

**Advanced College of Engineering and Management**

**A PROJECT REPORT ON PLAGIARISM TEST**

**“TrueType: A Web-Based Plagiarism Detection System Using Sentence-BERT, Cosine Similarity, and Citation Parsing”**

Faculty of Humanities and Social Science

Tribhuvan University

Kirtipur, Kathmandu, Nepal

**Submitted by**

|  |  |
| --- | --- |
| Name: Anusha Shrestha | Name:Ujjwal Shrestha |
| Roll No.: 6-2-376-1-2021 | Roll No.: 6-2-376-16-2021 |

**Submitted to**

Advanced College of Engineering and Management

Department of Bachelor in Computer Application

Kalanki, Kathmandu

Table of Contents

[1. Introduction 1](#_Toc1)

[2. Problem Statement 2](#_Toc2)

[3. Objectives 2](#_Toc3)

[4. Scope and Limitation 2](#_Toc4)

[4.1 Scope 2](#_Toc5)

[4.2 Limitation 3](#_Toc6)

[5. Methodology 3](#_Toc7)

[5.1 Requirement Identification 3](#_Toc8)

[5.1.1 Study of existing system 4](#_Toc9)

[5.1.2. Requirement Analysis 4](#_Toc10)

[5.2. Feasibility Study 5](#_Toc11)

[5.2.1. Technical Feasibility 6](#_Toc12)

[5.2.2. Economic Feasibility 6](#_Toc13)

[5.2.3 Operational Feasibility 6](#_Toc14)

[5.3. High Level Design of the System 6](#_Toc15)

[5.3.1. Flow Chart of proposed System 6](#_Toc16)

[5.3.2. Description of Algorithms 7](#_Toc17)

[6. Gantt Chart 8](#_Toc18)

[6. 9](#_Toc19)

[7. References 10](#_Toc20)

**Table of Figures**

[Figure 1: Waterfall model of our Proposed System 3](#_Toc1)

[Figure 2: Use Case Diagram of our Proposed System 5](#_Toc2)

[Figure 3: Flowchart of our Proposed System 7](#_Toc3)

[Figure 4 Gantt chart 9](#_Toc4)

# 1. Introduction

It is really annoying to see someone else taking credit for your own work right? Have you ever thought that only if you could punish the person who stole your ideas or projects and presented them in a different way, slightly modifying and gaining the favor of everyone? Or at least you could prove to others that was your own real idea?

Plagiarism is derived from Latin word “plagiarius” which means “kidnapper,” who abducts the child. The word plagiarism entered the Oxford English dictionary in 1621. Plagiarism has been defined by the Encyclopedia Britannica as “the act of taking the writings of another person and passing them off as ones own.” It is an act of forgery, piracy, and fraud and is stated to be a serious crime of academia.[1] The act of copying someone else words, ideas, or work and passing them off as one's own without giving due credit is known as plagiarism which is a serious offensive crime. It is seen as unethical and a significant problem in the creative, professional, and academic domains. Intentional or unintentional plagiarism frequently takes the form of direct copying, poor paraphrasing, or incorrect source citation.

To combat this, plagiarism checkers use advanced database software to scan for matches between your text and existing texts. These tools are often used by universities to scan student assignments, and there are also commercial plagiarism checkers available for individuals who want to verify their own work before submitting it. However, many of these tools are either expensive or have limited features, leaving users in need of a more accessible solution.

Behind the scenes, plagiarism checkers crawl web content and index it, scanning your text for similarities against a database of existing content on the internet. Exact matches are highlighted using keyword analysis and can also identify non-exact matches (paraphrasing plagiarism).

This is where our system comes in. We aim to create a web app designed to detect plagiarism with easy implementation, a user-friendly interface, nearly accurate results, clear reports, and potential suggestions for improvement. Unlike traditional methods, our app will scan a variety of file formats, including .txt, .docx and .pdf, and identify both exact matches and subtle non-exact matches (such as paraphrasing), which are often harder to detect.

By developing this tool, we hope to help individuals protect their intellectual property and ensure proper credit is given where it’s due.

# 2. Problem Statement

In today's world, it has become increasingly common for individuals, particularly those in positions of power and authority, to take credit for the work of others.

**Rise of Digital Platforms:** The easy access to online content has made it simpler for individuals to copy, modify, and present others' work as their own.

**Impact on Creators:** Plagiarism diminishes the efforts of the true creators and affects their recognition and rewards.

**Need for Proper Citation:** Giving due credit through citations allows creators to share knowledge while respecting intellectual property.

**Evolving Tactics in Plagiarism:** Plagiarism methods, like paraphrasing and altering texts, have become more sophisticated, which complicates detection.

# 3. Objectives

The primary objective of this project is to detect plagiarism and ensure that the original creator’s work is rightfully recognized. The key objectives include:

* **Develop a Plagiarism Detection System**: Create an efficient system that can identify copied, minor changed or paraphrased content.
* **Enhance Citation Awareness**: Encourage proper citation practices by providing users with plagiarism reports and source references.
* **Provide a User-Friendly Interface**: Design an intuitive platform for users to upload and analyze documents with ease.

# 4. Scope and Limitation

Every tool has its own advantages and limitations. Some of the key ones are:

## **4.1 Scope**

* **Plagiarism Detection**: Identifies exact matches, minor edits, and paraphrased content.
* **Text-Based Comparison**: Compares uploaded documents with a reference database.
* **Admin Dashboard:** Allows management of users, documents, and system settings.
* **File Upload Support:**  Accepts multiple file formats (e.g., .txt, .pdf, .docx).
* **Threshold-Based Flagging:** Allows customization of similarity percentage to flag plagiarism.
* **Database Management:** Stores reference materials, past reports, and user submissions.

## **4.2 Limitation**

* **Limited Dataset:** The system can only compare with the database it has; unknown external sources won’t be detected.
* **Paraphrasing Complexity:** May struggle to detect highly advanced paraphrasing or AI-generated content.

# 5. Methodology

we are using the waterfall methodology in this project as it is a popular project management methodology that follows a sequential, linear approach to software development. In this methodology, each stage of the development process must be completed before moving onto the next stage. This methodology of software development is utilized due to the scope of the project and the available time frame. It is also well suited for projects that have a clear and well-defined requirements and specifications.

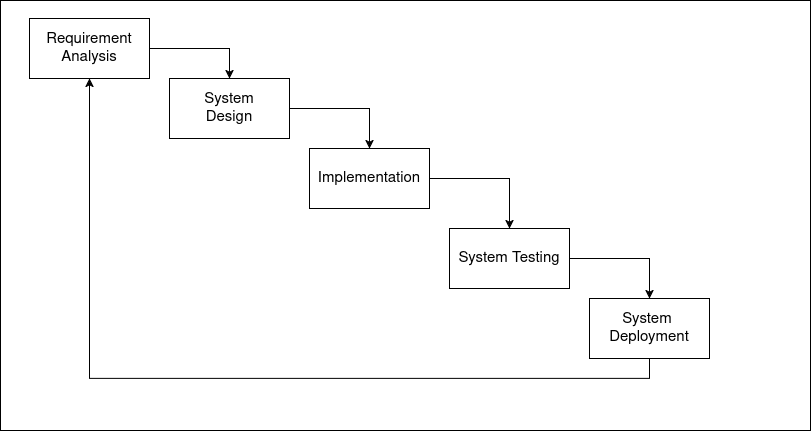


Figure 1: Waterfall model of our Proposed System

## 5.1 Requirement Identification

Requirement Identification is the process of gathering, analyzing, and defining what a system must do to meet user need. It is a critical phase in software development, ensuring that the final product aligns with expectations.

### 5.1.1 Study of existing system

For this project, we conducted a study on existing plagiarism checker apps and software to understand their strengths and weaknesses. We researched popular tools such as Turnitin, Grammarly, and Copyscape, testing their features like detecting exact text matches, paraphrased content, report generation, and overall user experience. While these tools were effective at detecting direct plagiarism, most struggled with paraphrasing. Additionally, many were costly and had slow processing times for larger documents. Our research highlighted the need for a more accurate, cost-effective plagiarism detection tool, particularly one that can identify paraphrased content. This insight will guide the development of our plagiarism checker.

### 5.1.2. Requirement Analysis

The requirements are to be collected before starting the project’s development life cycle. To design and develop a system, functional as well as non- functional requirements of the system have been studied.

**Functional requirements:** The different core functionality of this proposed project are:

* **User Interaction (Without Authentication)** :
  + Users can upload files (PDF, DOCX, or plain text) for plagiarism checking.
  + After analysis, a detailed plagiarism report is generated and can be viewed or downloaded.
* **Admin Panel :** 
  + Admins can manage and update the repository of documents and sources used for plagiarism comparison.
* **Plagiarism Detection Engine** – The system processes uploaded documents and compares them against a document repository using multiple strategies:
  + Sentence-BERT – for semantic similarity detection
  + Cosine Similarity – for similarity scoring between sentence embedding
  + Citation Parsing – for checking citation integrity and proper attribution
* **Report Generation** – After analysis, the system generates a detailed plagiarism report that includes:
  + **Overall Similarity Score :** A percentage indicating how much of the document is similar to existing sources.
  + **Highlighted Plagiarized Sections :** Matched or similar segments are visually marked for easy review.
  + **Matched Source Details :** Information on the original sources from which content appears to be plagiarized.

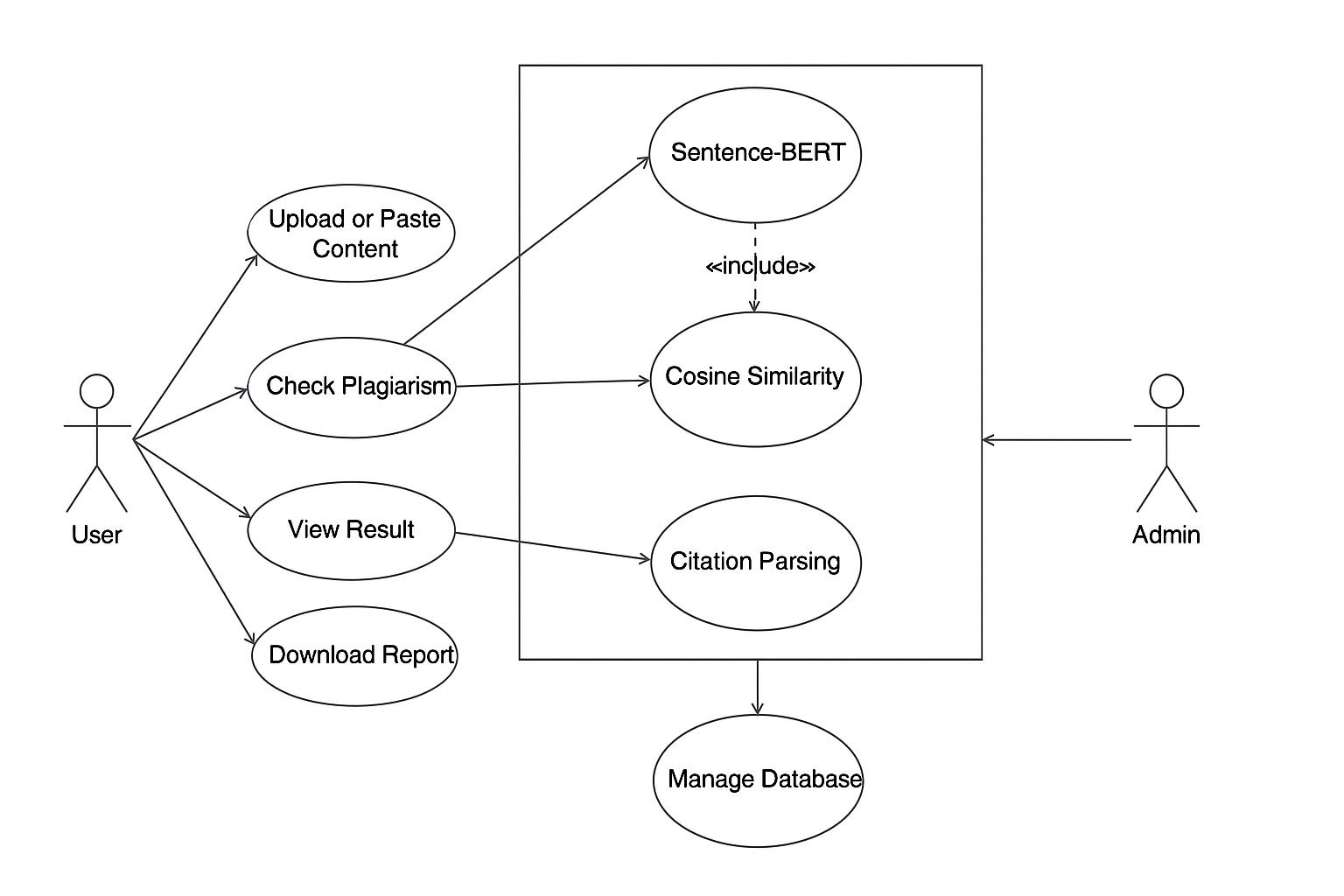


Figure 2: Use Case Diagram of our Proposed System

**Non-Functional Requirements:**

* **Fast Processing** – The system should quickly analyze large documents and return results efficiently.
* **Real-Time Response** – The system should generate plagiarism reports within seconds or minutes.
* **Simple User Interface** – The UI should be easy to use, even for non-technical users.
* **Error Handling** – The system should detect and handle errors like incorrect file formats or missing inputs.

## 5.2. Feasibility Study

Before starting the development of the plagiarism detection system, several feasibility aspects were evaluated to ensure the system could be built within the given time and meet all the requirements.

### **5.2.1. Technical Feasibility**

The system will be built using existing technologies such as core programming languages (e.g., Python) and libraries, with no need for advanced hardware. The algorithms will be designed to identify both exact matches and paraphrased content. Since the system doesn’t require sophisticated hardware, it can be developed and run on a personal computer. This ensures there are no significant technical hurdles, and the system can be developed on a personal computer without specialized devices.

### **5.2.2. Economic Feasibility**

The system will be developed using entirely open-source tools and libraries, so there are no costs for proprietary software. This eliminates the need for recurring expenses, apart from internet access for research purposes and accessing freely available tools and resources.

### **5.2.3 Operational Feasibility**

The plagiarism detection system will be a web-based application, which is accessible via any modern browser, meaning no specialized devices or hardware are needed to run it. The user interface will be designed to be simple, intuitive, and easy to navigate, with clear graphical icons to guide the user through the process of checking documents for plagiarism.

## 5.3. High Level Design of the System

The design of the plagiarism detection system is centered around providing an efficient and accurate tool for checking both exact and paraphrased content. The system will be a web-based application with a user-friendly interface and robust back-end algorithms for text comparison.

### 5.3.1. Flow Chart of proposed System

A flowchart is a diagram that visually represents a process, using different symbols to denote operations, decisions, and process flow. The flowchart of our proposed system is given below in figure:

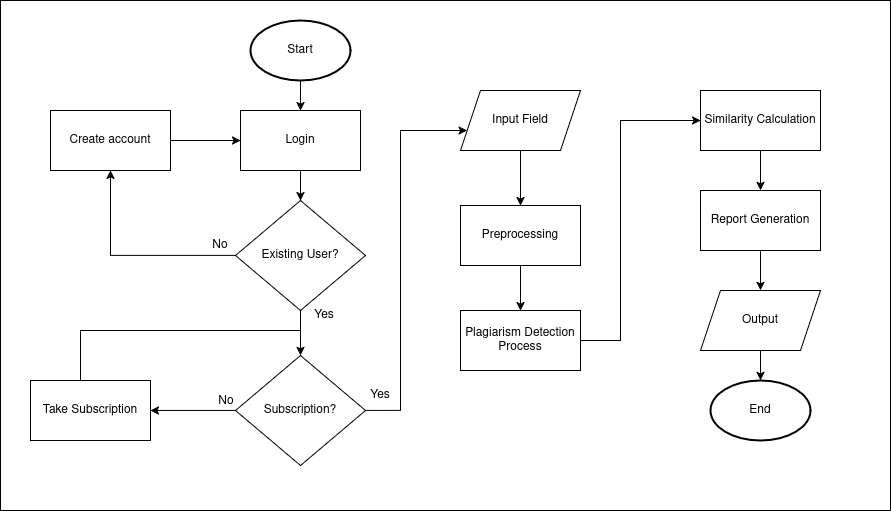


Figure 4: Flowchart of our Proposed System

### 5.3.2. Description of Algorithms

The list of algorithm that we used in this proposed system are listed below:

1. **Rabin-Karp Algorithm:** Used for efficient string matching to detect exact matches of text in the uploaded document against a predefined set of sources. It working mechanism are:
   * 1. **Hash the Pattern:** Compute a hash value for the pattern string.
     2. **Hash the Text:** Compute hash values for sub-strings of the text that are the same length as the pattern.
     3. **Compare Hashes:** If the hash of a text sub-string matches the pattern’s hash, perform a direct string comparison to confirm a match.
     4. **Slide the Window:** Use a rolling hash to efficiently update the hash value when sliding over the text.
2. **Sentence-BERT Algorithm:** This algorithm will focus on semantic similarity by analyzing the meaning of sentences to identify paraphrased content. It will convert sentences into vector embedding and compare them for similarities.SBERT follows a three-step approach:
   * 1. **Input Encoding:** Converts text into dense vector representations (embedding).
     2. **Embedding Comparison:** Uses cosine similarity or other distance metrics to compare embedding.
     3. **Similarity Score Calculation:** Determines how semantically similar two sentences are.
3. **Citation Analysis Algorithm:** Citation analysis is a technique used to evaluate how references and citations are used within a document. It helps detect source-based plagiarism by analyzing if the cited sources are properly referenced or manipulated. The algorithm follows a structured process:

* Extract Citations: Identify in-text citations and references from the document.
* Match Citations with References: Check if all in-text citations have a corresponding reference in the bibliography.
* Validate Source Authenticity
  + Compare citations with real sources (Google Scholar, IEEE, etc.).
  + Detect fake or incorrect citations
* Check for Missing Citations: Identify content that should be cited but lacks a reference.
* Analyze Citation Frequency & Patterns: Detect excessive self-citations or manipulation.
* Generate a Report
  + Flag missing, incorrect, or manipulated citations.
  + Assign a plagiarism risk score based on citation integrity.

# **6. Gantt Chart**

Gantt chart is used to plan and track the progress of the project by breaking down the project's activities or tasks into smaller, more manageable sub-tasks, and then assigning a start date, end date, and duration to each one.

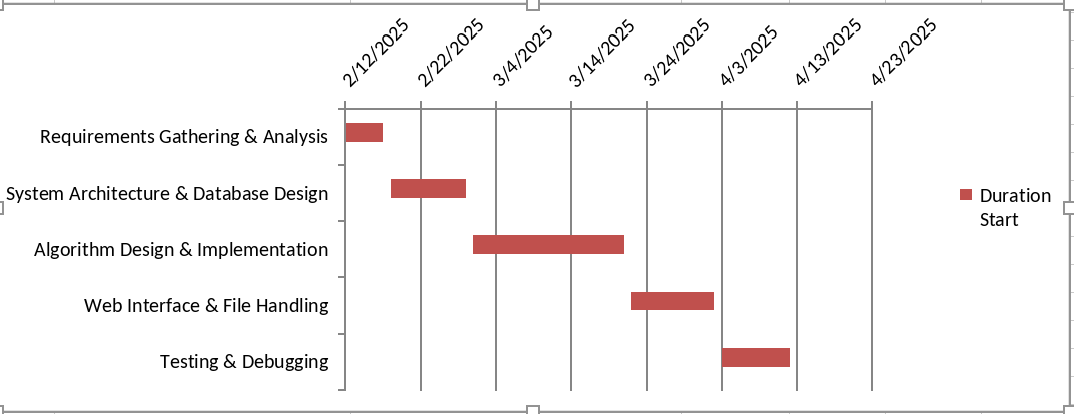


Figure 5 Gantt chart

The project timeline begins with Requirements Gathering & Analysis (5 days) to identify the system’s goals and features, followed by System Architecture & Database Design (10 days) to plan the system structure and create the necessary database schema. Next, Algorithm Design & Implementation (20 days) focuses on developing the core plagiarism detection algorithms, such as Rabin-Karp, AST, and Sentence-BERT. After that, Web Interface & File Handling (11 days) involves creating the user interface for document upload and result display, while Testing & Debugging (9 days) ensures the system functions correctly and is free of bugs. Finally, Documentation (5 days) will be completed to provide thorough project documentation.

# 6.

# 7. References

[1]Dr. Ish Kumar Dhammi and Rehan Ul Haq, “What is plagiarism and how to avoid it?,” p. 50, Dec. 2016, [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC5122250/

Appendices and screenshots

