidades actuales, posee en su repertorio reproducir videos de YouTube, conocer la temperatura del lugar encontrado, Hacer búsquedas en Google, recolectar información sobre una persona o acontecimiento en Wikipedia, establecer un nombre de usuario y preguntarle al asistente como se encuentra.

En resumen, este proyecto se enfoca en una asistente personal muy básica con fines de entretenimiento y educativo más que comercial, si esta llegara cubrir mas rasgos para la manipulación remota, esta si pudiera a llegar a ser de uso comercial ya sea para un control organizacional de equipos o datos empresariales.

# Plan de desarrollo económico

Por ahora el precio total para desarrollar este proyecto fue de $360 pesos mexicanos tomando en cuenta las siguientes partes:

1. ESP32($205.00)
2. Sensor de temperatura DHT11($155.00)

Todas las otras funciones del proyecto son parte del software, claro sin tomar en cuenta los dispositivos en los que se ejecuta como lo son el celular o la computadora.

Ahora, si en algún momento se tratara de expandir este proyecto, podría ser útil para muchas cuestiones, por ejemplo, podría implementarse más sensores de temperatura para monitorear ya sea la temperatura interna de la computadora o de diversos sitios, también se podría implementar diversas funcionalidades que pudieran ser controladas mediante comandos de voz, ya sea detección de gases, reproducir sonido por medio del esp32, etc.

# Diagramas Funcionales

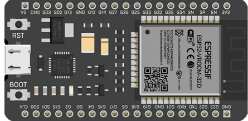
## Hardware Utilizado



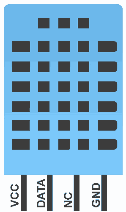
**Computadora:** Se necesita una computadora de cualquier tipo ya sea personal o laptop que le permita correr diversos programas (de preferencia que tenga Docker instalado con el bróker Mosquitto y Python).



**Dispositivo móvil Android:** Con suficiente espacio para instalar una aplicación (3.74 MB) y con Android 6.0 en adelante (No disponible en dispositivos IOS por el momento).



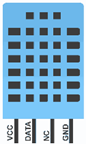
**Esp32:** Cualquier Esp32 que puedas encontrar siempre y cuando tenga un modulo de wifi, el usado en este proyecto es el ESP32 DevKit Module Wroom 32 de Espressif



**Sensor de Temperatura DHT11**: Cualquier sensor de temperatura de este modelo, usualmente vienen con cables Dupont Hembra a Hembra, pero en el caso de que no vengan puedes comprar el tipo que necesites de acuerdo con tu proyecto.

## Diagrama De Despliegue





**Computadora con Mosquitto Corriendo, el programa de Python siendo el receptor de mensajes y conectada a una red Wifi**

**Dispositivo Móvil Android conectado a una red Wifi**

**El Esp32 Debe estar conectado a la corriente y a la red Wifi**

**Conexión Por Pines**

**Mensajes al tópico Weather**

**Mensajes al tópico Commands**

**ESP32**

**Sensor de temperatura DHT11**

# Protocolos

Como observamos en el anterior grafico podemos notar que estamos usando una serie de herramientas de software para lograr la conexión entre los dispositivos, una de ellas siendo el programa **Docker** ,este programa es un contenedor que automatiza el despliegue de aplicaciones ,en resumen no ayudara a ejecutar ciertas aplicaciones que podremos usar para lograr el intercambio de mensajes entre dispositivos ,para ello se usara **Mosquitto** ,este es un bróker que nos ayudara a implementar el protocolo **MQTT.**

## ¿Qué es el protocolo MQTT?

*“MQTT o por sus siglas (Message Queue Telemetry Transport) es un protocolo de transporte de mensajes Cliente/Servidor basado en publicaciones y subscripciones a los denominados “tópicos”. Cada vez que un mensaje es publicado será recibido por el resto de los dispositivos adheridos a un tópico del protocolo. El protocolo MQTT funciona sobre TCP/IP o sobre otros protocolos de red con soporte bidireccional y sin pérdidas de datos.”*

(http://www.tst-sistemas.es/mqtt/, s.f.)

En nuestro proyecto estamos usando una conexión local por lo tanto usaremos la IP de nuestra computadora siendo esta la 192.168.1.2

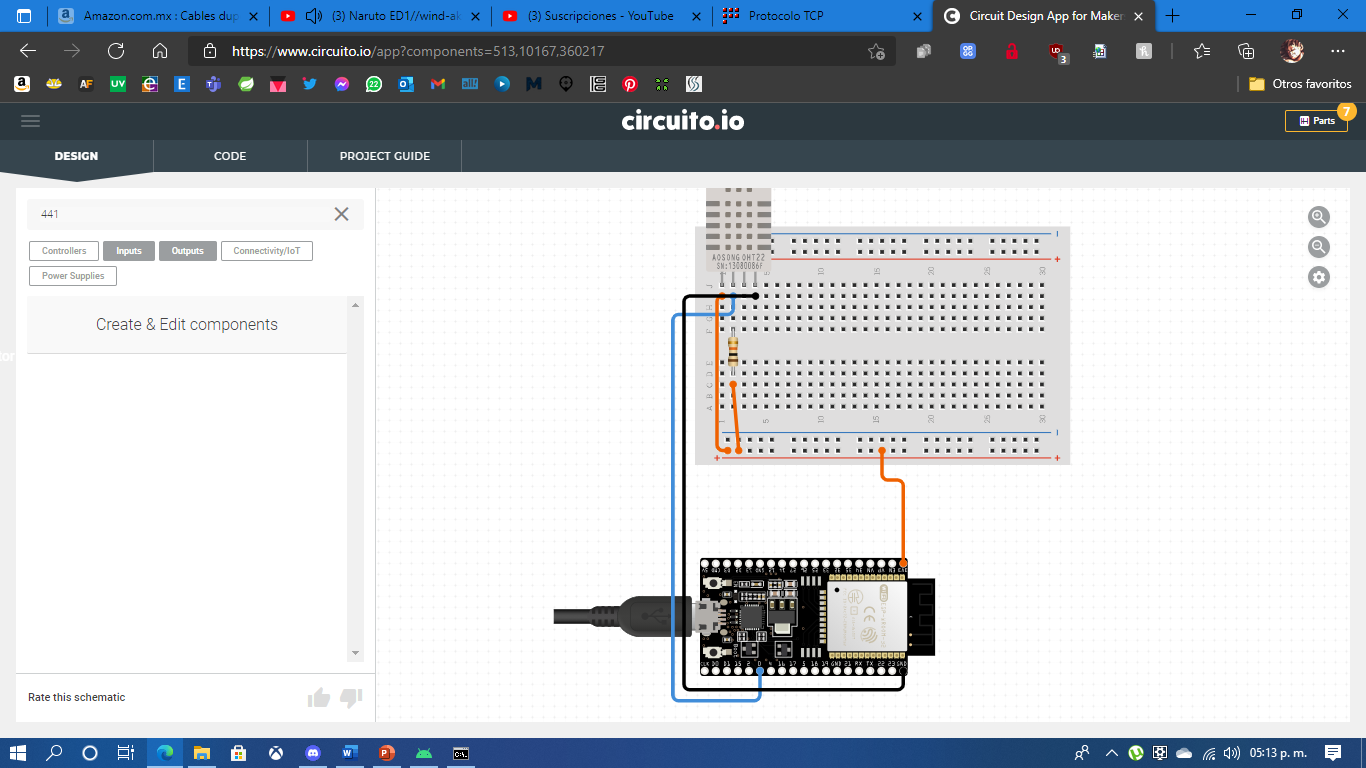
## ¿Qué es el protocolo TCP?

*“TCP o por sus siglas (Transmission Control Protocol)  es una forma segura de intercambio de datos, al requerir de la conexión y autorización entre el emisor y el receptor, o el cliente y el servidor, antes de producirse la transferencia; una vez ambas partes hayan autorizado la transmisión, podrá iniciarse el envío y recepción de datos.”* (InfoTecs, 2020)

Al momento de crear nuestro Docker con **mosquitto** tendremos que configurar los puertos que se utilizaran, en este caso configuraremos el **puerto 1883** el cómo se configura este puerto se verá más adelante.

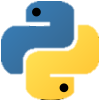
# Diagrama Esquemático

2 formas de conectar el Sensor de temperatura al ESP32 en este proyecto usaremos el izquierdo (Usando el GPIO15 para la conexión con DATA)



# Desarrollo de la aplicación

Una vez Explicado cómo se conectará el ESP32 tendremos que presentar las herramientas para el desarrollo de software que usaremos para este proyecto, es recomendable que se tenga un poco de conocimiento usando estas herramientas antes de empezar a programar el código.



## Python

“Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código.​” (WikiPedia, s.f.)

Python es lo que nos permitirá crear la mayoría de las funciones de nuestra asistente, así mismo tendremos que importar varias librerías, como lo son el Text to speech. En todo caso Python se encargará de crear toda la aplicación para que nuestra asistente haga su trabajo en nuestra computadora.

## Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y está basado en IntelliJ IDEA. (Android, s.f.)

Así como Python es lo que nos ayudara a crear la aplicación para nuestra computadora, Android Studio es lo que nos ayudara a crear nuestra aplicación para nuestro dispositivo móvil, en esencia es una aplicación muy sencilla que lo único que hará será tomar el audio de nuestro micrófono y convertirlo a texto gracias a una Api de Google.

## Docker

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos.​ (WikiPedia, s.f.)

Como se había mencionado antes, Docker será la principal Herramienta que nos permitirá crear nuestro bróker de mosquitto permitiéndonos así la comunicación entre nuestros dispositivos.

## Arduino

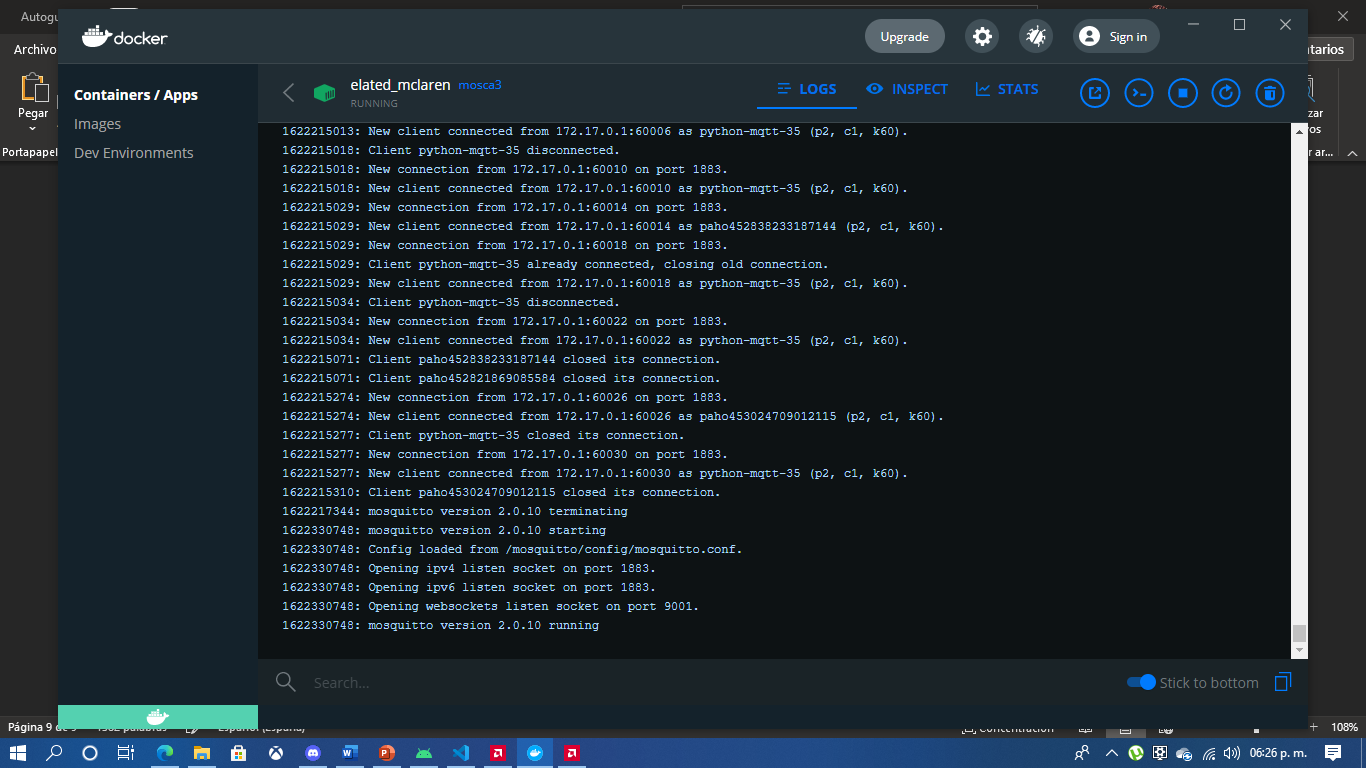
Arduino es una plataforma de desarrollo basada en una placa electrónica de hardware libre que incorpora un microcontrolador reprogramable y una serie de pines hembra.

En este caso mas que en al hardware que produce, nos enfocaremos en la herramienta de software que nos proporciona, esta nos ayudara a configurar nuestro ESP32 para que permita la lectura de datos de nuestro sensor de temperatura y además mandar esos datos por medio de nuestro bróker MQTT (Mosquitto).

# Aspectos de conexión de la aplicación

## Creación del Bróker MQTT en Docker

Una de las primeras cosas que debemos realizar en este proyecto es crear nuestro bróker de MQTT con Docker, para ello tendríamos que descargar la imagen de mosquitto, configurarla para que funcione con el puerto 1883 y correr esa imagen.



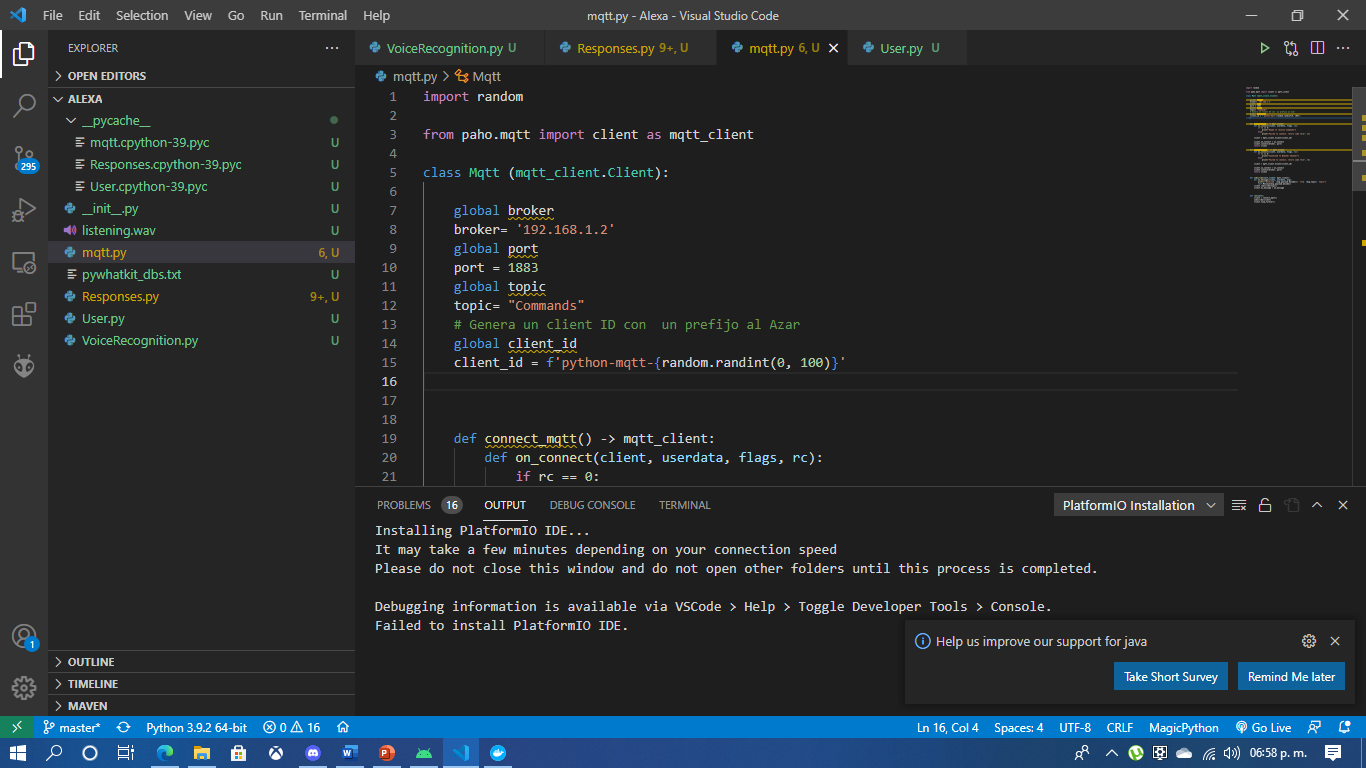
(Aplicación de Docker corriendo Mosquitto)

Podemos probar mandando mensajes entre suscriptor y publicador para asegurarnos que funcione adecuadamente (Usa el tópico que quieras).

Si por el momento tiene **dudas** de cómo realizar estos procesos refiérase **al video adjunto a este Documento** en él se explica más a detalle todo el procedimiento**.**

## Conexión Python con MQTT

Una vez que tengamos nuestro servidor tendremos que empezar a crear nuestra aplicación de Python, antes de agregarle cualquier funcionalidad de nuestra asistente, tenemos que configurar nuestro MQTT en Python para que haga la función de suscriptor de mensajes o de forma más simplificada, Python se encargara de recibir todos los mensajes en el tópico asignado. Para este proyecto configuraremos Python de la siguiente manera:



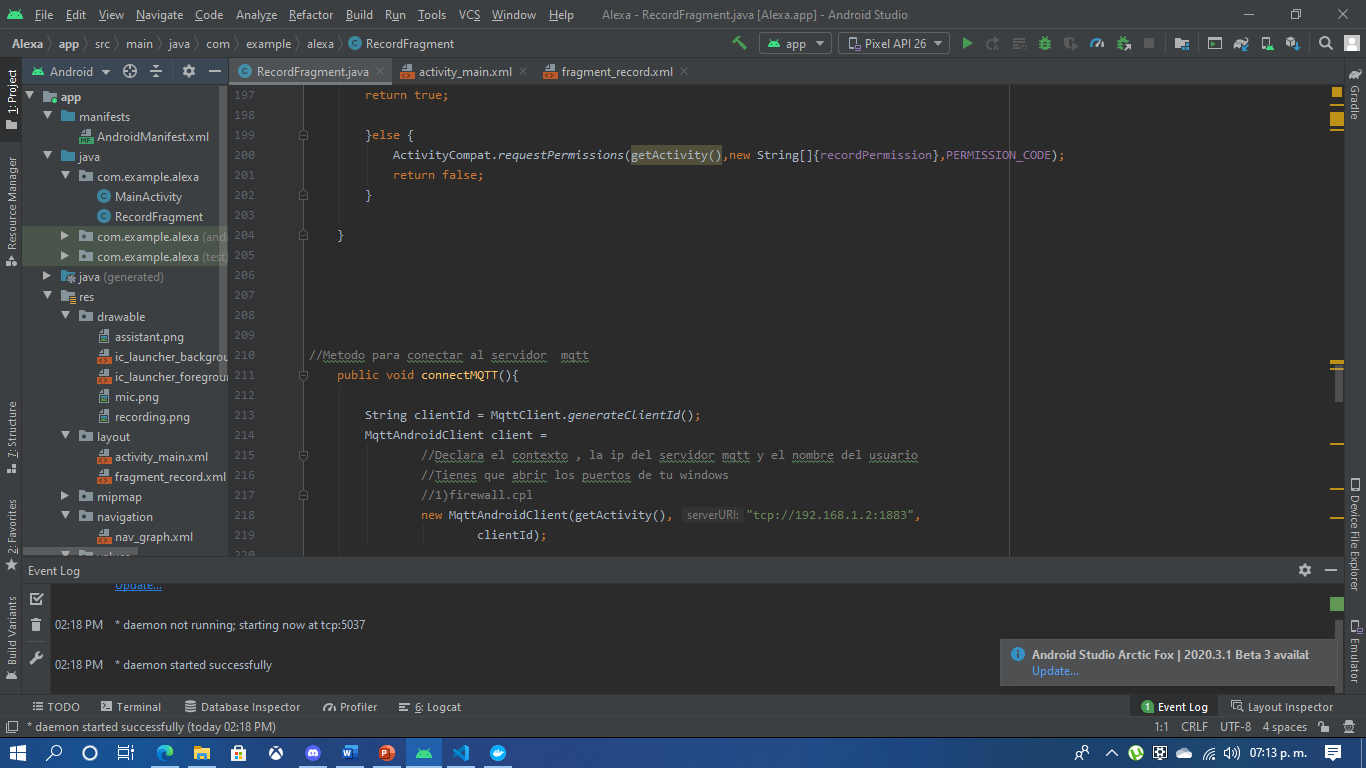
(Configuración de MQTT en Python)

Tenemos que configurar cual es la dirección de nuestro bróker, puesto que es una conexión local se usara la IP de nuestro ordenador, configuraremos también el puerto del TCP y el tópico al que queramos suscribirnos.

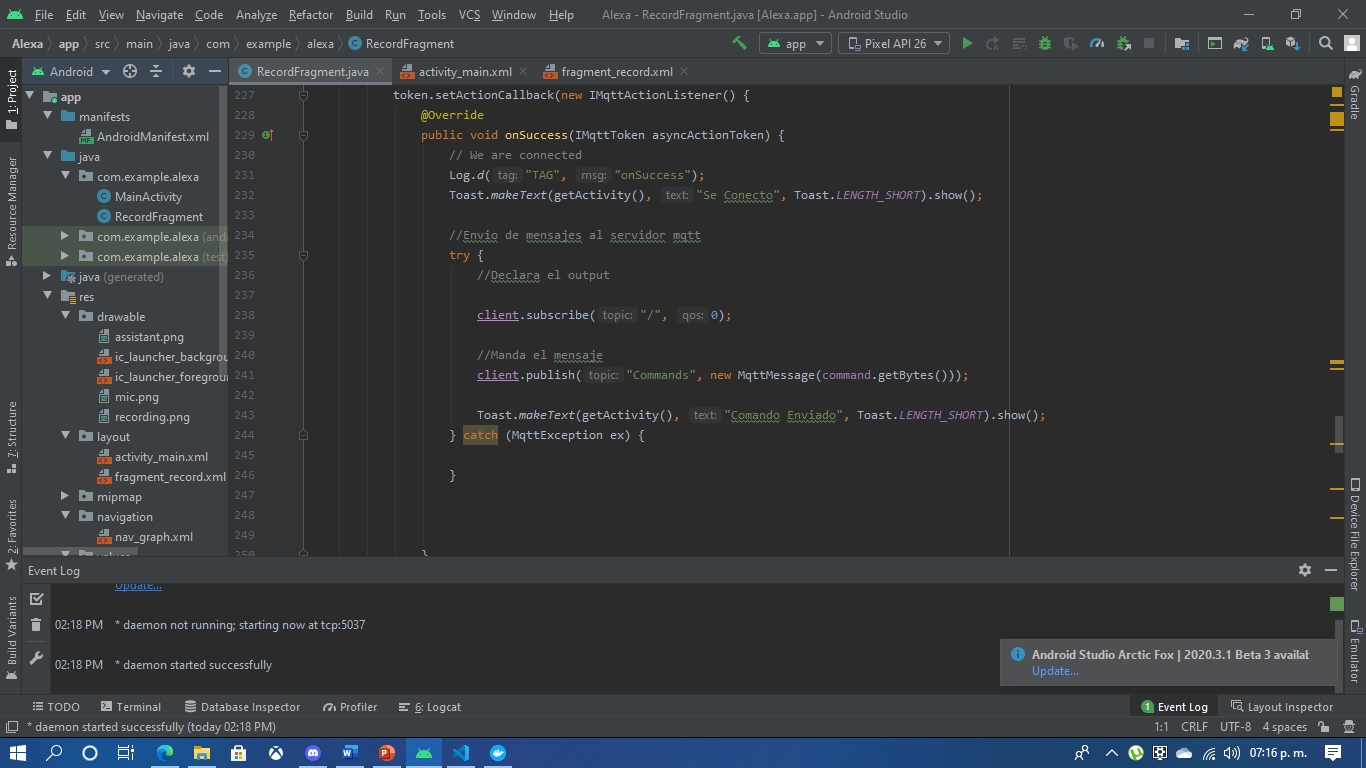
Con esta configuración podremos conectarnos a nuestro bróker de mosquitto que se encuentra en nuestra máquina (o en cualquier otra maquina remota) y mandar mensajes desde Python a nuestro Docker.

## Conexión Android Con MQTT

Al igual que en Python en Android Studio tenemos también que configurar nuestro MQTT para poder configurarlo agregando claro las librerías necesarias, para ello tenemos que realizar un configuración similar ,solo que en este caso la aplicación fungirá como un publicador, encargándose de ser la que mande los mensajes desde la aplicación a nuestro computador (Cabe destacar que para que haya una comunicación remota tienes que abrir los puertos que estés usando en tu ordenador ,en este caso el puerto 1883),entonces la configuración tendría que quedar similar a esta:



(Configuración de la conexión MQTT en Android)

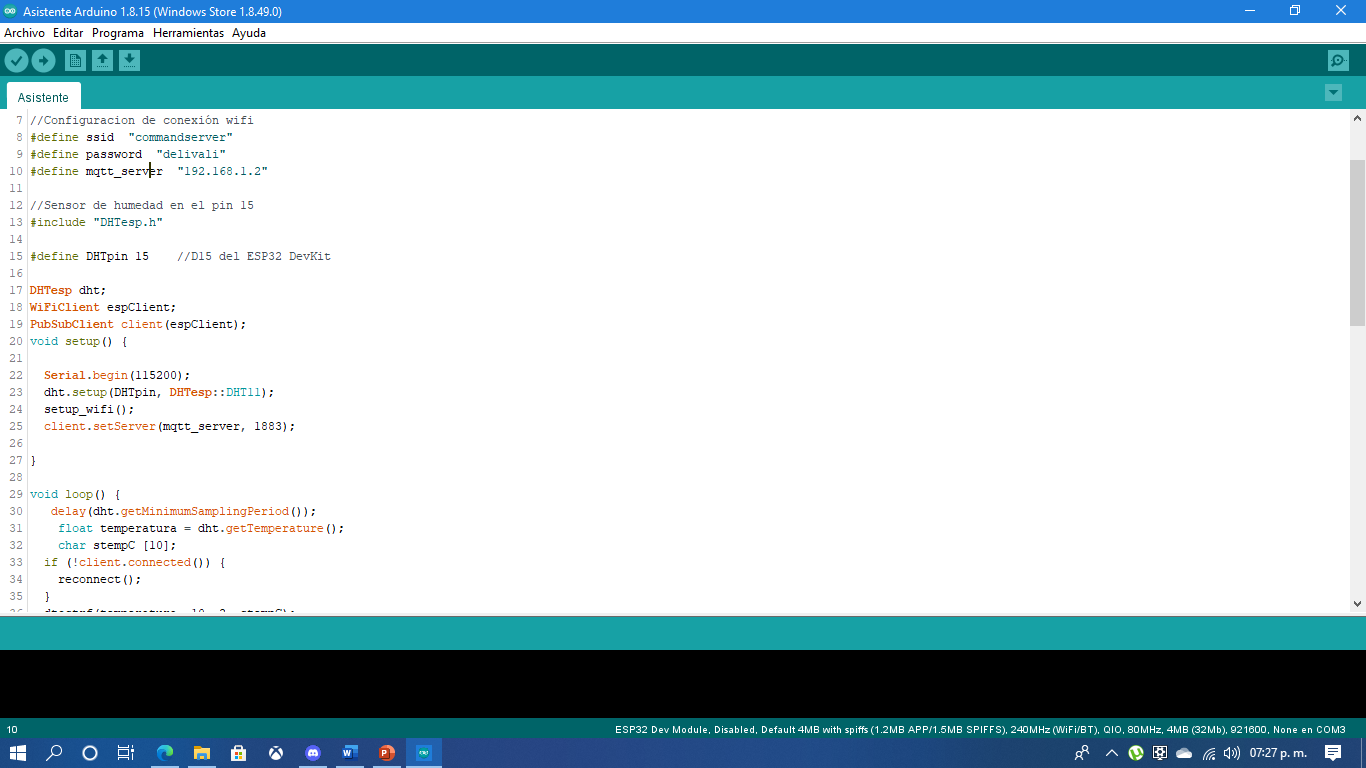


(Configuración del tópico a utilizar durante la conexión en Android)

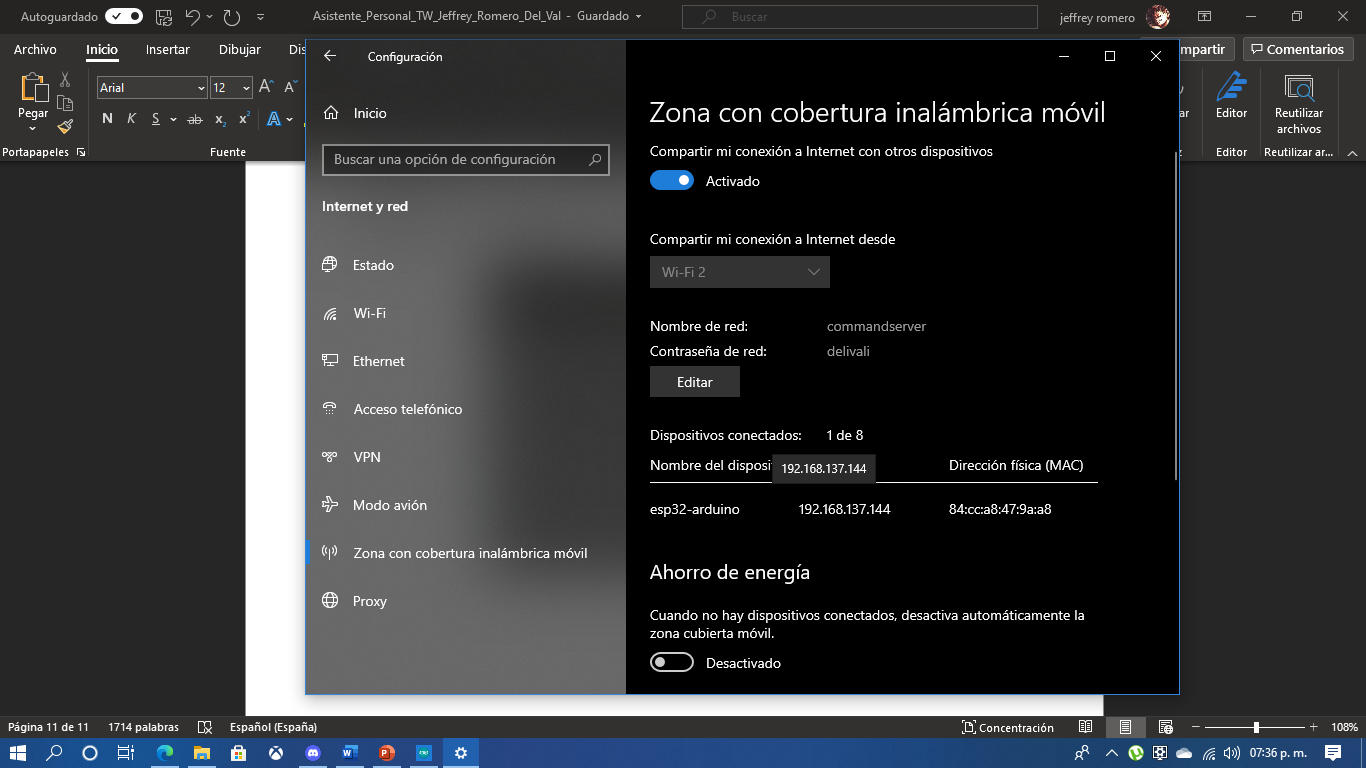
Similar a Python se tiene que configurar la dirección del bróker al que se este conectando, el puerto que se este utilizando y por ultimo el tópico que se utilizara.

## Conexión Arduino con MQTT

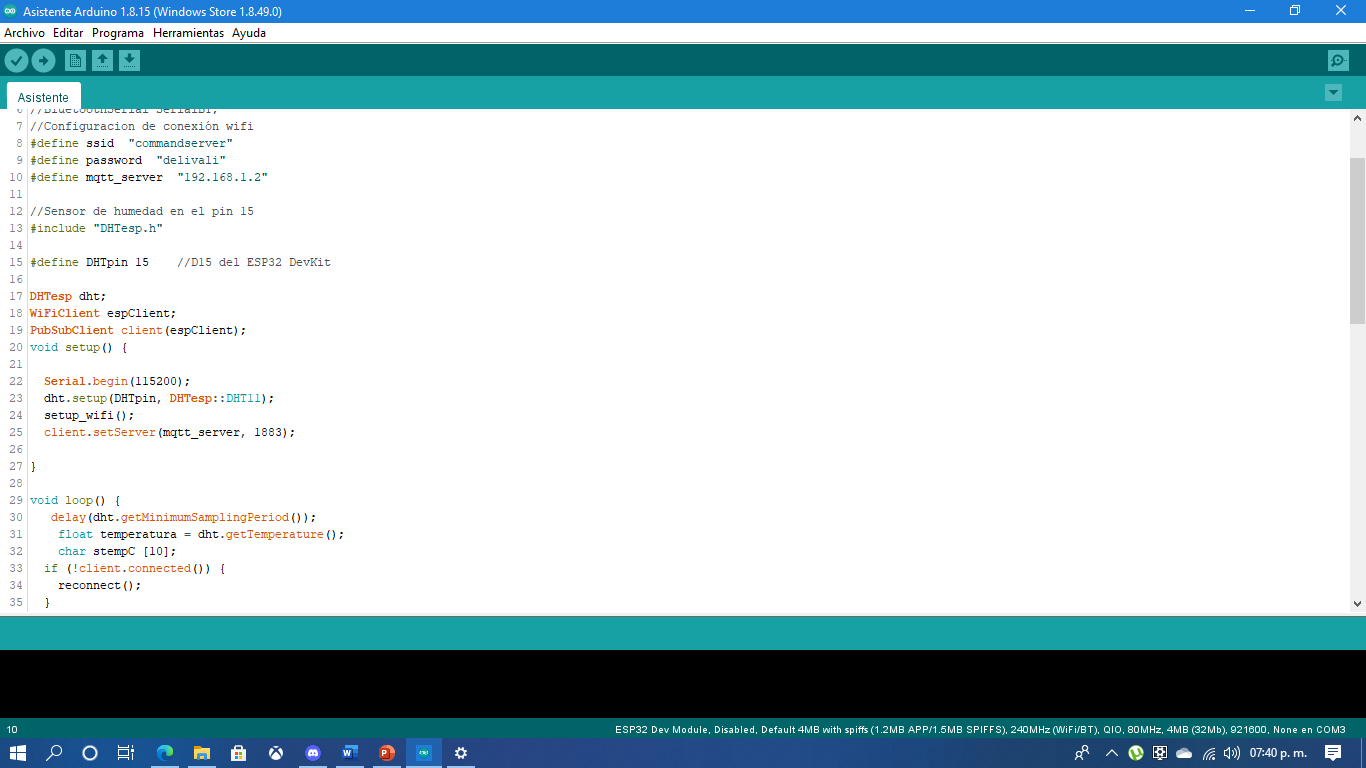
Ya que tengamos una conexión entre los 2 dispositivos principales, es hora de agregar también nuestra conexión a MQTT con nuestro ESP32 para ello configuraremos de igual manera la dirección del bróker a ejecutar, el tópico que usara y el puerto que está utilizando, además de eso tenemos que configurar una conexión wifi que se pueda usar en nuestro ordenador para compartir con nuestro ESP32. Empecemos primero con la Configuración WiFi de nuestro ESP32



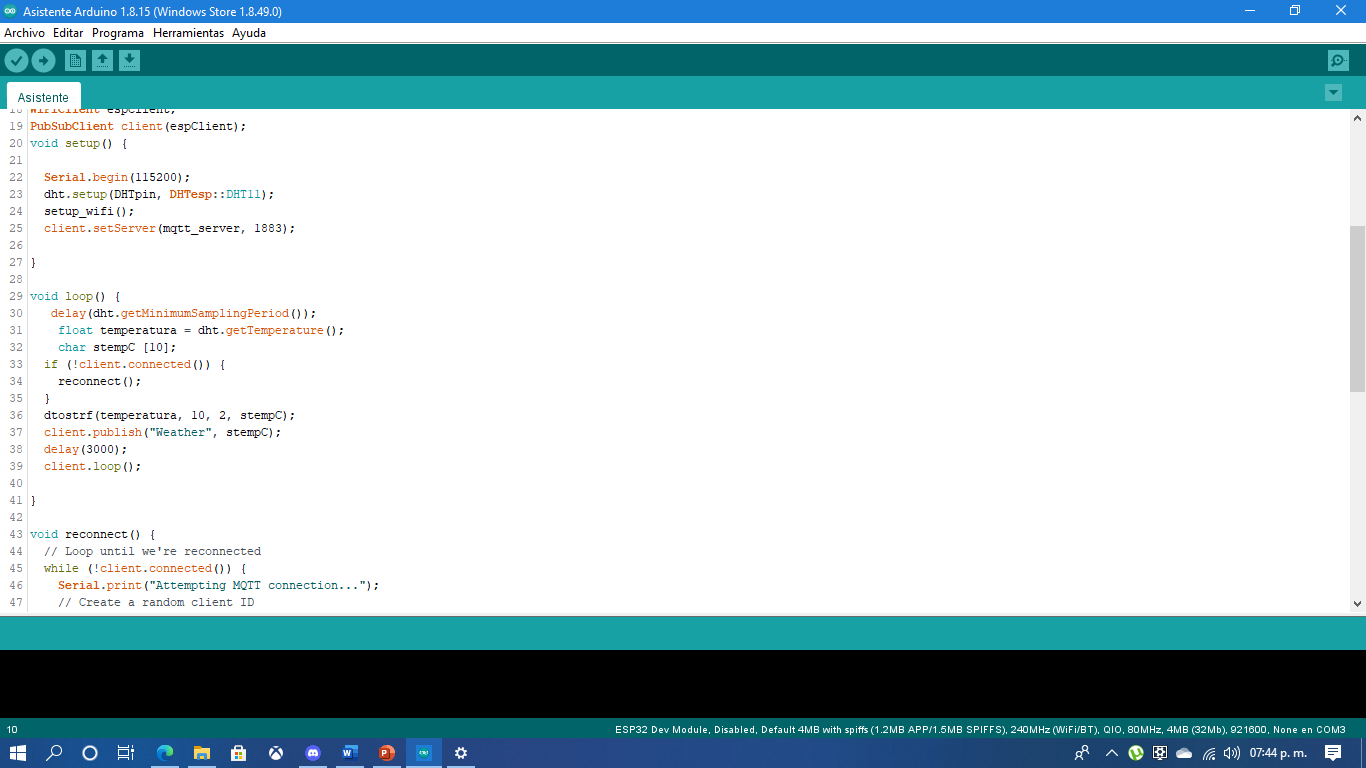
Esta seria nuestra configuración en el programa Arduino, usaremos estas variables para el procedimiento de conexión (En el video se encuentra el procedimiento completo para la implementación), en cuanto a la configuración en nuestro ordenador se vería algo así:



Una vez configurado nuestra conexión Wifi con el ESP32 podremos hacer la configuración del servidor MQTT, para ello debemos al igual que en las anteriores configurar el puerto, la dirección del bróker y su tópico:



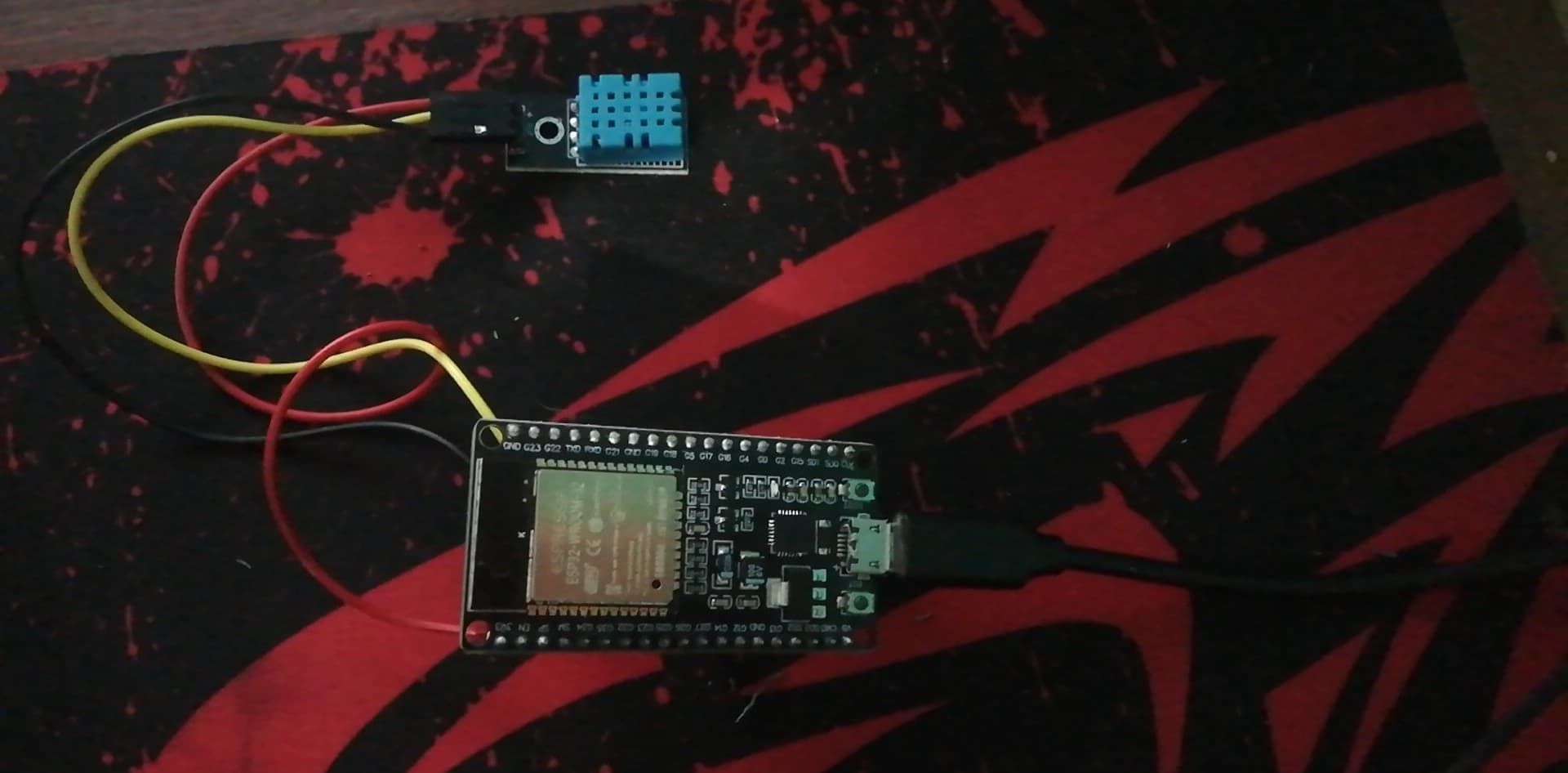
En este Caso la variable mqtt\_server equivale a la dirección 192.168.1.2 que es la dirección de nuestro bróker, y de ahí prosigue el puerto y por último para configurar el envió de mensajes le asignamos el siguiente tópico:



Como pueden notar aquí el tópico cambia siendo este el de ‘Weather’ y el anterior siendo ‘Commands’, debido a que el esp32 mandara consecutivamente la temperatura del cuarto por lo tanto no puede interferir con los mensajes que le mandaremos desde el móvil.Con cada MQTT en los 3 puntos podremos mandar mensajes y recibirlos respectivamente.

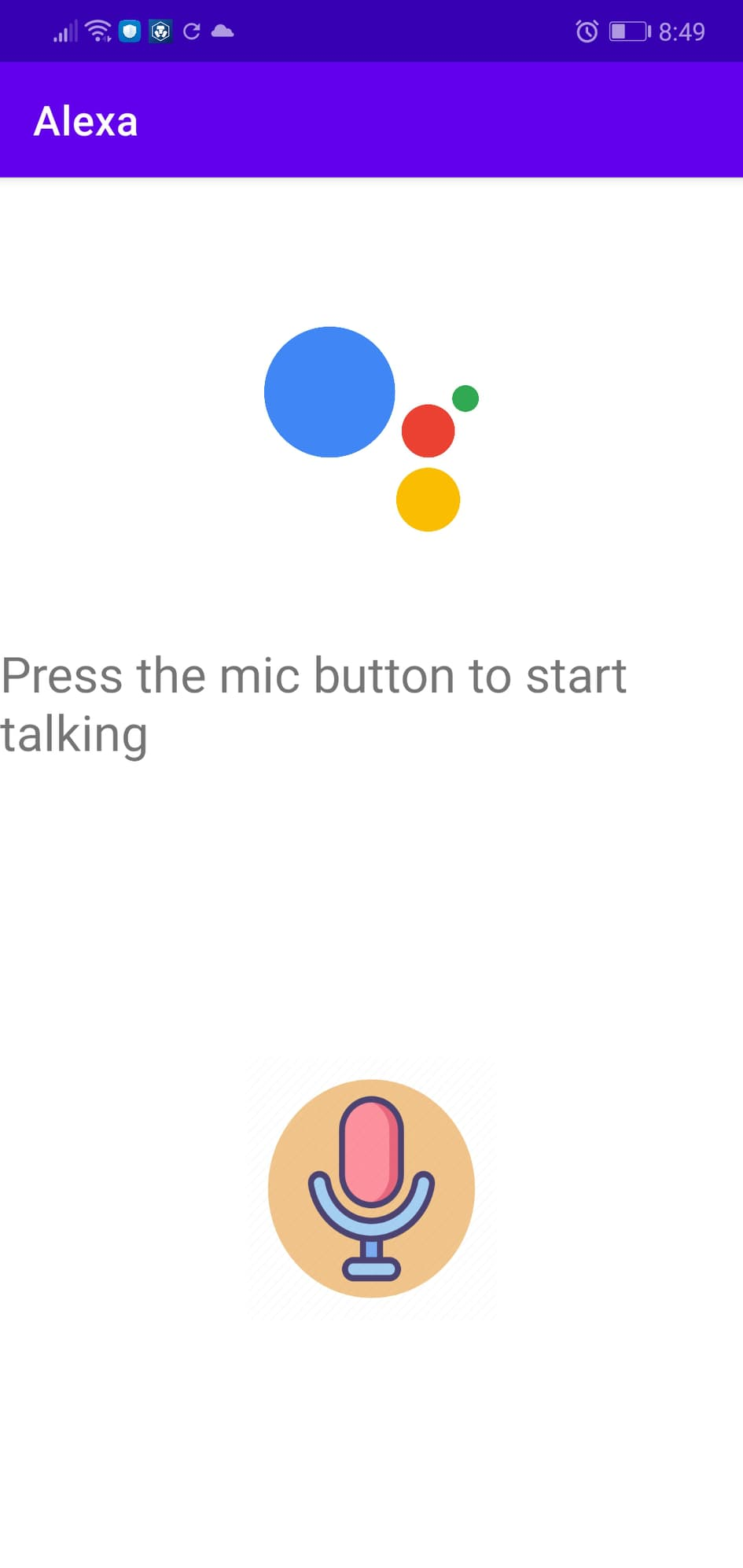
# Imagen del dispositivo final

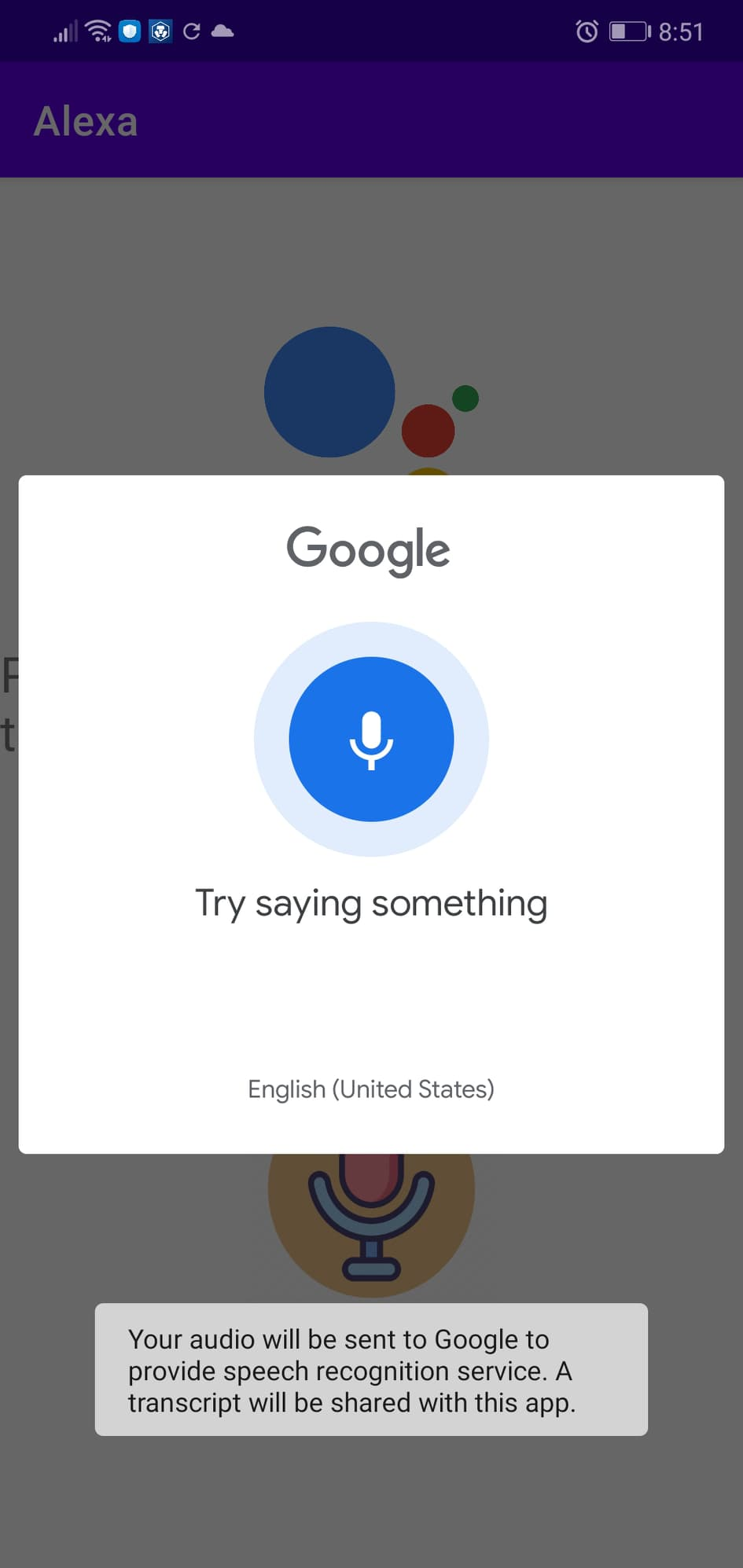
## ESP32 con Sensor DHT11



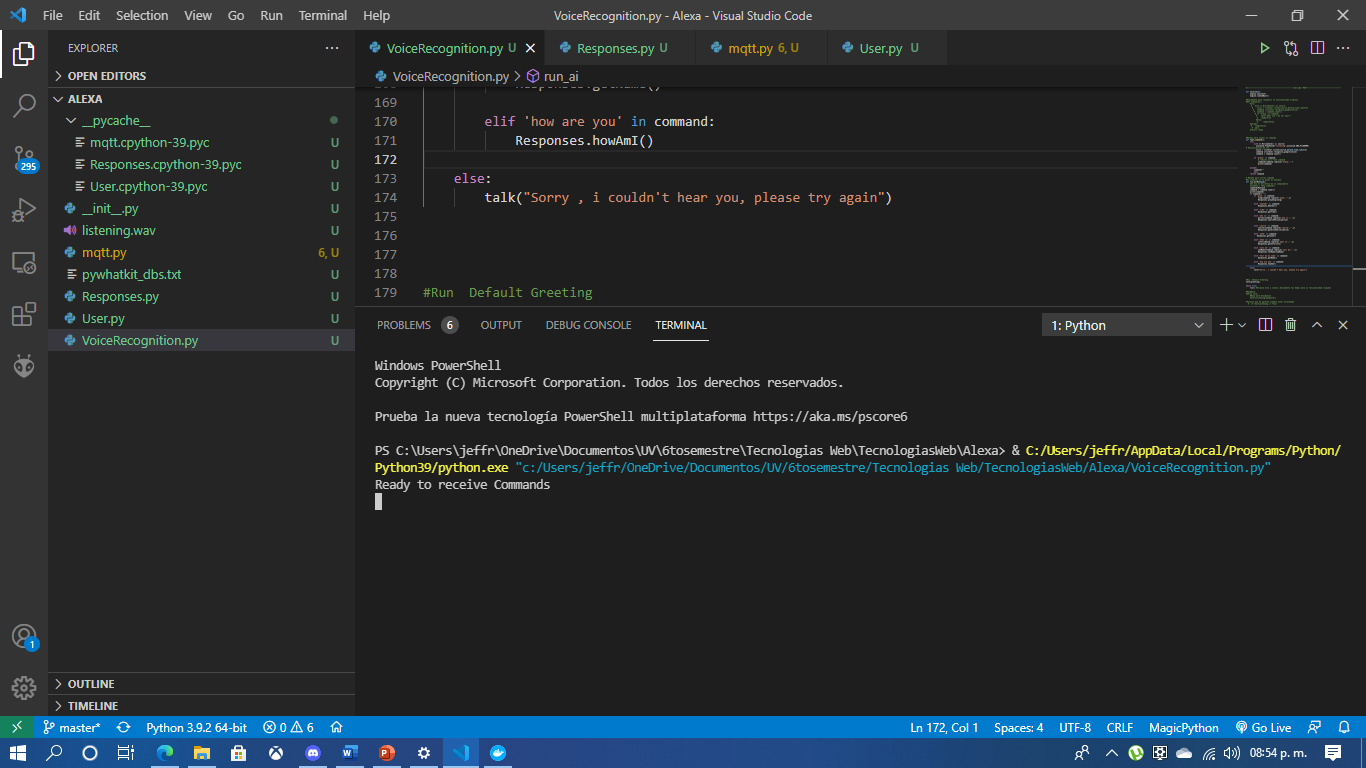


## Aplicación de Android Corriendo en Dispositivo





## Programa de Python Corriendo



# Bibliografía

Android, D. (s.f.). *Developer Android*. Obtenido de Developer Android: https://developer.android.com/studio/intro?hl=es-419

*http://www.tst-sistemas.es/mqtt/*. (s.f.). Obtenido de TST: http://www.tst-sistemas.es/mqtt/

InfoTecs. (05 de Junio de 2020). *InfoTecs*. Obtenido de InfoTecs: https://infotecs.mx/blog/protocolo-tcp.html

WikiPedia. (s.f.). *WikiPedia*. Obtenido de WikiPedia: https://es.wikipedia.org/wiki/Docker\_(software)