Reflection: Why stack naturally models undo operations?

A stack follows the LIFO (Last In, First Out) principle: the last thing you put on top is the first thing you take out.

Undo operations in real systems (like Irembo forms, text editors, or browsers) are reversals of recent actions. For example:

- In Irembo, if you just clicked “Confirm”, pressing back (undo) should take you from Confirm → Fill Address, not all the way back to Fill Name.

- In a text editor, if you typed letters A → B → C, pressing Undo removes C first, not A.

- In a browser, the last visited page is the first one you go back from.

This makes stack a natural model because:

- Preserves history in order – Every new action is pushed onto the stack. The most recent action is always on top, ready to be undone first.

- Locality of change – Undo is concerned with the latest modification, not something in the middle. Stack captures this by design.

- Reversible actions – When you pop from the stack, you expose the previous state. This mirrors how undo restores the system to just-before-last action.

- Efficient retrieval – Both push (recording a new action) and pop (undoing it) are O(1) operations, making stacks extremely efficient for real-time undo systems.

So in summary:

A stack is a history keeper. It grows as you perform actions and shrinks as you undo them. Its LIFO nature mirrors the logic of undo, where the most recent action is the first candidate for reversal.