Automatización de Ahorro de Energía en Sistema SmartHome

Introducción

La automatización en hogares inteligentes (SmartHome) tiene como uno de sus objetivos principales la optimización del consumo energético. Para lograrlo, los sistemas automatizados pueden gestionar el estado de los dispositivos conectados según su prioridad y uso, reduciendo así el gasto innecesario de electricidad.

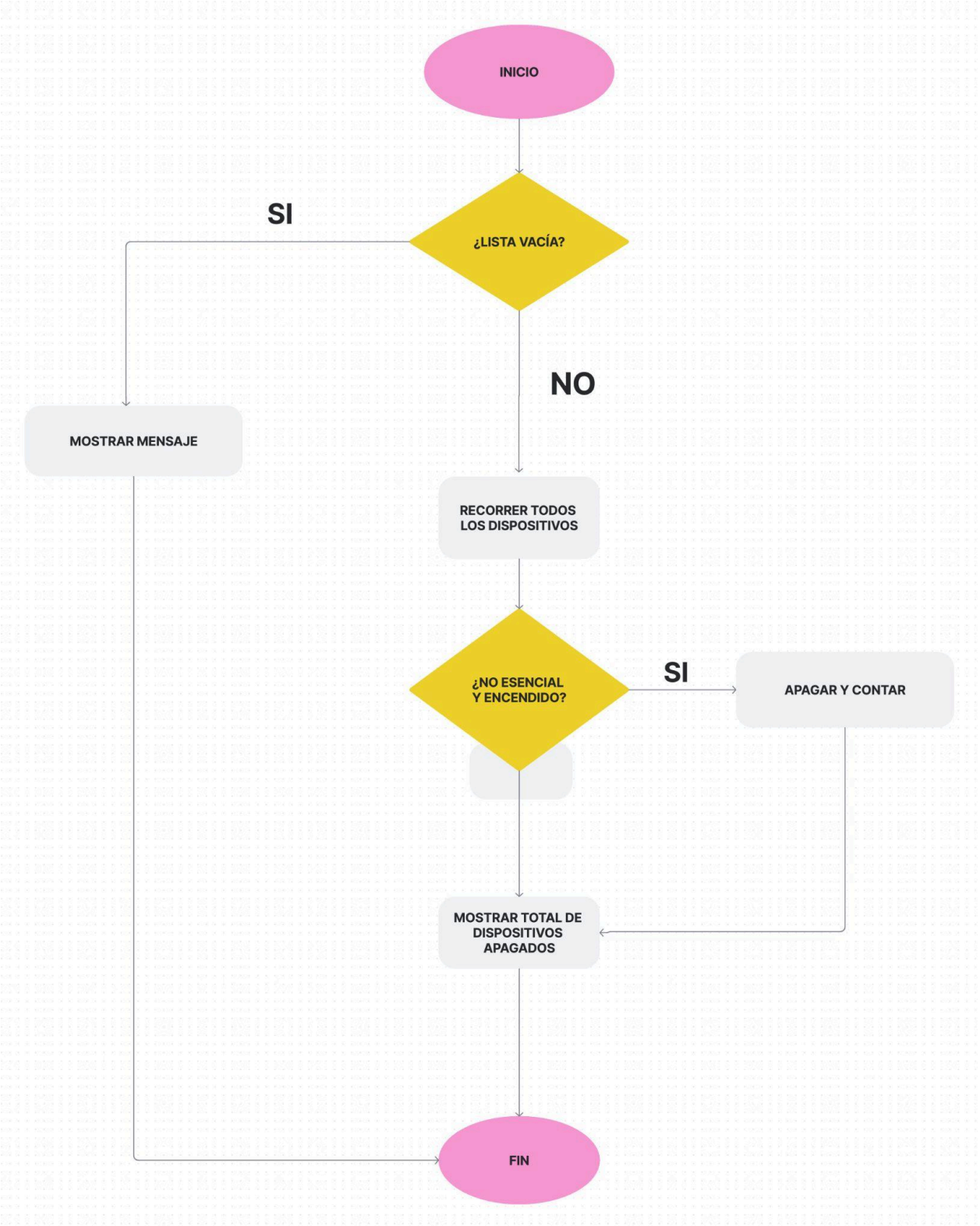
Funcionamiento del Script de Automatización

El archivo `administrador_automatizacion.py` contiene una función principal llamada `activar_modo_ahorro`, cuyo propósito es implementar un mecanismo automático de ahorro energético.

Función: `activar_modo_ahorro(dispositivos)`
Objetivo: Apagar todos los dispositivos conectados que no estén marcados como esenciales.
Parámetros:
- `dispositivos`: una lista de diccionarios, donde cada diccionario representa un dispositivo con al menos tres claves:
- `"estado"`: puede ser `"encendido"` o `"apagado"`.
- `"es_esencial"`: booleano que indica si el dispositivo es crítico para el funcionamiento básico del hogar.
- `"id"`: identificación/nombre del dispositivo.

Lógica de Funcionamiento

1. Si la lista de dispositivos está vacía, muestra un mensaje indicando que no hay dispositivos registrados.
2. Recorre todos los dispositivos de la lista.
3. Para cada dispositivo no esencial que esté encendido, cambia su estado a apagado.
4. Lleva un conteo de cuántos dispositivos fueron apagados.
5. Muestra un mensaje indicando cuántos dispositivos no esenciales fueron apagados.



Ventajas del enfoque

- Ahorro automático sin intervención del usuario.
- Priorización de dispositivos esenciales garantiza la comodidad y seguridad.
- Escalable a sistemas más complejos con sensores y horarios programados.

Interacción entre Módulos

El sistema está estructurado de forma modular para mantener la claridad y facilidad de mantenimiento. A continuación, se describe cómo se relacionan los componentes clave:

1. main.py

- Es el punto de entrada. Llama a `mostrar_menu()` desde `interfaz_menu.py`.

2. interfaz_menu.py

- Gestiona las opciones del usuario.

- Llama a funciones de `administrador_dispositivos.py` para gestionar dispositivos.

- Llama a `activar_modo_ahorro()` desde `administrador_automatizacion.py` para aplicar la lógica de ahorro.

3. administrador_dispositivos.py

- Agrega, elimina, lista y busca dispositivos. Esencial para alimentar los datos del sistema.

4. administrador_automatizacion.py

- Aplica la lógica de ahorro energético. Modifica los estados de los dispositivos no esenciales.

El flujo general es: main -> menú -> llamada a funciones de administración -> aplicación de lógica de automatización.

Ingreso controlado de datos esenciales

Este código usa el método get() al acceder a las claves "es_esencial" y "estado". Esto permite que, si alguna de esas claves no existe en un dispositivo, el programa no se detenga por error:

-dispositivo.get("es_esencial", False) devuelve False si no encuentra la clave, tratando el dispositivo como no esencial.

-dispositivo.get("estado") == "encendido" evita un KeyError si "estado" no está presente.

Así, el código maneja casos límite de forma segura, permitiendo que el sistema siga funcionando incluso con dispositivos mal definidos.

Conclusión

Este script es un ejemplo básico pero eficaz de cómo automatizar el ahorro de energía en un sistema SmartHome.