***“El código fuente debe ser consistente con el diagrama relacional”***significa que las entidades y relaciones que aparecen en el diagrama deben estar reflejadas correctamente en el código.  
***“Se sugiere no dejar las instrucciones de las evidencias en el README.txt”***

Se borro esa parte y se aclaró en commit

***“Confirmaciones no claras en cuanto a comentarios de cambios”***

Commitiar con mas detalle de modificaciones

**Diagrama relacional**

El diagrama incluye las siguientes entidades:

* **Usuarios**: id\_usuario, nombre, email, contraseña.
* **Dispositivos**: id\_dispositivo, nombre, tipo, estado, id\_vivienda.
* **Automatizaciones**: id\_automatizacion, descripcion, condicion, accion, id\_dispositivo.
* **Viviendas**: id\_vivienda, direccion, id\_usuario

**Entidad Automatizaciones**

* El código no tiene funciones que las manejen como entidad (crear, listar, modificar o eliminar automatizaciones).
* En el código no existe la **tabla** con los atributos: id\_automatizacion, descripcion, condicion, accion, id\_dispositivo.

**No hay persistencia ni relación con dispositivos**

* Las automatizaciones no se guardan en una lista, archivo o base de datos.
* No se vinculan dinámicamente con dispositivos (id\_dispositivo), como indica el diagrama.

**Relaciones entre entidades no reflejadas**

* En el diagrama, **Automatizaciones** está relacionada con **Dispositivos**
* En el código, estas relaciones no están implementadas explícitamente (por ejemplo, no hay funciones que permitan ver qué dispositivos están automatizados).

**Entidad Viviendas**

* No hay un módulo**viviendas.py** que maneje funciones como **crear\_vivienda**, **mostrar\_vivienda**, **buscar\_vivienda**

**No se refleja la relación con Usuarios**

* En el diagrama, cada vivienda está asociada a un usuario (id\_usuario), pero en el código no hay forma de vincular viviendas a usuarios.

**No se refleja la relación con Dispositivos**

* En el diagrama, cada dispositivo pertenece a una vivienda (id\_vivienda), pero en el código los dispositivos no están conectados a ninguna vivienda.

**Correcciones:**

1. Agregar al modulo **automatizaciones.py** funciones como **crear\_automatizacion**, **mostrar\_automatizaciones**, **buscar\_por\_dispositivo**

automatizaciones = []

def **crear\_automatizacion**(descripcion, condicion, accion, id\_dispositivo):

    nueva = {

        "id\_automatizacion": len(automatizaciones) + 1,

        "descripcion": descripcion,

        "condicion": condicion,

        "accion": accion,

        "id\_dispositivo": id\_dispositivo

    }

    automatizaciones.append(nueva)

def **mostrar\_automatizaciones**():

    for auto in automatizaciones:

        print(auto)

def **buscar\_por\_dispositivo**(id\_dispositivo):

    for auto in automatizaciones:

        if auto["id\_dispositivo"] == id\_dispositivo:

            print(auto)

1. crear un módulo **viviendas.py** con funciones que trabajen sobre una lista de diccionarios

viviendas = []

def **crear\_vivienda**(direccion, id\_usuario):

    nueva = {

        "id\_vivienda": len(viviendas) + 1,

        "direccion": direccion,

        "id\_usuario": id\_usuario

    }

    viviendas.append(nueva)

def **mostrar\_viviendas**():

    for v in viviendas:

        print(v)

def **buscar\_viviendas\_por\_usuario**(id\_usuario):

    for v in viviendas:

        if v["id\_usuario"] == id\_usuario:

            print(v)

**### IGNORAR ESTA PARTE NO SE TERMINO ###  
POO Para base de datos:**

**1. Usuario** (Representa a una persona que usa el sistema.)

Atributos:  
correo (str)  
nombre (str)  
contraseña (str)  
rol (str) - (admin o estándar)

**2. Vivienda (**Representa una casa o propiedad.**)**

* Contiene una lista de ubicaciones (self.ubicaciones) y automatizaciones (self.automatizaciones).
* Tiene métodos para agregar ubicaciones.

**3. Ubicacion**

* Representa una zona dentro de la vivienda (ej. cocina, patio).
* Contiene una lista de dispositivos.
* Tiene métodos para agregar dispositivos.

**4. Dispositivo**

* Representa un aparato que puede encenderse, apagarse y configurarse.
* Tiene atributos como nombre, tipo (ej. luz, aire acondicionado), estado (encendido/apagado) y configuraciones (como intensidad, volumen, etc.).
* Métodos:
  + encender() → cambia el estado a "encendido"
  + apagar() → cambia el estado a "apagado"
  + configurar(atributo, valor) → guarda configuraciones específicas

**5. Automatizacion**

* Representa una regla que se ejecuta automáticamente.
* Tiene atributos como nombre, hora de inicio y fin, lista de dispositivos afectados y estado (activa/inactiva).
* Método:
  + ejecutar(hora\_actual) → si la hora actual está dentro del rango, enciende los dispositivos asociados.

**Relaciones y principios aplicados**

* **Composición**: Vivienda → Ubicaciones → Dispositivos
* **Agregación**: Usuario → Viviendas
* **Encapsulamiento**: Cada clase gestiona sus propios datos y comportamientos
* **Abstracción**: Se modelan solo los atributos y métodos relevantes
* **Herencia**: No es necesaria por ahora, pero podría usarse si se agregan tipos de usuarios o dispositivos especializados