Informe Técnico: Proyecto IoT - Asistente de Hogar Inteligente

Introducción

Este informe detalla el desarrollo de un sistema de automatización del hogar utilizando microcontroladores y plataformas IoT. El dispositivo central fue el ESP-WROOM-32, que permite la conexión a redes Wi-Fi para la comunicación con servicios en la nube. El objetivo principal fue permitir el control remoto de dispositivos domésticos, como luces y motores, empleando Google Assistant como interfaz de usuario. Para ello, se evaluaron múltiples herramientas IoT, seleccionando Arduino Cloud por su compatibilidad y facilidad de implementación.

Herramientas Utilizadas y Evaluación

ESP Rainmaker

Descripción:
ESP Rainmaker es una plataforma desarrollada específicamente para dispositivos basados en ESP32, que facilita la integración con asistentes de voz como Google Assistant y Amazon Alexa. También incluye una aplicación móvil para el control de dispositivos IoT.
Ofrece funcionalidades avanzadas como la gestión de estados en la nube, un modelo de permisos para usuarios y soporte integrado para múltiples tipos de dispositivos (interruptores, termostatos, sensores, entre otros).

Ventajas:

- Configuración directa con dispositivos ESP32.
- Interfaz móvil amigable para usuarios finales.
- Soporte para comandos de voz sin necesidad de configuraciones complejas adicionales.

Desventajas:

- Limitación técnica crítica: En este proyecto, no fue posible cargar correctamente el código en el microcontrolador ESP-WROOM-32 debido a problemas de compatibilidad con la versión específica de firmware utilizada.
- Documentación limitada para depurar errores relacionados con la carga de firmware.

Motivo de Descartar:
A pesar de sus ventajas, la imposibilidad de realizar pruebas funcionales completas debido al problema con el firmware llevó a la decisión de no utilizar ESP Rainmaker en este proyecto.

Node-RED con MQTT y Ngrok

Descripción:
Node-RED es una herramienta de programación basada en flujos visuales que permite conectar dispositivos IoT de manera gráfica. En este proyecto, se utilizó junto con el protocolo MQTT para enviar y recibir mensajes entre el ESP-WROOM-32 y los dispositivos en la nube. Ngrok se empleó para crear túneles seguros que exponen servidores locales a Internet, facilitando las pruebas remotas.

Ventajas:

- Altamente personalizable: Permite la creación de flujos complejos para automatizar tareas específicas.
- Compatible con múltiples plataformas IoT y protocolos estándar como HTTP, MQTT y WebSockets.
- Ecosistema amplio: Una comunidad activa con múltiples nodos preconfigurados.

Desventajas:

- Inestabilidad: La conexión con el broker MQTT (HiveMQ) a través de Ngrok presentó problemas intermitentes, lo que dificultó la comunicación estable entre los dispositivos.
- Complejidad inicial: Configurar y sincronizar todos los elementos (Node-RED, MQTT, Ngrok) resultó más laborioso que el enfoque basado en Arduino Cloud.
- Requiere mantenimiento constante del túnel de Ngrok, que tiene limitaciones en su versión gratuita.

Motivo de Descartar:
Si bien esta solución es poderosa y versátil, la inestabilidad en las conexiones y la alta complejidad inicial no la hicieron adecuada para este proyecto, que requería simplicidad y confiabilidad.

Adafruit IO con IFTTT

Descripción:
Adafruit IO es una plataforma en la nube diseñada para aplicaciones IoT que permite almacenar y visualizar datos en tiempo real. En este proyecto, se utilizó en combinación con IFTTT (If This Then That), un servicio que automatiza tareas creando "recetas" basadas en eventos y acciones.

Ventajas:

- Interfaz intuitiva: Adafruit IO ofrece paneles de control gráficos que facilitan el monitoreo y control de dispositivos.
- Integración con múltiples servicios: IFTTT permite conectar Adafruit IO con asistentes de voz, redes sociales, notificaciones y más.
- Facilidad de configuración: Ambas plataformas ofrecen herramientas simples para usuarios sin experiencia avanzada.

Desventajas:

- Restricciones en las cuentas gratuitas:
  1. En Adafruit IO, la cuenta gratuita tiene un límite de datos almacenados y conexiones simultáneas.
  2. En IFTTT, la opción gratuita permite crear solo dos "recetas" (instrucciones), lo cual resultó insuficiente para las necesidades del proyecto.
- Dependencia de servicios pagos: Para desbloquear más recetas o capacidades, es necesario adquirir planes premium.

Motivo de Descartar:
Aunque ambas plataformas son accesibles y funcionales, las limitaciones de las versiones gratuitas (en especial las de IFTTT) hicieron inviable su uso para este proyecto, que requería múltiples controles y configuraciones.

Arduino Cloud

Descripción:
Arduino Cloud es una plataforma desarrollada por Arduino para gestionar dispositivos IoT. Su integración nativa con hardware Arduino y ESP32 permite la creación de aplicaciones IoT de manera sencilla. Incluye características como la definición de variables en la nube, un editor de código en línea y compatibilidad con asistentes de voz como Google Assistant.

Ventajas:

- Compatibilidad: El ESP-WROOM-32 fue reconocido sin problemas, lo que facilitó la carga del firmware y las pruebas iniciales.
- Estabilidad: La comunicación entre el ESP32 y la nube fue confiable durante todas las pruebas.
- Interfaz amigable: Incluye herramientas gráficas que reducen la curva de aprendizaje para nuevos usuarios.
- Integración con Google Assistant: Permite conectar dispositivos IoT con Google Home sin configuraciones adicionales complejas.

Desventajas:

- Algunas características avanzadas requieren una suscripción paga.
- La interfaz de usuario, aunque intuitiva, puede resultar limitada para usuarios con necesidades de personalización muy específicas.

Motivo de Selección:
Arduino Cloud ofreció la combinación perfecta de estabilidad, facilidad de uso e integración con Google Assistant, convirtiéndola en la opción ideal para el proyecto.

1.Pasos Claves del Desarrollo

1. Configuración en Arduino Cloud

1. Creación de la cuenta y proyecto IoT:
    - Accede a Arduino Cloud y crea una cuenta si no tienes una.
    - En la sección de *IoT Cloud*, crea un nuevo proyecto llamado *Thing*. Este *Thing* será el contenedor de las variables que representan las luces LED.
    - Asigna un nombre representativo como "Controlador_Luces".
2. Registro del ESP-WROOM-32:
    - Conecta el microcontrolador ESP-WROOM-32 al ordenador mediante un cable USB.
    - Desde la plataforma, utiliza el *IoT Cloud Device Manager* para registrar el ESP32.
    - Durante este proceso, la plataforma generará un identificador único (*Device ID*) y las credenciales necesarias para la conexión segura.
3. Definición de variables en la nube:
    - En el proyecto *Thing*, crea las variables necesarias para controlar las luces. Por ejemplo:
        - Luz1: Tipo *Boolean* (on/off), asignada al GPIO23.
        - Luz2: Tipo *Boolean* (on/off), asignada al GPIO22.
    - Configura cada variable para que sea accesible desde Google Assistant seleccionando la opción de integración con servicios en la nube.
    - Verifica que las variables estén correctamente vinculadas a pines GPIO específicos en el ESP32.
4. Descarga del archivo de configuración:
    - Descarga el archivo thingProperties.h, generado automáticamente por Arduino Cloud. Este archivo incluye la configuración necesaria para que el ESP32 se conecte a la nube.

2.Programación del ESP-WROOM-32

1. Preparación del entorno de desarrollo:
    - Asegúrate de tener instalado Arduino IDE y agrega soporte para el ESP32 desde el Administrador de Placas.
    - Instala las bibliotecas requeridas:
        - ArduinoIoTCloud.
        - WiFi.
2. Código base para el control de luces LED:
    - Crea un nuevo sketch en Arduino IDE e incluye el archivo thingProperties.h.
    - Define los pines GPIO y configura las funciones para controlar las luces:

```
/*
  Sketch generated by the Arduino IoT Cloud Thing "Untitled"
https://create.arduino.cc/cloud/things/420eca02-64a9-4dc5-a563-b6ab9edd04ba

  Arduino IoT Cloud Variables description

  The following variables are automatically generated and updated when changes are made to the Thing

  CloudSwitch cocina;
  CloudSwitch cuarto;
  CloudSwitch pasillo;

  Variables which are marked as READ/WRITE in the Cloud Thing will also have functions
  which are called when their values are changed from the Dashboard.
*/

#include "thingProperties.h"

const int Switch1 = 23; // Para el cuarto
const int Switch2 = 22; // Para la cocina
const int Switch3 = 21; // Para el pasillo
const int Switch4 = 19; // (Se puede usar para otro dispositivo o dejarlo vacío)

void setup() {
  // Inicializa la comunicación serial
  Serial.begin(9600);
  delay(1500);

  pinMode(Switch1, OUTPUT);
  pinMode(Switch2, OUTPUT);
  pinMode(Switch3, OUTPUT);
  pinMode(Switch4, OUTPUT);

  // Inicializa las propiedades de Arduino Cloud
  initProperties();

  // Conecta con Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);
```

```
41.
42.    setDebugMessageLevel(2);
43.    ArduinoCloud.printDebugInfo();
44.  }
45.
46.  void loop() {
47.    ArduinoCloud.update();
48.  }
49.
50.  /*
51.    Since cuarto is READ_WRITE variable, onCuartoChange() is
52.    executed every time a new value is received from IoT Cloud.
53.  */
54.  void onCuartoChange() {
55.    if (cuarto) {
56.      digitalWrite(Switch1, HIGH);
57.      Serial.println("Cuarto ON");
58.    } else {
59.      digitalWrite(Switch1, LOW);
60.      Serial.println("Cuarto OFF");
61.    }
62.  }
63.
64.  /*
65.    Since cocina is READ_WRITE variable, onCocinaChange() is
66.    executed every time a new value is received from IoT Cloud.
67.  */
68.  void onCocinaChange() {
69.    if (cocina) {
70.      digitalWrite(Switch2, HIGH);
71.      Serial.println("Cocina ON");
72.    } else {
73.      digitalWrite(Switch2, LOW);
74.      Serial.println("Cocina OFF");
75.    }
76.  }
77.
78.  /*
79.    Since pasillo is READ_WRITE variable, onPasilloChange() is
80.    executed every time a new value is received from IoT Cloud.
81.  */
82.  void onPasilloChange() {
83.    if (pasillo) {
84.      digitalWrite(Switch3, HIGH);
85.      Serial.println("Pasillo ON");
86.    } else {
87.      digitalWrite(Switch3, LOW);
88.      Serial.println("Pasillo OFF");
89.    }
90.  }
91.
```

- Sube el código al ESP-WROOM-32 utilizando Arduino IDE.

3.Pruebas iniciales del código:

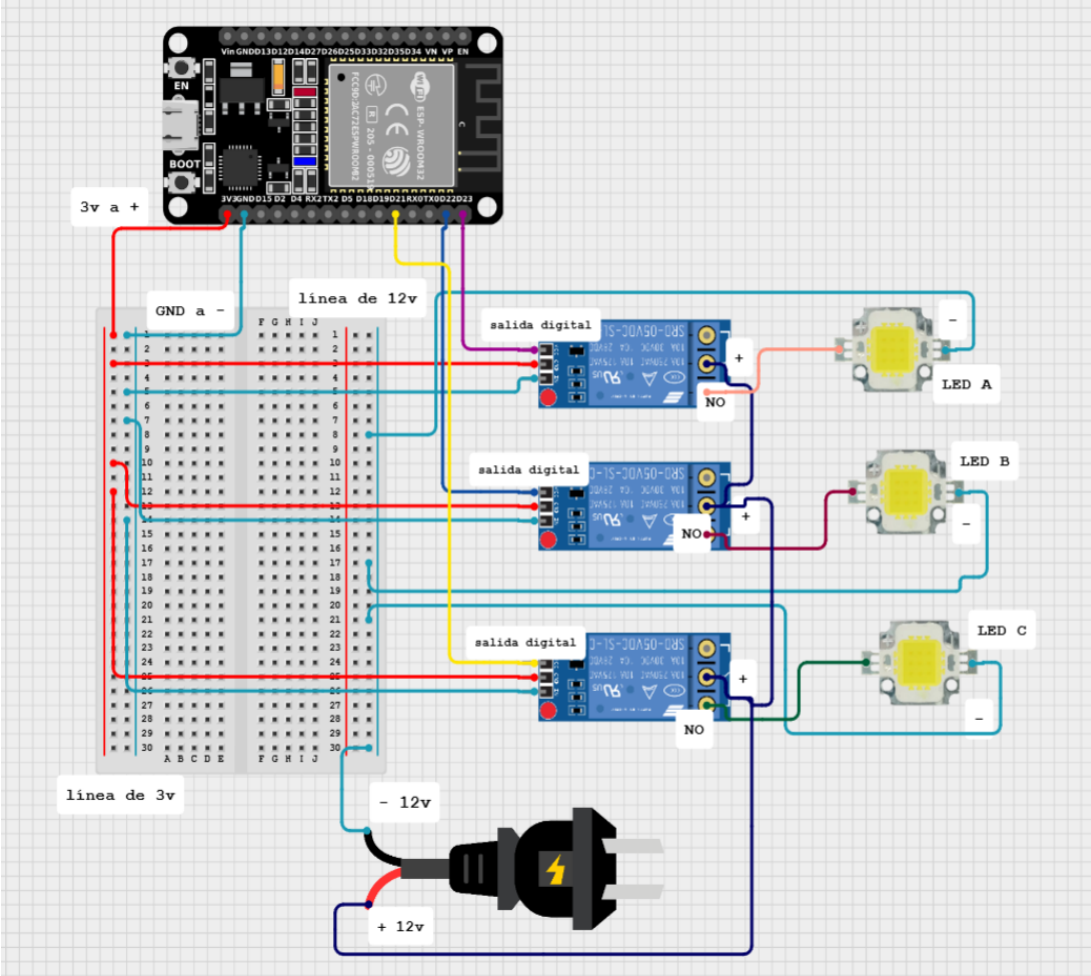
- Verifica la funcionalidad encendiendo y apagando las luces desde la consola de Arduino Cloud.
- Asegúrate de que las luces LED responden adecuadamente.

3. Integración con Google Assistant

1. Configuración en Arduino Cloud:
   - Activa la integración con Google Assistant desde la sección *Cloud Integrations*.
   - Vincula las variables Luz1 y Luz2 con interruptores compatibles con Google Home.
2. Sincronización con Google Home:
   - Desde la app de Google Home, añade Arduino Cloud como un dispositivo nuevo.
   - Vincula las luces registradas en Arduino Cloud para que puedan controlarse por comandos de voz.
3. Pruebas de comandos de voz:
   - Realiza pruebas con comandos como "Ok Google, enciende Luz 1" o "Ok Google, apaga Luz 2".

4. Esquema de Conexión

1. ESP-WROOM-32:
   - Alimentación: Conecta el pin 3V3 al positivo del relé y GND al negativo.
   - GPIO23 (Luz1): Controla la primera luz LED de 12V.
   - GPIO22 (Luz2): Controla la segunda luz LED de 12V.
2. Relés:
   - Entrada:
     - VCC: Conéctalo al pin 3V3 del ESP32.
     - GND: Conéctalo al GND del ESP32.
     - IN1 e IN2: Conéctalos a GPIO23 y GPIO22, respectivamente.
   - Salida:
     - Línea COM: Conéctala al terminal positivo de las luces LED.
     - Línea NO (Normally Open): Conéctala a la fuente de alimentación de 12V.
3. Luces LED:
   - Fuente de alimentación de 12V: Asegúrate de que las luces LED estén conectadas correctamente con una fuente externa de 12V para evitar sobrecargar el ESP32.



Futuras Mejoras: Mantenibilidad y Escalabilidad

Mantenibilidad

- Código modular:
  - Dividir funciones en módulos independientes para facilitar actualizaciones.
- Documentación técnica:
  - Proveer guías detalladas para usuarios y desarrolladores.
- Pruebas automatizadas:
  - Implementar un sistema que valide cambios antes de implementar.

Escalabilidad

- Sensores avanzados:
  - Integrar sensores de temperatura, movimiento y humedad.
- Optimización energética:

    - Usar convertidores de voltaje dedicados y explorar fuentes renovables, como paneles solares.
- Ampliación del ecosistema IoT:
    - Incorporar persianas automáticas, cámaras de seguridad y cerraduras inteligentes.
- Seguridad robusta:
    - Implementar cifrado avanzado y autenticación de dos factores.
- Compatibilidad multiplataforma:
    - Desarrollar aplicaciones para dispositivos móviles y soporte para asistentes como Alexa o Siri.

Conclusión

Arduino Cloud demostró ser la mejor opción para este proyecto, ofreciendo estabilidad, facilidad de implementación y una integración eficaz con Google Assistant. Este sistema constituye una base sólida para la expansión futura, permitiendo la adición de nuevos dispositivos, mejoras en la seguridad y avances en la automatización mediante inteligencia artificial.