
Introduction

This Python program employs Tkinter to create a user-friendly text correction tool. The graphical interface allows users to input or load text from a file, offering features such as grammatical correction using the TextBlob library, removal of punctuation marks, and spell-checking with the SpellChecker library. The corrected text, text without punctuations, potential misspelled words, and suggested corrections are presented in distinct sections of the interface. Users can conveniently save the corrected text to a new file. The tool serves as a helpful resource for enhancing the precision and clarity of textual content.

Team Members:

Sharan Poojari - 221080058

Tanmay Mene - 221080044

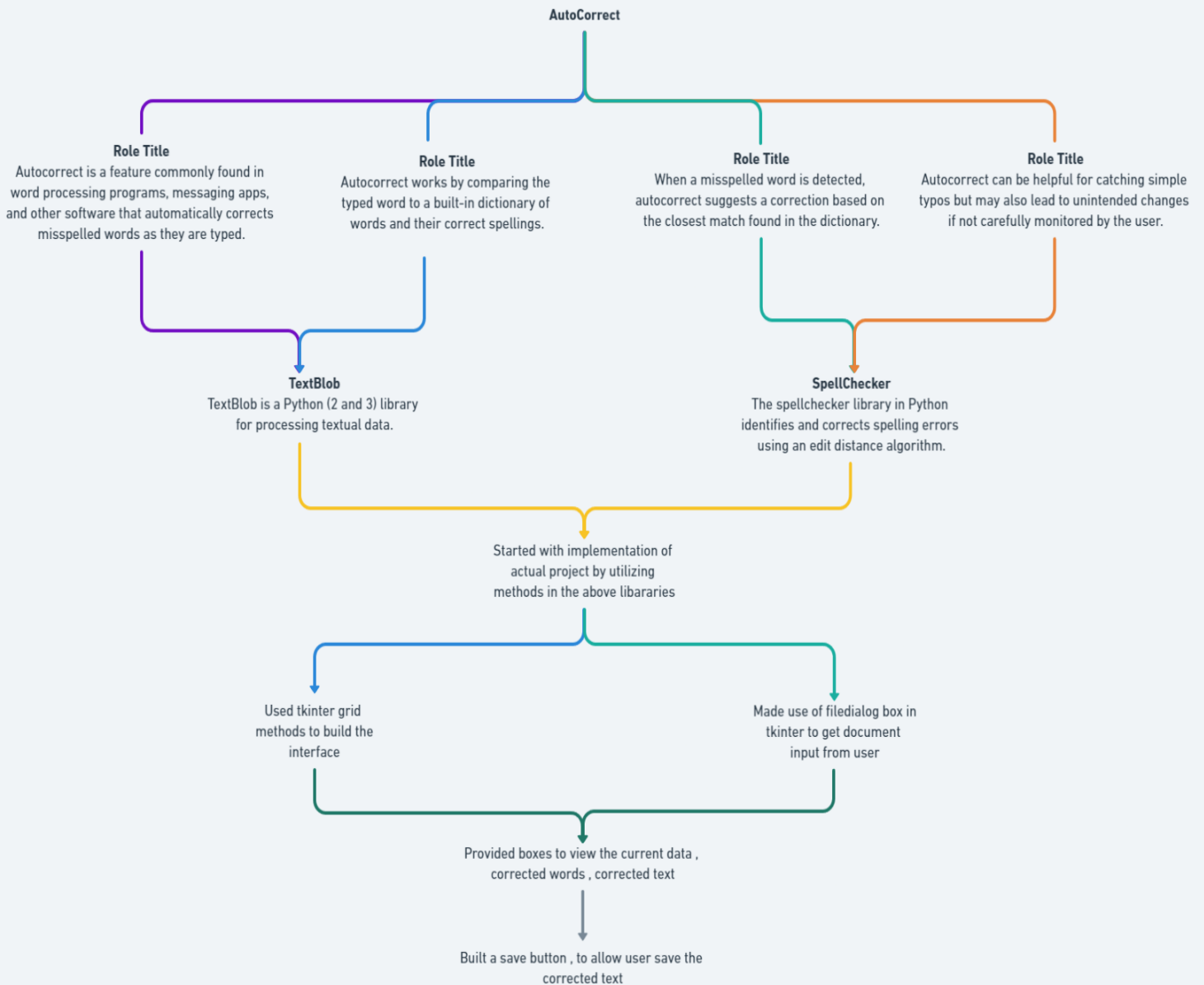
Problem Statement:

Create a user-friendly Python tool for spell-checking documents, including text files, Word docs, and PDFs. The goal is to enhance accuracy and professionalism by efficiently identifying and correcting spelling errors in written content.

Motivation:

We want to make a helpful tool because everyone makes spelling mistakes, and it can make writing look not so good. This tool will be easy for people to use and fix spelling errors in different types of documents like essays or reports. By using this tool, we hope people can write better and save time checking for mistakes.

Methodology:



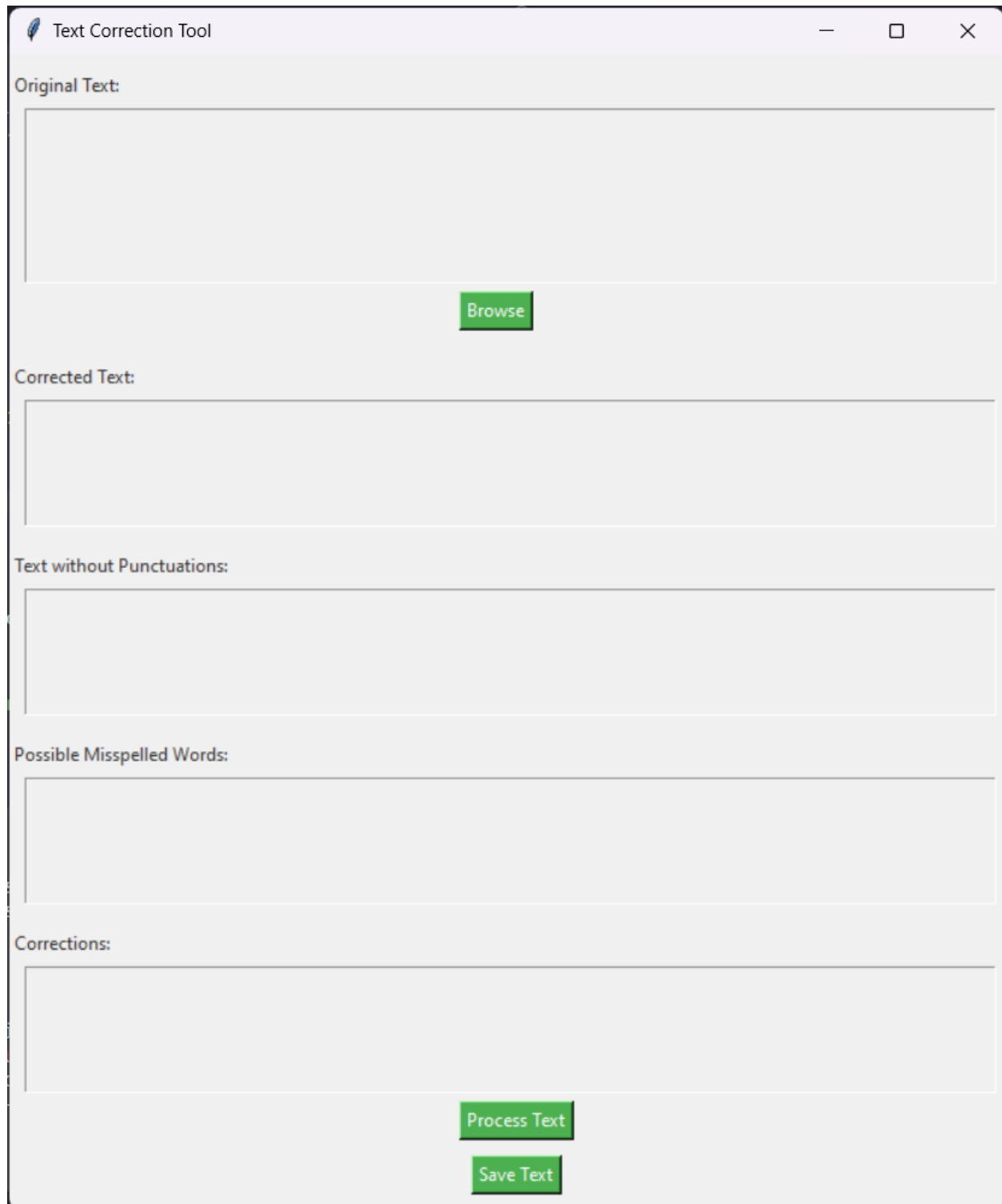
Detailed explanation of each function:

1. **useTextBlob(text):** This function uses the TextBlob library to correct the spelling in the input text. It creates a TextBlob object from the input text and calls the correct() method on it, which returns a TextBlob object with corrected spelling. The function then converts this object to a string and returns it.
2. **removePunctuations(text):** This function removes all punctuation from the input text. It defines a string of punctuation characters, then iterates over each character in the input text. If the character is not in the punctuation string, it adds the character to a new string. The function then returns this new string, which is the input text without any punctuation.
3. **useSpellchecker(text):** This function uses the SpellChecker library to check the spelling of the words in the input text. It first removes punctuation from the text, then splits the text into a list of words. It identifies any unknown words in the list (i.e., words that are likely misspelled) and for each unknown word, it finds the most likely correct spelling using the correction() method from the SpellChecker library. The function returns a dictionary where the keys are the misspelled words and the values are the corrected spellings.
4. **browseFile():** This function opens a file dialog that allows the user to select a file to open. If a file is selected, the function reads the content of the file and inserts it into a text box in the user interface.
5. **processText():** This function gets the text from a text box in the user interface and processes it in several ways: it corrects the spelling using TextBlob, removes punctuation, and checks the spelling using SpellChecker. It then updates several text boxes in the user interface with the processed text and the results of the spelling checks.
6. **askForSaving():** This function opens a file dialog that allows the user to select a file to save to. It gets the text from a text box in the user interface, corrects the spelling using TextBlob, and writes the corrected text to the selected file.

Pseudo Code:

1. Import necessary libraries (tkinter, re, filedialog, TextBlob, SpellChecker)
2. Initialize SpellChecker
3. Define a function 'useTextBlob' that takes a text as input and returns the corrected text using TextBlob
4. Define a function 'removePunctuations' that takes a text as input and returns the text after removing all punctuations
5. Define a function 'useSpellchecker' that takes a text as input, removes punctuations, splits the text into words, identifies misspelled words, and returns a dictionary of corrections for each misspelled word
6. Define a function 'browseFile' that opens a file dialog for the user to select a file, reads the content of the file, and inserts the content into the 'originalText' text box
7. Define a function 'processText' that:
 - Retrieves the text from the 'originalText' text box
 - Corrects the text using TextBlob and inserts the corrected text into the 'correctedText' text box
 - Removes punctuations from the text and inserts the punctuation-free text into the 'textWithoutPunctuationTextbox' text box
 - Checks the spelling of the punctuation-free text using SpellChecker and inserts the misspelled words and their corrections into the 'misspelledWordsTextbox' and 'correctionsTextbox' text boxes respectively
8. Define a function 'askForSaving' that opens a save file dialog for the user to select a file, retrieves the text from the 'originalText' text box, corrects the text using TextBlob, and writes the corrected text into the selected file

Output:



The screenshot shows a web application window titled "Text Correction Tool". The interface is divided into several sections for text processing:

- Original Text:** A large text input area at the top.
- Browse:** A green button located below the "Original Text" input.
- Corrected Text:** A large text output area below the "Browse" button.
- Text without Punctuations:** A large text output area below the "Corrected Text" area.
- Possible Misspelled Words:** A large text output area below the "Text without Punctuations" area.
- Corrections:** A large text output area below the "Possible Misspelled Words" area.
- Process Text:** A green button located below the "Corrections" area.
- Save Text:** A green button located below the "Process Text" button.

Discussion:

The Text Correction Tool described is a comprehensive application designed to enhance the quality of textual content. It leverages the capabilities of Python libraries, TextBlob and SpellChecker, to rectify grammatical and spelling errors. The tool is equipped with a user-friendly Graphical User Interface (GUI), which allows users to effortlessly input text, initiate the correction process, and view the corrected text. This makes the tool an invaluable asset for proofreading and improving text. To deal with global user base, the tool could be further improved by incorporating support for additional languages. This would involve using different dictionaries or language models for the spell checking process. Lastly, the user experience could be significantly enhanced by improving the visual design of the GUI. This could include clear labels for buttons, logical organization of elements, or the use of color to guide the user's attention, making the interface more modern and intuitive.