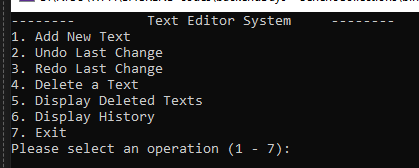
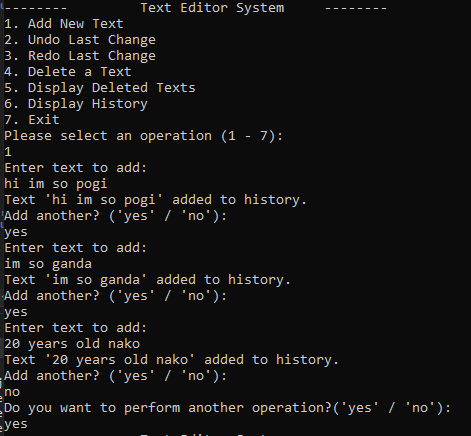


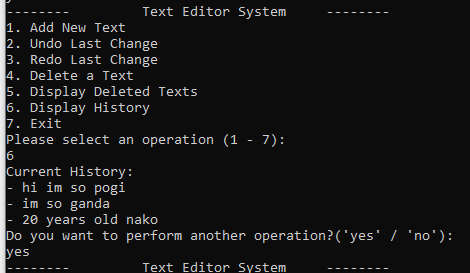
**OUTPUT**

****

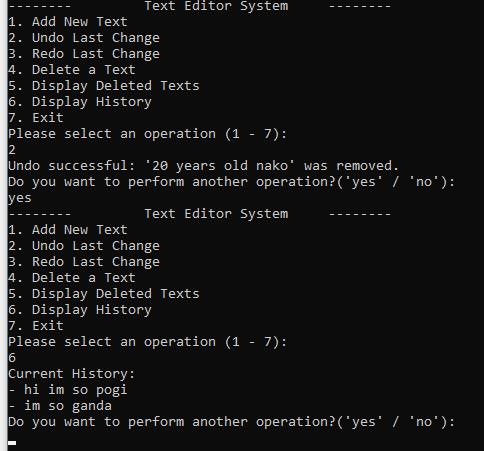
**ADD TEXT**

****

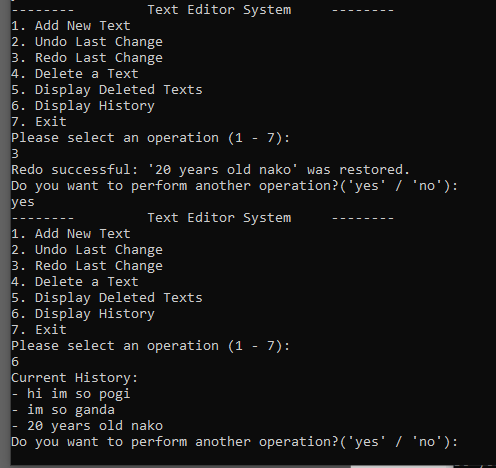
**DISPLAY HISTORY**

****

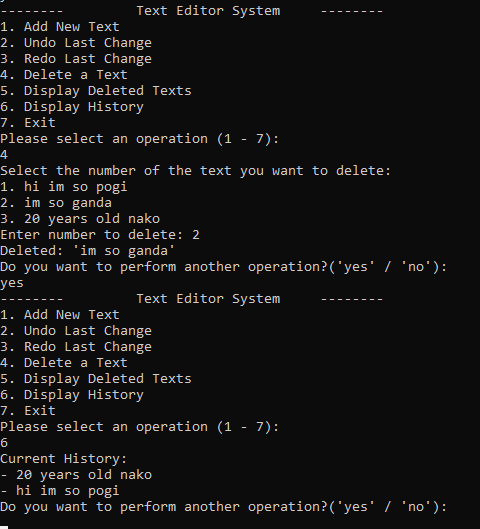
**UNDO LAST CHANGE**

****

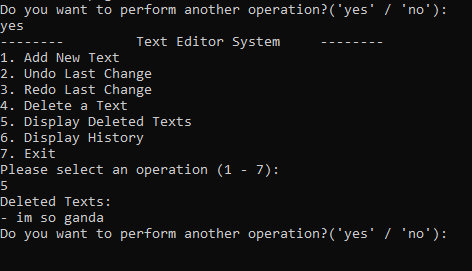
**REDO LAST CHANGE**

****

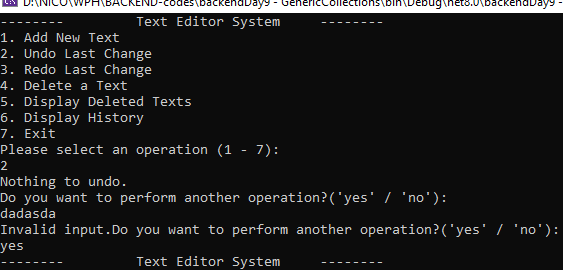
**DELETE A TEXT**

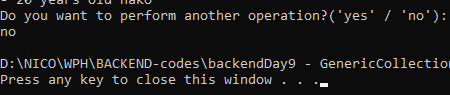
****

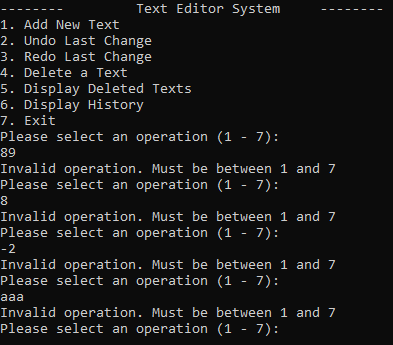
**DISPLAY DELETED TEXT**

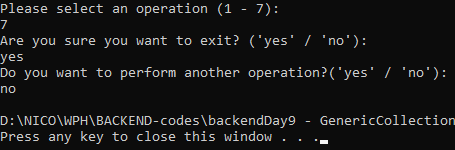
****

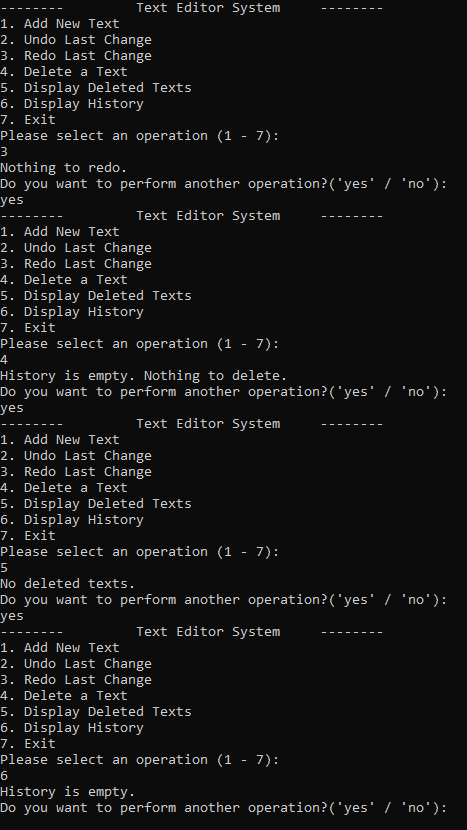
**VALIDATIONS**

****

****

****

****

****

**REFLECTION**

In my program, I organized everything by separating different parts into their own classes to keep things clean and easy to manage. I created a TextEditor class that handles all the main features like adding text, undoing, redoing, deleting, and showing history or deleted texts. Each action is in its own method, which helped me avoid repeating code and made the program easier to read and update. I also made an InputHelper class to handle all the user inputs and validation, like checking if the input is empty or if the user picked a number within a valid range for the menu.

I used try-catch blocks especially in the InputHelper methods, mostly to catch invalid inputs. For example, if the user enters something that’s not a number or leaves a text field blank, the program shows a helpful error message instead of crashing. I made sure that the user always sees clear messages — whether they typed something wrong or successfully did an action like adding or deleting text. I also added confirmation prompts like "yes" or "no" to guide them through the process smoothly.

One thing I found a bit tricky was the undo and redo system, especially how to properly clear the redo stack whenever a new text is added. Deleting a specific item from the stack was very challenging for me since stacks don’t support direct access and I encountered different errors trying it, so I had to convert it to a list, delete the selected item, and then rebuild the stack. But I learned a lot from doing that. I also added a feature that keeps track of deleted texts in a separate list and lets the user view them later, which I thought was a cool addition. Overall, this project helped me understand how to use generic collections like stacks and lists in a real program, and how to make a simple but user-friendly console application in C#.