Objects get **too tightly coupled** when multiple operations (e.g., tax, discount) are embedded in them.

We need a **Visitor interface** to **separate operations from objects**.

Objects provide an accept(visitor) method to **delegate behavior to visitors**.

Visitor uses **double dispatch** to run the right operation for each object type without type checks.

This allows **adding new operations without modifying object classes** (follows Single Responsibility Principle).

Visitor pattern **decouples “what to do” from “who to do it on”**, improving maintainability and extensibility.

Best when object types are fixed but operations change or grow frequently.