How to make a plagiarism detector in Python

Build a plagiarism detection engine that can help you understand string matching, file operations, and the user interface. You also discover natural language processing techniques to enhance applications.

In this article, let's learn how to build a plagiarism checker and powerful features of the Difflib module with TipsMake.com!

Tkinter and Difflib . Modules

To build a plagiarism detector, you will use the Tkinter and Difflib modules. Tkinter is a simple, cross-platform library that you can use to create graphical user interfaces quickly.

The Difflib module is part of the Python standard library, which provides classes and functions that compare strings such as strings, lists, and files. Thanks to it, you can build programs like autocorrect text, simple version control system or a text summarization tool.

How to build a plagiarism detector in Python

Import the required modules. Define a method load_file_or_display_contents() that takes entry and text_widget as arguments. This method will load a text file and display its content in a text widget.

Use get() to get the file path. If the user does not enter any information, use askopenfilename() to open a file dialog window to select the file you want to check for plagiarism. If the user selects this file path, deletes the previous entry, if any, from start to finish and inserts the selected path.

```
import tkinter as tk from tkinter import filedialog from difflib import SequenceM
```

Open the file in read mode and save the content in the text variable . Delete the content of text_widget and insert the text you retrieved earlier.

```
with open(file_path, 'r') as file: text = file.read() text_widget.delete(1.0, t
```

Define a method, compare_text() that you will use to compare two pieces of text and calculate the percentage similarity. Use Difflib's SequenceMatcher() class to compare strings and determine similarities. Set the custom comparison function to None to use the default comparison and pass the text you want to compare.

Use scaling to determine similarity in a floating-point format that you can use to calculate percentage similarity. Use get_opcodes() to retrieve a group of operations that you can use to highlight similar sections of text and return that section along with the percentage of similarity.

```
def compare_text(text1, text2): d = SequenceMatcher(None, text1, text2) similari
```

Define a show_similarity() method . Use get() to retrieve the text from both text boxes and feed them to the compare_text() function . Delete the content of the resulting textbox and insert the percentage of similarity. Remove the ' same ' tag from the previous highlight (if any).

```
def show_similarity(): text1 = text_textbox1.get(1.0, tk.END) text2 = text_textbo
```

get_opcode() returns 5 tuples: opcode string, first string start index, first string end index, second string start index, and second string end index.

The opcode string can be one of four values: replace, delete, insert, and equal. You would use replace when part of the text in both strings is different, and someone has replaced part of the content with another. Delete will be used when part of the text exists in the first string, not the second.

Insert is used when part of the text is not present in the first string but in the second string. You get equal results when the pieces of content are the same. Store all these values ??in the appropriate variables. If the opcode string is equal , add the same tag to the text string.

```
 for opcode in diff: tag = opcode[0] start1 = opcode[1] end1 = opcode[2] start2 =
```

Initialize the Tkinter root window. Name the window and define a frame within it. Arrange the frame with appropriate padding in both directions. Define two labels to show Text 1 and Text 2 . Set the parent component it's inside and what it displays.

Define 3 text boxes, two for the text you want to compare and one to show the results. Declare the parent element, width and height, set the packing option to tk.WORD to ensure that the program wraps words at the nearest boundary and doesn't break any words in between.

```
root = tk.Tk() root.title("Text Comparison Tool") frame = tk.Frame(root) frame.pa
```

Define 3 buttons, two to download files and one to compare. Specifies the parent element, the text it will display and the function it will run when it is clicked. Create two input widgets to enter the file path and define the parent element and its width.

Arrange all these elements in rows and columns using the grid manager. Use pack to sort compare_button & text_textbox_diff . Add the appropriate padding at the required position.

```
file_entry1 = tk.Entry(frame, width=50) file_entry1.grid(row=1, column=2, columns
```

Highlight text has been highlighted the same on yellow background and red font color.

```
text_textbox1.tag_configure("same", foreground="red", background="lightyellow")
```

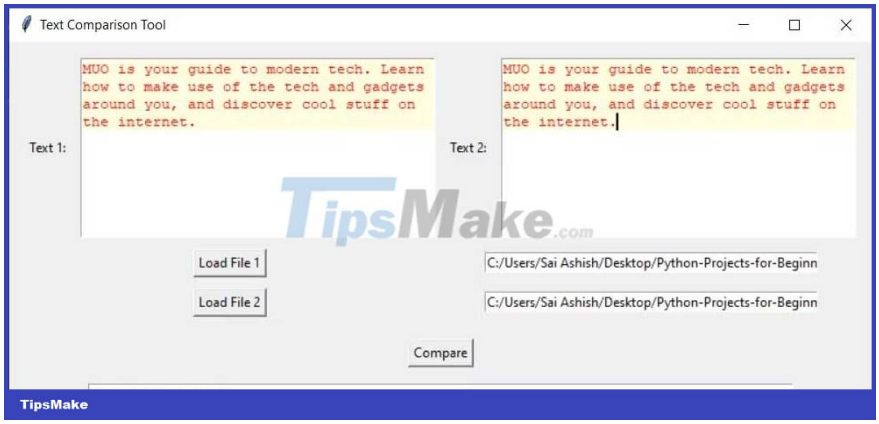
The mainloop() function tells Python to loop through the Tkinter event and listen for the event until you close the window.

```
root.mainloop()
```

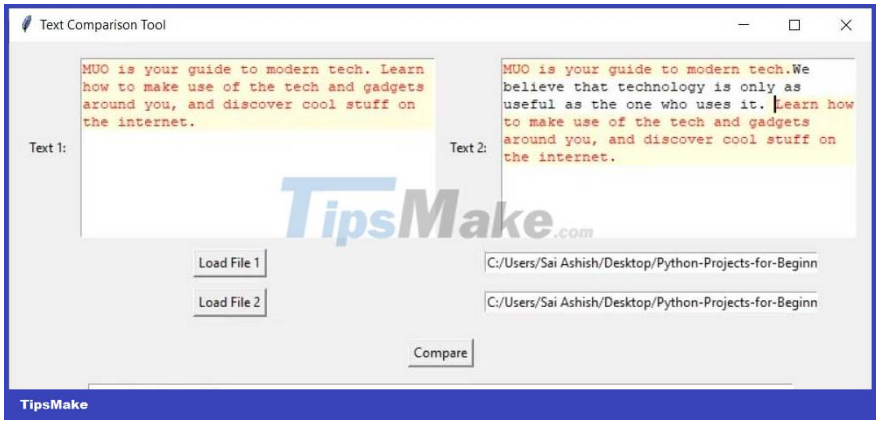
Put it all together and run the code to detect plagiarism.

Example results of plagiarism detection tool

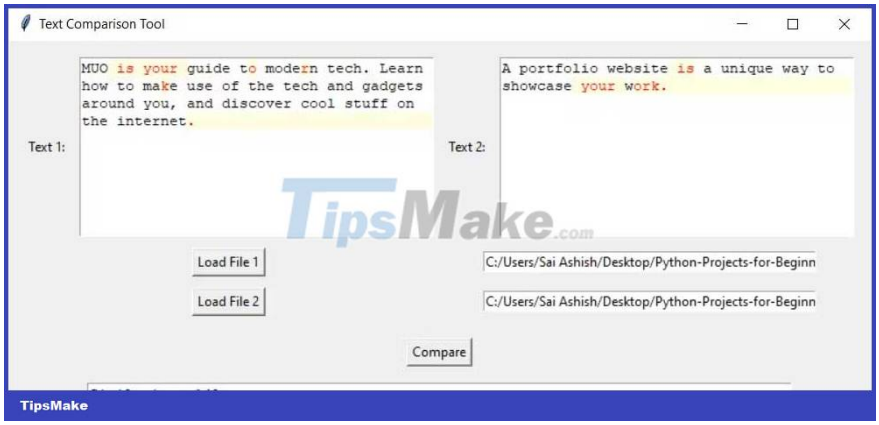
When running this program, it shows a window. When the Load File 1 button is pressed, a file dialog box opens and asks you to select the file. When selecting a file, this program displays the contents of the first text box. When entering the path and clicking Load File 2 , the program displays the content in the second text box. When you click the Compare button , you will have 100% similarity and it will highlight all the same text exactly.



If you add another line to a textbox, and then click Compare , this program highlights the same part and keeps the rest.



If there are very few similarities, the program highlights some letters or words, but the percentage of similarity is quite low.



Above is how to create a plagiarism detection tool in Python . As you can see, it's pretty simple, isn't it? Good luck!

You finished reading the article "**How to make a plagiarism detector in Python**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.
