**Project Name:** Plagiarism Checker

**Github Link:** https://github.com/projectsforstudents2022/Plagiarism\_Checker.git

**Why was this project created?**

The internet provides answers to all queries in the modern world. Therefore, anyone can quickly copy and utilize content from the web at any time, this is known as Plagiarism. When someone plagiarizes a piece of writing, they rephrase it, copy it, and omit sources. Plagiarism is challenging to identify since people often paraphrase texts rather than copying them verbatim.

**What problem is it solving?**

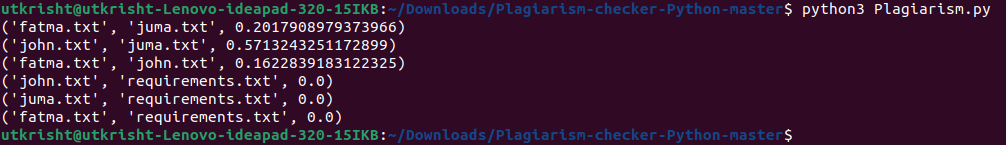
Any plagiarism detector's goals include identifying textual similarities and verifying the document's originality. Furthermore, it suggests that no text in the paper is a direct copy of another author's. The original document is indexed, and the copied document is compared to a set of previously saved documents using cosine similarity.

**Entire explanation of project**

* **PROPOSED APPROACH**

We unveiled a technique to identify plagiarism in English. It is based on a content-based approach that primarily involves identifying the texts' fingerprints based on English language peculiarities and comparing their logical representations using heuristic techniques. The texts are simplified and changed into better representations by preprocessing. The key preprocessing steps are tokenization, stop-word elimination, rooting, and synonym substitution. The logical document representation's primary objective is to reduce computing time by avoiding pointless comparisons. Because of this, we establish heuristic algorithms for the document, paragraph, and sentence levels of the tree. Two documents are compared at the document level using their shared hashes and a predetermined threshold. The comparison procedure moves on to the next level (paragraph) if there is a likely similarity between the documents and the number of hashes in the intersection subset is larger than a threshold; otherwise, it is stopped. If resemblance is found at the paragraph level, we move on to the sentence level if necessary. We use the Longest Common Substring (LCS) metric to determine whether two phrases may be similar. We compare the length of the LCS with the length of the minimum sentence multiplied by a specific threshold after computing the LCS between two sentences. If the LCS is longer than the threshold, we look for similarity between the two sentences; otherwise, the procedure moves on to the following phrase.

* **RESULT**



**CONCLUSION**

The software that has been developed can look for terms that have been used before and calculate a match score using cosine. It is based on a content-based approach that primarily involves comparing logical representations of texts by employing heuristic methods and fingerprinting them according to language specificity.