Isp_Act34

✅ Activity 34: Fix the Smart Device Design (Interface Segregation Principle)

🧠 Scenario:

You are developing a smart home system. You started by creating a `SmartDevice` interface with the following methods:

- `turn_on()`
- `turn_off()`
- `play_music()`
- `display_video()`

This worked well for a `SmartTV`.

But now you’ ve added a `SmartLight` — and you realize this device has **no use** for playing music or displaying videos.

You’ re being **forced to implement methods** that don’ t make sense for that device.

That’ s a violation of the **Interface Segregation Principle**.

🔧 Instructions:

1. Review the current `SmartDevice` interface that includes all methods.
2. Identify which methods make sense for which devices.
3. Refactor the interface:
   - Create **smaller, more specific interfaces** (e.g., switchable, music, and video features).
4. Assign each class to implement **only the interfaces it actually needs**:
   - `SmartLight` should only turn on/off.
   - `SmartTV` may support all functionalities.
   - Optionally, add a `SmartSpeaker` that only plays music.
5. Test your new structure by creating one object of each class and calling only the methods it supports.

🎯 Challenge Bonus:

Can you add a new device called `SmartFan` that supports only turning on/off — **without editing your existing interfaces or classes**?