**UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS**

(Universidad del Perú, DECANA DE AMÉRICA)

**FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA**

ESCUELA PROFESIONAL DE INGENIERÍA DE SOFTWARE



**PROYECTO:**

SISTEMA DE ALARMA DE SEGURIDAD EN EL HOGAR   
CON ESP32, SENSORES PIR Y NOTIFICACIONES (MENSAJES)   
POR WHATSAPP

**CURSO:**

INTERNET DE LAS COSAS

**DOCENTE:**

JOSE ALFREDO HERRERA QUISPE

**PRESENTADO POR:**

VALLE ATENCIO, LUIS ALFONSO

Ciudad Universitaria, Lima - Perú

**2024**

Contenido

[1. **Introducción** 4](#_Toc172495602)

[o Objetivos del proyecto 4](#_Toc172495603)

[o Contexto y justificación 4](#_Toc172495604)

[2. **Importancia** 4](#_Toc172495605)

[a. Seguridad mejorada 4](#_Toc172495606)

[b. Automatización y eficiencia 4](#_Toc172495607)

[c. Integración IoT 4](#_Toc172495608)

[d. Aplicación práctica de sensores 4](#_Toc172495609)

[e. Notificación instantánea 5](#_Toc172495610)

[f. Escalabilidad y adaptabilidad 5](#_Toc172495611)

[3. **Tecnologías** 5](#_Toc172495612)

[o ESP32 5](#_Toc172495613)

[o Sensores PIR (Passive Infrared) 5](#_Toc172495614)

[o WhatsApp API 5](#_Toc172495615)

[o Protocolos de Comunicación (HTTP) 5](#_Toc172495616)

[o Plataforma de Nube 5](#_Toc172495617)

[o Desarrollo de Software 5](#_Toc172495618)

[**4.** **Requerimientos** 5](#_Toc172495619)

[o Hardware 6](#_Toc172495620)

[o Software 6](#_Toc172495621)

[o Servicios en la Nube 6](#_Toc172495622)

[5. **Arquitectura de Software Utilizada** 6](#_Toc172495623)

[o Modelo Cliente-Servidor con API Restful 6](#_Toc172495624)

[o Capa de Aplicación IoT 6](#_Toc172495625)

[o Protocolos de Comunicación 6](#_Toc172495626)

[o Componentes Principales de la Arquitectura 7](#_Toc172495627)

[6. **Patrón de Diseño** 7](#_Toc172495628)

[o Patrón de Diseño Cliente-Servidor 7](#_Toc172495629)

[o Características del Patrón Cliente-Servidor en el Código 7](#_Toc172495630)

[o Beneficios del Patrón Cliente-Servidor en este Contexto 8](#_Toc172495631)

[7. **Flujo de Funcionamiento** 8](#_Toc172495632)

[o Detección de Movimiento 8](#_Toc172495633)

[o Envío de Datos 8](#_Toc172495634)

[o Procesamiento en la Nube 8](#_Toc172495635)

[o Notificación por WhatsApp 8](#_Toc172495636)

[o Interacción Usuario 8](#_Toc172495637)

[8. **Desarrollo del Proyecto** 8](#_Toc172495638)

[o Metodología utilizada 8](#_Toc172495639)

[o Actividades realizadas 13](#_Toc172495640)

[ API WhatsApp (Configuración) 13](#_Toc172495641)

[ Instalamos la Librería de ESP32 en Arduino 28](#_Toc172495642)

[ Instalando Driver del ESP32 para Windows 37](#_Toc172495643)

[ Seleccionamos el ESP32 en el IDE Arduino. 43](#_Toc172495644)

[ Escribimos el código que va ejecutar el sistema. 45](#_Toc172495645)

[ Ejecutamos el código desarrollado. 49](#_Toc172495646)

[ Recepción de mensajes en nuestro número de WhatsApp. 50](#_Toc172495647)

[9. **Resultados** 51](#_Toc172495648)

[o Principales hallazgos y datos obtenidos 51](#_Toc172495649)

[10. **Conclusiones** 52](#_Toc172495650)

[o Eficiencia en la Detección de Movimiento 52](#_Toc172495651)

[o Integración Exitosa de Tecnologías 52](#_Toc172495652)

[o Notificaciones Instantáneas por WhatsApp 52](#_Toc172495653)

[o Seguridad y Privacidad 52](#_Toc172495654)

[o Potencial de Escalabilidad y Adaptabilidad 52](#_Toc172495655)

[o Aplicación de IoT 52](#_Toc172495656)

[11. **Referencias** 53](#_Toc172495657)

[o Fuentes consultadas 53](#_Toc172495658)

1. **Introducción**

## Objetivos del proyecto

El objetivo principal de este proyecto es desarrollar un sistema de alarma utilizando tecnología de Internet de las Cosas (IoT) para detectar movimiento mediante sensores PIR (Passive Infrared Sensor) y enviar notificaciones en tiempo real a través de WhatsApp. Los objetivos específicos incluyen:

* Implementar un sistema de detección de movimiento utilizando tres sensores PIR conectados a una placa ESP32.
* Configurar la placa ESP32 para que envíe mensajes de texto a través de WhatsApp cuando se detecte movimiento.
* Diseñar una interfaz de usuario sencilla para monitorear y controlar el sistema de alarma.

## Contexto y justificación

El Internet de las Cosas (IoT) ha revolucionado la forma en que interactuamos con nuestros entornos, permitiendo la conexión y control remoto de dispositivos cotidianos. En este contexto, el desarrollo de sistemas de seguridad como el propuesto puede mejorar la eficiencia y la respuesta ante situaciones de riesgo en hogares y pequeños negocios. La capacidad de recibir notificaciones instantáneas a través de plataformas ampliamente utilizadas como WhatsApp ofrece una ventaja significativa en la capacidad de respuesta y la accesibilidad del usuario final.

1. **Importancia**

## Seguridad mejorada

Proporciona una capa adicional de seguridad al detectar movimientos no deseados en áreas específicas de la maqueta (simulación real). Esto es crucial para proteger propiedades y bienes.

## Automatización y eficiencia

Automatiza el proceso de detección y notificación, lo que permite una respuesta rápida y eficiente ante eventos de seguridad sin necesidad de supervisión constante.

## Integración IoT

Demuestra la capacidad de integrar dispositivos IoT (Internet of Things) para crear soluciones prácticas y útiles en entornos reales, promoviendo el uso innovador de la tecnología.

## Aplicación práctica de sensores

Utiliza sensores PIR, que son eficaces para detectar movimiento humano, mostrando cómo tecnologías simples pueden aplicarse de manera efectiva para resolver problemas cotidianos.

## Notificación instantánea

Al utilizar WhatsApp para enviar alertas, garantiza que los usuarios reciban notificaciones casi inmediatas, lo que es crucial en situaciones de emergencia o seguridad.

## Escalabilidad y adaptabilidad

El diseño del sistema permite escalar el número de sensores y áreas monitoreadas según sea necesario, mostrando flexibilidad y adaptabilidad a diferentes contextos y tamaños de espacio.

1. **Tecnologías**

## ESP32

Es una placa de desarrollo ampliamente utilizada en proyectos de IoT debido a su potencia de procesamiento, conectividad Wi-Fi y Bluetooth integrada, así como su capacidad para manejar entradas y salidas digitales y analógicas. Permite la comunicación con los sensores PIR y la transmisión de datos a través de Internet.

## Sensores PIR (Passive Infrared)

Son sensores de movimiento que detectan cambios en el nivel de radiación infrarroja en su campo de visión. Son ideales para aplicaciones de seguridad, ya que pueden identificar movimientos humanos dentro de un área específica sin necesidad de emitir señales activas.

## WhatsApp API

Para enviar mensajes por WhatsApp, el proyecto necesita utilizar la API de WhatsApp Business. Esta API permite la integración de WhatsApp con aplicaciones externas, como el sistema de alarma basado en ESP32 y sensores PIR. Es necesario configurar y autenticar adecuadamente la API para enviar mensajes desde el sistema.

## Protocolos de Comunicación (HTTP)

Para la comunicación entre la placa ESP32 y el servidor que maneja la API de WhatsApp, se utilizó el HTTP (Hypertext Transfer Protocol). Sin embargo, hay protocolos como MQTT (Message Queuing Telemetry Transport) que es el ideal para aplicaciones IoT debido a su bajo consumo de ancho de banda y su capacidad para manejar múltiples dispositivos conectados.

## Plataforma de Nube

Para gestionar y almacenar los datos de detección de movimiento, así como para coordinar la entrega de mensajes por WhatsApp, es necesario utilizar una plataforma en la nube como Meta Cloud (también hay otros como AWS IoT, Google Cloud IoT o Azure IoT). Estas plataformas ofrecen servicios para manejar dispositivos IoT, almacenar datos y ejecutar lógica de aplicación.

## Desarrollo de Software

Se requiere desarrollo de software para programar la lógica del sistema de alarma en la ESP32, incluyendo la configuración de los sensores PIR, la comunicación con la plataforma de nube y la integración con la API de WhatsApp. Esto puede involucrar lenguajes de programación como Arduino IDE (para ESP32) y otros según los requerimientos específicos del proyecto.

1. **Requerimientos**

## Hardware

ESP32, sensores PIR, conexión Wi-Fi estable.

## Software

Programación en Arduino IDE para la ESP32, configuración de HTTP para la comunicación con la API de WhatsApp.

## Servicios en la Nube

Plataforma de nube para gestionar la lógica de la alarma y recibir datos de los sensores.

1. **Arquitectura de Software Utilizada**

## Modelo Cliente-Servidor con API Restful

- **Cliente**: La placa ESP32 actúa como el cliente en este modelo. Es responsable de enviar solicitudes HTTP POST al servidor en la nube para enviar mensajes de WhatsApp cuando se detecta movimiento.  
  
- **Servidor**: El servidor en la nube (representado por la URL `[https://graph.facebook.com/v16.0/111290641852610/messages`](https://graph.facebook.com/v16.0/111290641852610/messages%60)) actúa como el servidor en el modelo cliente-servidor. Es donde se recibe la solicitud HTTP POST con los datos del mensaje de WhatsApp y se procesa para su envío.  
  
- **API Restful**: Se utiliza una API Restful para la comunicación entre el cliente (ESP32) y el servidor en la nube. Esto se evidencia en el uso de HTTPClient para enviar la solicitud POST con el payload JSON que contiene los datos del mensaje a la API de WhatsApp.

## Capa de Aplicación IoT

- **ESP32 y Sensores PIR**: Forman la capa de aplicación IoT del sistema. La ESP32 maneja la lectura de los sensores PIR para detectar movimientos y posteriormente envía los datos al servidor a través de la API Restful.

## Protocolos de Comunicación

**Wi-Fi (IEEE 802.11)**

- **Uso**: La placa ESP32 utiliza Wi-Fi para conectarse a la red local y a Internet.  
- **Función:** Permite la comunicación bidireccional entre la ESP32 y servicios en la nube donde se gestiona la lógica de la alarma y la integración con WhatsApp.  
- **Beneficios**: Facilita una conexión estable y rápida, necesaria para la transmisión de datos de detección de movimiento y para recibir comandos de configuración o actualizaciones remotas.

**HTTP (Hypertext Transfer Protocol)**

- **Uso**: Protocolo estándar para la transferencia de datos en la web.  
- **Función**: Se utiliza para comunicarse con la API de WhatsApp a través de solicitudes POST para enviar mensajes de texto.  
- **Beneficios**: Facilita la integración con servicios externos como WhatsApp, permitiendo la notificación instantánea al usuario final cuando se detecta movimiento.

**TLS (Transport Layer Security)**

- **Uso**: Protocolo de seguridad que garantiza la privacidad e integridad de los datos durante la transmisión.  
- **Función**: Es esencial al utilizar Wi-Fi y HTTP para proteger las comunicaciones contra interceptaciones y manipulaciones no autorizadas.  
- **Beneficios**: Asegura que los datos sensibles, como las alertas de seguridad enviadas a través de WhatsApp, estén protegidos contra accesos no autorizados.

**HTTP y HTTPS**

Se utilizan para la comunicación entre la ESP32 y el servidor en la nube. HTTP se usa para enviar datos (en este caso, el mensaje de WhatsApp) al servidor, mientras que HTTPS (a través del uso de TLS) garantiza que esta comunicación sea segura y protegida.

## Componentes Principales de la Arquitectura

* + - ESP32: Cliente que envía datos (en este caso, el mensaje de WhatsApp) al servidor.
    - Servidor en la Nube: Recibe las solicitudes HTTP POST y procesa los mensajes de WhatsApp para su envío.
    - API de WhatsApp: Utilizada a través del servidor en la nube para enviar mensajes de texto a través de la plataforma WhatsApp.

1. **Patrón de Diseño**

## Patrón de Diseño Cliente-Servidor

**1. Cliente (ESP32):**  
- En el código, la ESP32 actúa como el cliente. Es responsable de iniciar y mantener la comunicación con el servidor en la nube.  
- Se establece una conexión Wi-Fi y se envían datos al servidor utilizando solicitudes HTTP.  
- El cliente (ESP32) está enfocado en la recolección de datos (detección de movimiento a través de los sensores PIR) y en la transmisión de estos datos al servidor.  
  
**2. Servidor en la Nube:**  
- El servidor en la nube (representado por la URL `[https://graph.facebook.com/v16.0/111290641852610/messages`](https://graph.facebook.com/v16.0/111290641852610/messages%60)) actúa como el servidor en el patrón Cliente-Servidor.  
- Es donde se reciben las solicitudes HTTP POST enviadas desde la ESP32.  
- Procesa los datos recibidos (en este caso, el mensaje de WhatsApp a enviar) y gestiona la lógica de la aplicación (como la autenticación con la API de WhatsApp y el envío efectivo del mensaje).

## Características del Patrón Cliente-Servidor en el Código

**Separación de Responsabilidades**: La ESP32 se encarga de la recolección de datos (detección de movimiento) y la inicialización de la comunicación. El servidor en la nube se encarga del procesamiento de los datos y la gestión de la lógica de la aplicación.

**Comunicación a través de Protocolos Estandarizados**: Se utiliza HTTP como protocolo de comunicación estándar entre el cliente (ESP32) y el servidor en la nube. Esto asegura una interoperabilidad y flexibilidad adecuada para la implementación del proyecto.

**Escalabilidad y Mantenimiento**: El patrón Cliente-Servidor facilita la escalabilidad del sistema, ya que nuevos clientes (más ESP32 con sensores PIR) pueden ser agregados sin afectar significativamente la infraestructura del servidor en la nube.

## Beneficios del Patrón Cliente-Servidor en este Contexto

**Seguridad**: Permite implementar capas de seguridad como TLS (HTTPS) para proteger la comunicación entre el cliente y el servidor.

**Eficiencia**: Distribuye las responsabilidades de manera eficiente entre el cliente y el servidor, optimizando el rendimiento del sistema.

**Flexibilidad**: Facilita la adopción de nuevas tecnologías y la evolución del sistema en el tiempo, adaptándose a cambios en los requisitos o en las capacidades de los dispositivos.

1. **Flujo de Funcionamiento**

## Detección de Movimiento

Los sensores PIR detectan movimiento en sus respectivos cuartos.

## Envío de Datos

La ESP32 recopila datos de los sensores PIR y los envía al servidor en la nube a través del protocolo HTTP.

## Procesamiento en la Nube

El servidor en la nube procesa los datos recibidos, determina si hay un evento de alarma (por ejemplo, movimiento detectado) y prepara un mensaje de alerta.

## Notificación por WhatsApp

El servidor utiliza la API de WhatsApp para enviar un mensaje de alerta al usuario final especificado, utilizando HTTP y TLS para garantizar la seguridad.

## Interacción Usuario

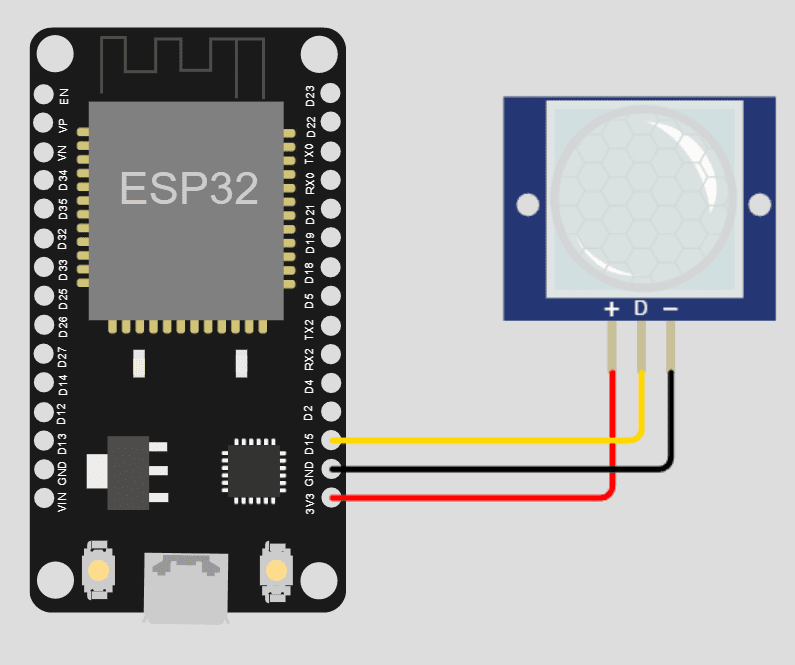
El usuario final recibe la alerta por WhatsApp y puede actuar en consecuencia según la situación detectada en la maqueta.

1. **Desarrollo del Proyecto**

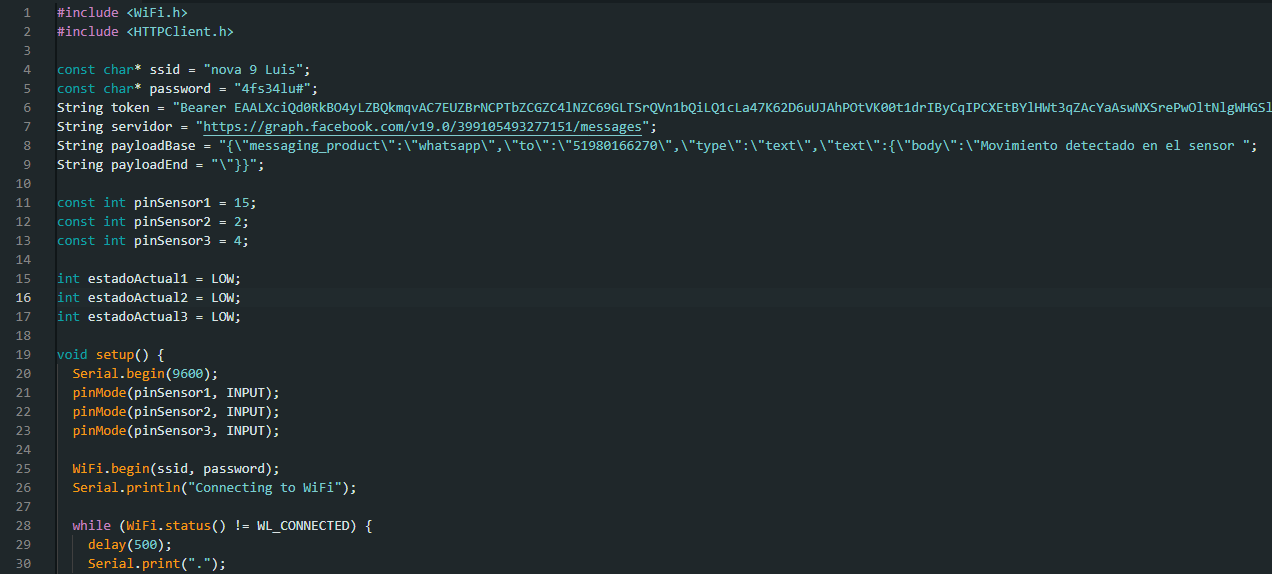
## Metodología utilizada

Para la implementación del sistema de alarma con detección de movimiento y notificación a WhatsApp, se seguirán los siguientes pasos:

1. **Selección de Componentes:**
   * Placa ESP32: Seleccionada por su capacidad de conectividad WiFi y Bluetooth, así como su facilidad de programación con Arduino IDE.
   * Sensores PIR: Tres sensores PIR serán utilizados para detectar movimiento en diferentes áreas de interés.
2. **Configuración del Hardware:**
   * Conexión de los sensores PIR a la placa ESP32.
   * Asegurar una alimentación adecuada y estable para todos los componentes.



1. **Desarrollo del Software:**
   * Programación de la placa ESP32 utilizando Arduino IDE.
   * Implementación de la lógica de detección de movimiento utilizando los sensores PIR.
   * Integración con la API de WhatsApp para enviar mensajes de texto.
   * Código:



Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Texto

Descripción generada automáticamente

1. **Pruebas y Depuración:**
   * Realización de pruebas para verificar la precisión de la detección de movimiento.
   * Depuración de posibles errores en la programación y la conexión de hardware.

## Actividades realizadas

Se desarrollaron distintas actividades para permitir que nuestro sistema de alarmas funcione, a continuación, vamos a detallar paso a paso cada una de ellas:

## API WhatsApp (Configuración)

Nos dirigimos a la página web de Meta for Developers o hacer clic al siguiente enlace.

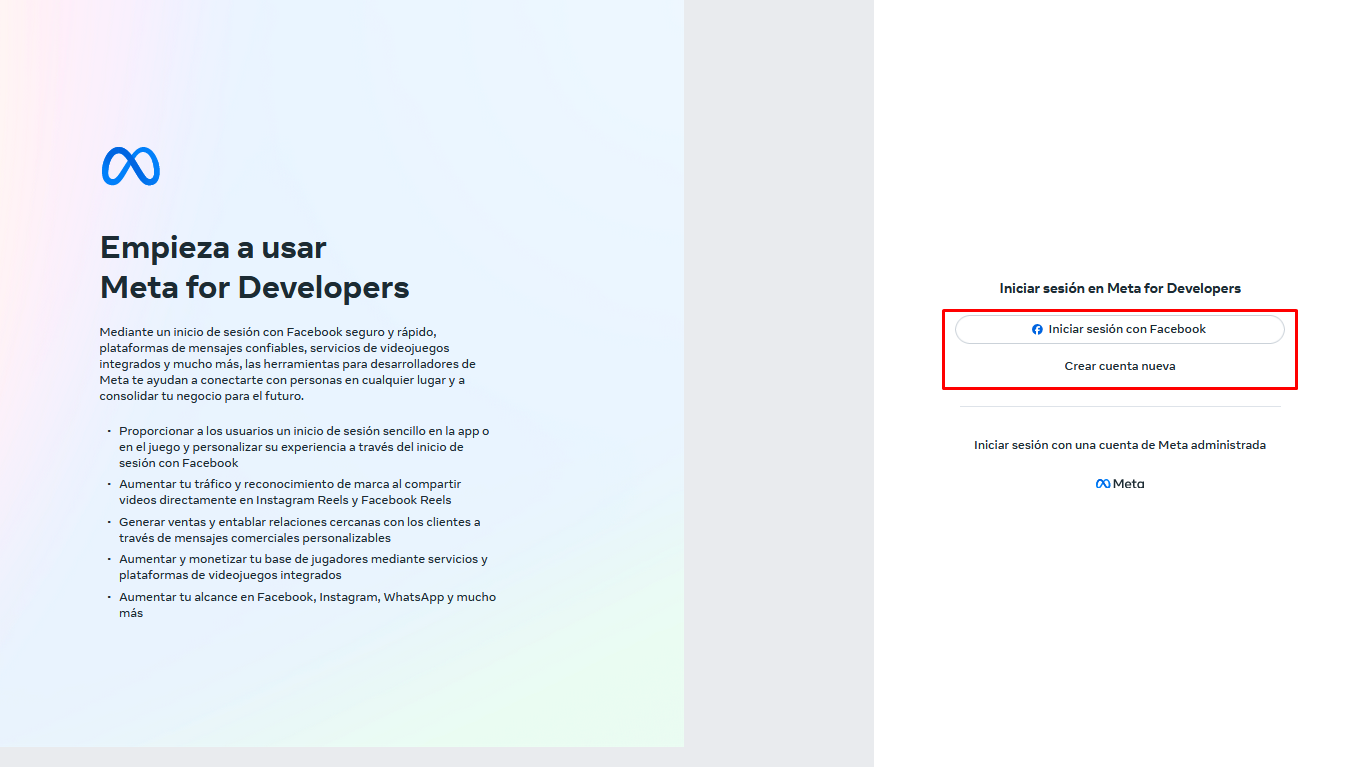
<https://developers.facebook.com/?no_redirect=1>

Elegimos la opción **Iniciar sesión**.

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Luego, elegimos una de las dos opciones **Iniciar sesión con Facebook** o **Crear cuenta nueva**.



Ingresamos las credenciales y le damos clic en **Iniciar sesión**.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Luego, nos direccionará a esta página y debemos elegir la opción **Empezar**.

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

En esta página debemos elegir **Crear app**.

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Luego, elegimos la opción que figura y clic en **Siguiente**.

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

Elegimos la opción **Otro** y luego clic en **Siguiente**.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Luego elegimos la opción **Negocios** y después en **Siguiente**.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Ponemos una descripción para nuestra App, el correo electrónico y luego clic en **Crear App**.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

En la página mostrada elegimos **Configurar** (donde figura el logo de WhatsApp).

Interfaz de usuario gráfica, Sitio web

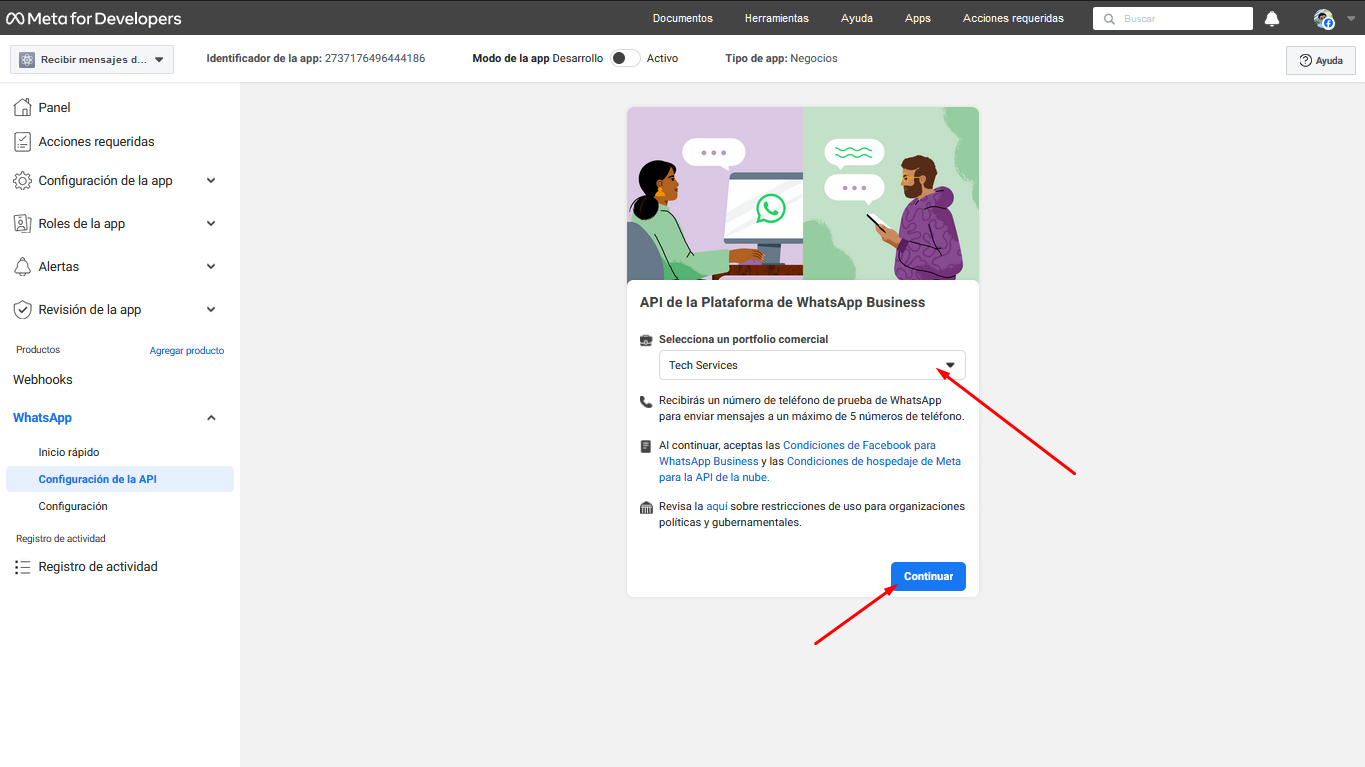
Descripción generada automáticamente

Luego clic en **Siguiente**.

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Luego, elegimos un portafolio comercial y le damos clic en **Continuar**.



Luego en la siguiente página que nos muestra, elegimos **Selecciona un número de teléfono del destinatario**.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Luego digitamos el número WhatsApp al cuan nos llegará los mensajes.

Una captura de pantalla de una computadora

Descripción generada automáticamente

Nos llegará un mensaje al WhatsApp con un código que debemos digitar en los recuadros mostrados y luego clic en **Siguiente**.

Una captura de pantalla de una computadora

Descripción generada automáticamente

Luego debemos copiar el token y configurar nuestro número de celular para que nos llegue el mensaje a nuestro aplicativo WhatsApp.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Ya quedaría listo para el envío de mensajes a WhatsApp.

## Instalamos la Librería de ESP32 en Arduino

Ahora vayamos a configurar la Librería de ESP32 en nuestro software Arduino.

Instalar el ESP32 en Arduino

Primero abrimos el IDE de Arduino en nuestro ordenador

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Elegir **Preferencias**.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Pegar el siguiente enlace en las partes señaladas en la captura de pantalla, luego clic en **OK**.

<https://dl.espressif.com/dl/package_esp32_index.json>

Captura de pantalla de un celular

Descripción generada automáticamente

Luego elegimos las opciones indicadas.

Interfaz de usuario gráfica, Aplicación

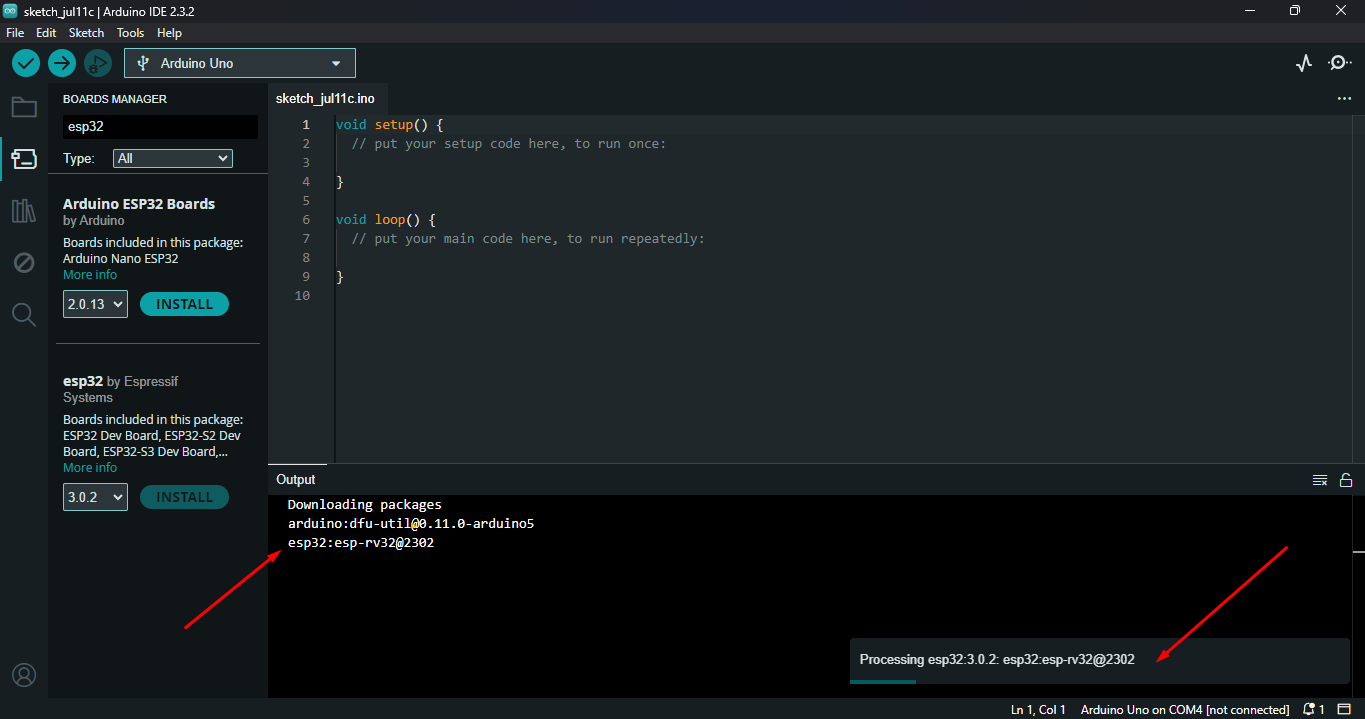
Descripción generada automáticamente

Realizamos la búsqueda del ESP32, al segundo resultado elegimos la opción de **Install**.

Texto

Descripción generada automáticamente

Esperemos a que cargue la operación.

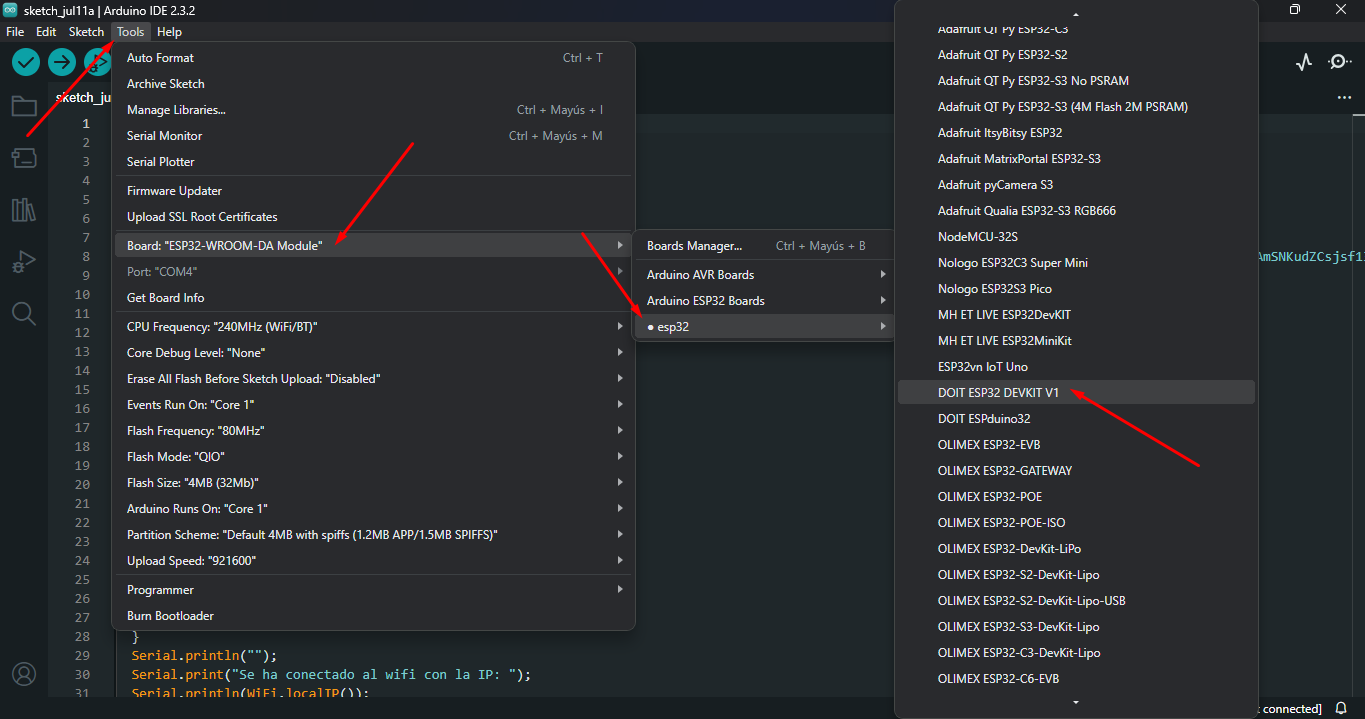


Listo, librería instalada.

Captura de pantalla de computadora

Descripción generada automáticamente

Ahora, debemos seleccionar nuestro ESP32 como equipo que estamos utilizando o vamos a conectar, tal como indica la captura.



En caso nuestro sistema operativo no nos detecta el controlador (driver) de nuestro ESP32, debemos descargarlo de la web.

Texto

Descripción generada automáticamente

## Instalando Driver del ESP32 para Windows

Una vez tengamos los drivers en nuestro sistema, procedemos a instalarlo, como indica en la captura.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Elegimos la opción señalada.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Buscamos la ubicación de los archivos del driver y le damos clic en **Aceptar**.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Validamos que sea la ruta correcta y le damos clic en **Siguiente**.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Finalmente se instala los controladores, y en nuestro caso esta como **COM5**.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Nuestro ESP32 ya esta configura y listo para usarse.

Como se aprecia, ya esta reconocido en Windows.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

## Seleccionamos el ESP32 en el IDE Arduino.

Ahora en nuestro Arduino se debe elegir el puerto de nuestro ESP32, tal como indica la captura.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Una vez hayamos seleccionado, nos debe figurar en la parte indicada que ya esta listo nuestro dispositivo para ser utilizado.

Texto

Descripción generada automáticamente

## Escribimos el código que va ejecutar el sistema.

Ahora, ponemos el código de nuestro programa en el entorno de desarrollo Arduino. Compilamos.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Compilación completada.

Captura de pantalla de computadora

Descripción generada automáticamente

Ahora subimos nuestro código al ESP32.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Código cargado en nuestro sistema ESP32.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

## Ejecutamos el código desarrollado.

Elegimos mostrar el Serial Monitor para evidenciar que ya está corriendo nuestro código.

Captura de pantalla de un celular

Descripción generada automáticamente

Listo, como se evidencia, ya esta en funcionamiento nuestro código.

## Recepción de mensajes en nuestro número de WhatsApp.

En caso se detecte un movimiento, nuestro ESP32 nos envía un mensaje por WhatsApp indicando lo ocurrido.

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

Descripción generada automáticamente

El número que nos asigna WhatsApp es aleatorio y gratuito.

Pantalla de computadora

Descripción generada automáticamente

1. **Resultados**

## Principales hallazgos y datos obtenidos

Durante el desarrollo y las pruebas del sistema de alarma con detección de movimiento, se lograron los siguientes resultados:

* Configuración exitosa de los sensores PIR con la placa ESP32.
* Implementación funcional de la detección de movimiento y la activación de alertas.
* Integración efectiva con la API de WhatsApp para el envío de mensajes de texto en tiempo real.

1. **Conclusiones**

## Eficiencia en la Detección de Movimiento

Los sensores PIR demostraron ser efectivos para detectar movimientos dentro de los diferentes cuartos de la maqueta. Su capacidad para detectar cambios en el calor emitido por objetos humanos los hace ideales para aplicaciones de seguridad.

## Integración Exitosa de Tecnologías

La utilización de la placa ESP32 permitió una integración fluida con los sensores PIR y la conectividad Wi-Fi. Esto facilitó la transmisión de datos a través de Internet hacia un servidor en la nube, donde se gestionó la lógica de la alarma y la integración con la API de WhatsApp.

## Notificaciones Instantáneas por WhatsApp

La configuración adecuada de la API de WhatsApp permitió enviar mensajes de alerta casi instantáneamente al usuario final cuando se detectó movimiento. Esto mejora significativamente la capacidad de respuesta ante eventos de seguridad.

## Seguridad y Privacidad

La implementación de protocolos seguros como TLS garantizó la protección de los datos transmitidos entre la ESP32, el servidor en la nube y la API de WhatsApp. Esto es crucial para asegurar que las alertas enviadas sean confiables y protegidas contra accesos no autorizados.

## Potencial de Escalabilidad y Adaptabilidad

El diseño del sistema permite su escalabilidad, lo que significa que puede expandirse fácilmente añadiendo más sensores PIR o dispositivos ESP32 según sea necesario. Esto lo hace adecuado para diferentes tamaños de espacio y requisitos específicos de seguridad.

## Aplicación de IoT

El proyecto demuestra de manera práctica cómo las tecnologías IoT pueden ser utilizadas para mejorar la seguridad residencial o comercial. La automatización del proceso de detección de movimiento y envío de alertas por WhatsApp muestra el potencial de IoT para soluciones cotidianas y útiles.

1. **Referencias**

## Fuentes consultadas

* + - Documentación oficial de ESP32 y Arduino IDE.
    - Especificaciones técnicas de los sensores PIR utilizados.
    - Documentación de la API de WhatsApp para desarrolladores.
    - Literatura sobre aplicaciones de IoT en seguridad residencial y comercial.