Tribhuvan University

Faculty of Humanities and Social Sciences

A PROJECT REPORT ON
Plagiarism Detection System

Submitted to:

Department of Computer Application

Nepal Kasthamandap College, Kalanki

In partial fulfillment of the requirements for the Bachelor in Computer Application

Submitted by:

Ananta Thapa [Reg no:6-2-144-01-2021]

July 2025

Under the Supervision of:

Er. Sujan Poudel

Tribhuvan University

Faculty of Humanities and Social Sciences

Nepal Kasthamandap College

Kalanki, Kathmandu

Bachelor in Computer Applications (BCA)

Supervisor's Recommendation

I hereby recommend that this project proposal prepared under my supervision by **Ananta Thapa** entitled "**Plagiarism Detection System**" in partial fulfillment of the requirement for the degree of Bachelor in Computer Application be forwarded for further approval.

Er. Sujan Poudel
Project Supervisor
BCA Department
Nepal Kasthamandap College

Tribhuvan University
Faculty of Humanities and Social Sciences

Nepal Kasthamandap College
Kalanki, Kathmandu
Bachelor in Computer Applications (BCA)

Letter of Approval

This is to certify that this project proposal prepared by **Ananta Thapa** entitled "**Plagiarism Detection System**" in partial fulfillment of the requirements for the degree of Bachelor in Computer Application has been evaluated and approved.

Er. Sujan Poudel
Supervisor
Nepal Kasthamandap College

Er. Sujan Poudel
Program Coordinator
Nepal Kasthamandap College

Roshan Poudel
Internal Examiner
BCA Department

Mr. Aanand KC
External Examiner
RR Campus

Abstract

The **Plagiarism Detection System** is a tool that helps ensure academic and professional integrity by identifying instances of plagiarism. Developed using Python and Natural Language Toolkit (NLTK), the system compares text documents and highlights similarities. This project aims to support educational institutions and individuals in maintaining originality in written works. The frontend is built using HTML, CSS, and JavaScript, while the backend is powered by Python and SQLite. Users can upload DOCX, PDF, or TXT files, and the system processes them to detect copied content. This tool fosters ethical writing practices by generating detailed reports and improving the quality of submitted materials.

Keywords: Plagiarism Detection System, File Comparison, Text Comparison, User Dashboard, Admin Dashboard, User Profile, Plagiarism Comparison.

Acknowledgement

I would like to express my heartfelt gratitude to my supervisor **Er. Sujan Poudel Sir** for guiding me throughout the course of this project. His continuous support and encouragement were vital to the completion of this proposal. I would also like to thank the Department of Computer Application at **Nepal Kasthamandap College** for their resources and environment. Lastly, my appreciation goes to my family and friends for their unwavering support throughout this academic journey.

Table of Contents

List of Figures

List of Tables

List of Listings

List of Abbreviations

Abbreviation	Description
NLTK	Natural Language Toolkit
NLP	Natural Language Processing
CSS	Cascading Style Sheet
DBMS	Database Management System
HTML	Hypertext Markup Language
PDS	Plagiarism Detection System
SQLite	Structured Query Language

Chapter-1: Introduction

1.1. Introduction

The majority of computer science courses require students to complete programming tasks as part of their coursework. Students are typically required to work independently because these projects can have a significant impact on their grades. Unfortunately, there is probably no way to stop students from working together when creating their programs.

Plagiarism is the act of stealing another person's intellectual property, including their literary work, creative output, and ideas. It is a clear violation of academic integrity, honesty, and originality to claim these as one's own without giving proper credit and approval. Whether it is writing essays, research papers, or artistic works, the technique has undoubtedly become more common in a variety of contexts. This is a fact that we often ignore.

The Plagiarism Detection System is a sophisticated web-based tool designed to detect and mark instances of plagiarism in written materials. The main objective of this project is to give users a trustworthy tool to verify the academic integrity and originality of their work. The Natural Language Toolkit (NLTK) toolkit and Python are used by this system to analyze text and find similarities with pre-existing sources. HTML, CSS, and JavaScript power the system's client-side operations by offering an easy-to-use interface, while Python and NLTK handle the intricate text analysis and comparison tasks on the server side. This combination guarantees that users have access to a powerful and comprehensive plagiarism detection tool, making it simple to verify the originality of content and upholding high standards of academic and professional integrity.

1.2. Problem Statement

Detecting plagiarism is essential for maintaining the integrity and uniqueness of professional and academic work. The following are the main problems that this system seeks to solve:

- Lack of system for precise detection of plagiarized content.
- Detection of source of plagiarism in the document.
- Lack of keeping search history of the person/users.
- Lack of system for uploading and comparing two documents.

The suggested Plagiarism Detection System seeks to resolve these problems in order to improve plagiarism detection's precision, effectiveness, and dependability while encouraging moral behavior and preserving the integrity of written material.

1.3. Objectives

The main goal of the Plagiarism Detection System is to detect and mark instances of plagiarism in order to guarantee the uniqueness and integrity of text works. The following are the main goals:

- To accurately identify plagiarism with detailed and reliable results.
- To effortlessly support lots of types of files, including TXT, DOCX, and PDF.
- To protect user information and documents so as to ensure security and privacy.

The Plagiarism Detection System hopes to accomplish these goals in order to offer a strong and all-inclusive instrument for preserving the authenticity and uniqueness of written material.

1.4. Scope And Limitations

1.4.1. Scope

The Plagiarism Detection System encompasses the following functionalities:

- Comparison of submitted text or documents against an internal, user-managed reference database.
- Support for input from .txt, .docx, and .pdf file formats.
- Enabling direct similarity comparison between two user-provided texts or files.
- Generation of downloadable PDF reports that summarize plagiarism detection results.
- Implementation of basic user authentication for system access control.

1.4.2. Limitations

The Plagiarism Detection System is subject to the following constraints:

- Detection accuracy is primarily limited to direct textual matches and may not identify sophisticated paraphrasing or non-English content effectively.
- Comparison is restricted to the internal, user-managed reference database, without internet-wide search capabilities.
- Scalability for processing very large documents or extensive reference libraries is limited, potentially impacting performance and multi-user deployment.

- The current feature set does not include in-browser highlighting of matches or automated citation generation.
- Security measures are foundational and require enhancement for public or high-security environments.

1.5. Development Methodology

Agile methodology is a collaborative and flexible approach to software development and project management. It emphasizes agile development, transparent communication, and flexibility in response to changing requirements. The main objective of Agile is to prioritize customer satisfaction while producing high-quality products in an effective way. [12]

The sequential phases of the project in Agile methodology are:

- Requirement Gathering: We collect all potential needs for the plagiarism detection system during the requirement gathering stage. This involves knowing the requirements of administrators, users, and other interested parties. Important features like database management, document upload, similarity identification, text analysis, user authentication, and API integration are noted. Research and other similar system are used to collect thorough needs.
- Requirement Analysis: The collected requirements are examined and documented in a thorough specification document during the requirement analysis process. A feasibility analysis will be carried out to determine whether the needs are feasible, achievable, and within the parameters. Priorities are set for requirements according to their significance and effect on the system.
- System Design: A system design is created during the system design phase after the required specifications have been examined. Both hardware and software components are part of the system architecture. The system's general architecture is established, including its components, data flow, and integration points. To direct the development process, models, diagrams, and design papers are created.
- System Development: On the basis of the design specifications, the system is developed in manageable, small components or modules during the system implementation phase. The process of developing and testing each unit for functioning is known as unit testing. After that, the components are combined to create a coherent system.
- Testing: The system is thoroughly tested during the testing phase to make sure it functions as planned. While functional testing confirms that the system satisfies the requirements, non-functional testing (such as performance, security, and usability testing) guarantees

the system's stability and reliability. Any flaws or weaknesses found during testing are located and corrected.

- Deployment: The system is ready to be deployed in the single user environment when testing is finished. The required network setups, servers, and databases will be made in local host as the system will not be made public. To make sure the system is ready for usage, last-minute checks are made.
- Review: After the system is ready and deployed, another iteration is launched to improvise the system according to the user demand. Another iteration can also be occurred for adding of new features and function.

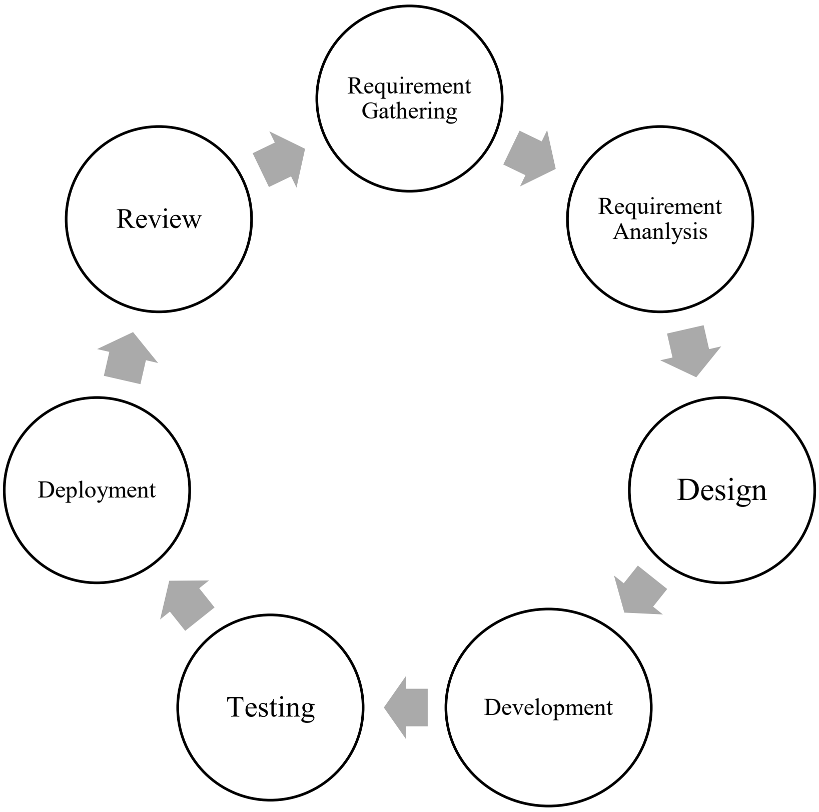


Figure 3.1 Agile Methodology

1.6. Report Organization

Introduction:

This chapter deals with the introduction of the system with its objectives and limitations along with the reason the system is made.

Background study and Literature Review:

This chapter summarizes the work that has been conducted in the field of data mining and also describes the features of some existing applications related to Plagiarism Detection System.

System Analysis and Design:

This chapter focuses on the different requirement of the system, which describes about the functional, non-functional, feasibility analysis, Entity Relational Diagram, Data Flow Diagram, design of the system with system architecture, database schema, and interface design.

Implementation and testing

This chapter emphasizes tools used in system development, implementing details and results of test performed.

Conclusion and Future Recommendation

This chapter highlights brief summary of lessons learnt, the outcome and conclusion of the whole project and explain what have been done and what future improvements could be done.

Chapter-2: Background Study and Literature Review

2.1.Background Study

The purpose of plagiarism detection systems is to find instances of copied material in professional and scholarly writings. With the help of developments in machine learning and natural language processing (NLP), these systems have undergone substantial development throughout time, increasing in precision and effectiveness. Modern systems use semantic analysis, which enables the identification of more complex kinds of plagiarism, like paraphrasing and obfuscation, in contrast to traditional methods that depended on basic text-matching algorithms. [1]

Important elements of systems for detecting plagiarism include:

i. Text-based detection: Examining the text to see if it resembles any previously published papers. This method entails reading the material to find parallels with other papers that have been published. It uses techniques including precise matching, fuzzy matching and stylometry to identify paraphrased and verbatim information, as well as writing style inconsistencies.

ii. Document-based detection: Finding copied passages by comparing full documents. This method compares whole papers in order to identify copied passages. For increased accuracy, it incorporates segmented analysis, which divides the document into smaller parts, and whole document comparison.

iii. Image-based detection: Identifying instances of plagiarism in schematics or pictures. This technique detects instances of plagiarism in images, schematics, and other visual materials. To identify unities in visual characteristics like color, texture, and shape, methods including image recognition, metadata analysis, and content-based image retrieval (CBIR) are used.

iv. Machine learning models: Algorithms are used to gradually enhance detection by identifying patterns. By spotting patterns in text and images, algorithms improve detection. These algorithms learn from fresh instances of plagiarism over time, identify pertinent features, and train on labeled data.

In order to preserve academic integrity and guarantee originality in scholarly writings, these mechanisms are essential. By automating the detection process, they lessen the workload for reviewers and educators while assisting institutions in maintaining high standards of integrity.

2.2.Literature Review

Plagiarism is becoming a serious problem for intellectual community. The detection of plagiarism at various levels is a major issue. The complexity of the problem increases when we are finding the plagiarism in the source codes that may be in the same language or they have been transformed into other languages. This type of plagiarism is found not only in the academic works but also in the industries dealing with software designing. The major issue with the source code plagiarism is that different programming languages may have different syntax. In this paper the authors will explain various techniques and algorithms to discover the plagiarism in source code. So, organization or academic institution can simply discover plagiarism in source code using these techniques. The authors will differentiate among these given techniques of plagiarism to discover how one technique is conflicting with the other. [2]

The lack of an evaluation framework is a serious problem for every empirical research field. In the case of plagiarism detection this shortcoming has recently been addressed for the first time in the context of our benchmarking workshop. This paper presents the evaluation framework developed in the course of the workshop. But before going into details, we survey the state of the art in evaluating plagiarism detection, which has not been studied systematically until now. [3]

Current research in the field of automatic plagiarism detection for text documents focuses on algorithms that compare plagiarized documents against potential original documents. Though these approaches perform well in identifying copied or even modified passages, they assume a closed world: a reference collection must be given against which a plagiarized document can be compared. [4]

The rise of Artificial Intelligence (AI) technology and its impact on education has been a topic of growing concern in recent years. The new generation AI systems such as chatbots have become more accessible on the Internet and stronger in terms of capabilities. The use of chatbots, particularly ChatGPT, for generating academic essays at schools and colleges has sparked fears among scholars. This study aims to explore the originality of contents produced by one of the most popular AI chatbots, ChatGPT. Moreover, ChatGPT was asked to verify if the essays were generated by itself, as an additional measure of plagiarism check, and it showed superior performance compared to the traditional plagiarism-detection tools. The paper discusses the need for institutions to consider appropriate measures to mitigate potential plagiarism issues and advise on the ongoing debate surrounding the impact of AI technology on education. Further implications are discussed in the paper. [5]

Being a growing problem, plagiarism is generally defined as "literary theft" and "academic dishonesty" in the literature, and it really has to be well-informed on this topic to prevent the problem and stick to the ethical principles. This paper presents a survey on plagiarism detection systems, a summary of several plagiarism types, techniques, and algorithms is provided. Common feature of deferent detection systems is described. At the end of this paper authors propose a web enabled system to detect plagiarism in documents, code and images, also this system could be used in E-Learning, E-Journal, and E-Business. [6]

Research is an original and systematic investigation undertaken to discover new facts and Information about a phenomenon. This paper is discussed about the plagiarism, Plagiarism Checker and Plagiarism detection tools from check research work through software like, Turnitin, Ithenticate, Plagiarism Checker, Viper, Dupli checker, Copy leaks, Paperrater, Plagium, Plagiarisma, Plagscan etc. [7]

The plagiarism in student assignments is a widespread and growing problem in the academic process. The traditional manual detection of this kind of plagiarism by human is difficult, not accurate, and time-consuming process. This project aims to create an online web-based plagiarism-detection system that can help university teachers to make a better judgment for student's work. [8]

Plagiarism is widely spreading nowadays, because of antiplagiarism software should be used in order to maintain the academic integrity of the individuals. The academic community should educate themselves about the concept of plagiarism and avoiding plagiarism. Furthermore, which will help them to enhance the quality and acceptance level of their publication. We are in Information age and there is lots of information available over internet. Most of the information over internet is produced as duplicate by the different authors. In order to maintain credentials of the original work of the author and improve the quality of research, plagiarism checking software may be useful. With help of plagiarism checking software, we can avoid the duplicity in research and academic writings. This paper gives an overview of various effective plagiarism detection methods that have been used for plagiarism detection to identify that how much work is original and how much work is illegal and copied from others' original works. [9]

In academia, plagiarism is certainly not an emerging concern, but it became of a greater magnitude with the popularization of the Internet and the ease of access to a worldwide source of content, rendering human-only intervention insufficient. Despite that, plagiarism is far from being an unaddressed problem, as computer-assisted plagiarism detection is currently an active area of research that falls within the field of Information Retrieval (IR) and Natural Language Processing (NLP). Many software solutions emerged to help fulfil this task, and this paper presents an overview of plagiarism detection systems for use in Arabic, French, and English academic and educational settings. The comparison was held between eight systems and was performed with respect to their features, usability, technical aspects, as well as their performance in detecting three levels of obfuscation from different sources: verbatim, paraphrase, and cross-language plagiarism. An in-depth examination of technical forms of plagiarism was also performed in the context of this study. In addition, a survey of plagiarism typologies and classifications proposed by different authors is provided. [10]

Plagiarism is a rising issue in academics as increasing resources constantly, and it plays a high impact on student's performance and quality of education. Several studies have been carried out to mitigate the issue. This study provides a comprehensive analysis of the previous research, plagiarism detection approaches, existing plagiarism detection tools, type of the tools, features of popular plagiarism detection tools and the challenges in plagiarism detection. Plagiarism detection can be divided into source code and natural language plagiarism detection. Natural language plagiarism detection tools can be categorized based on the mode of detection, type of application, mode of services and languages. This study analyses existing plagiarism detection approaches and previous researchers' used approaches. In addition, it provides an overview of popular plagiarism detection tools, their features and the challenges when using them. Further, the study can support to develop an effective approach and tool to control the issue of efficiency in future. Although several tools are available for the plagiarism detection process, none of them is effective in accuracy and efficiency. [11]

Chapter-3: System analysis and design

3.1 System Analysis

The Plagiarism Detection System's system analysis entails analyzing current plagiarism detection techniques, comprehending user needs, and locating technological solutions. The system uses Python and the Natural Language Toolkit (NLTK) package to provide a precise and effective way to identify plagiarism. Analyzing present systems to find weaknesses and potential areas for development, assessing user needs for comprehensive reporting and simplicity of use, and making sure the system is safe and scalable are all part of the process. Both functional and non-functional needs are addressed in the analysis, which aids in the creation of a reliable system that efficiently detects plagiarism while preserving data integrity and offering an intuitive user interface.

3.1.1 Requirement Analysis

Project has the following functional and non-functional requirements.

i. Functional Requirement

- Text Analysis: The system will carefully examine and compare text documents to accurately identify instances of plagiarism.
- User authentication: Users will be able to access the platform through a secure login process, ensuring their data is protected.
- Document Upload: Users can upload documents in various formats, such as PDF, DOCX, and TXT, for plagiarism checks.
- Similarity Detection: The system will recognize and highlight similar text passages between the uploaded document and existing sources.
- Database Management: The system will maintain a collection of text documents for comparison purposes.

ii. Non-Functional Requirement

- Performance: The system will be set up to handle and analyze documents quickly so that users can get results on time.
- Usability: All users should be able to easily navigate and utilize the system thanks to its user-friendly and intuitive interface.
- Reliability: Gaining users' trust and confidence we will make it so that the system will consistently delivering correct results.
- Compliance: It is an absolute must to follow all applicable laws and guidelines pertaining to data privacy and intellectual property rights.

Use-Case Diagram

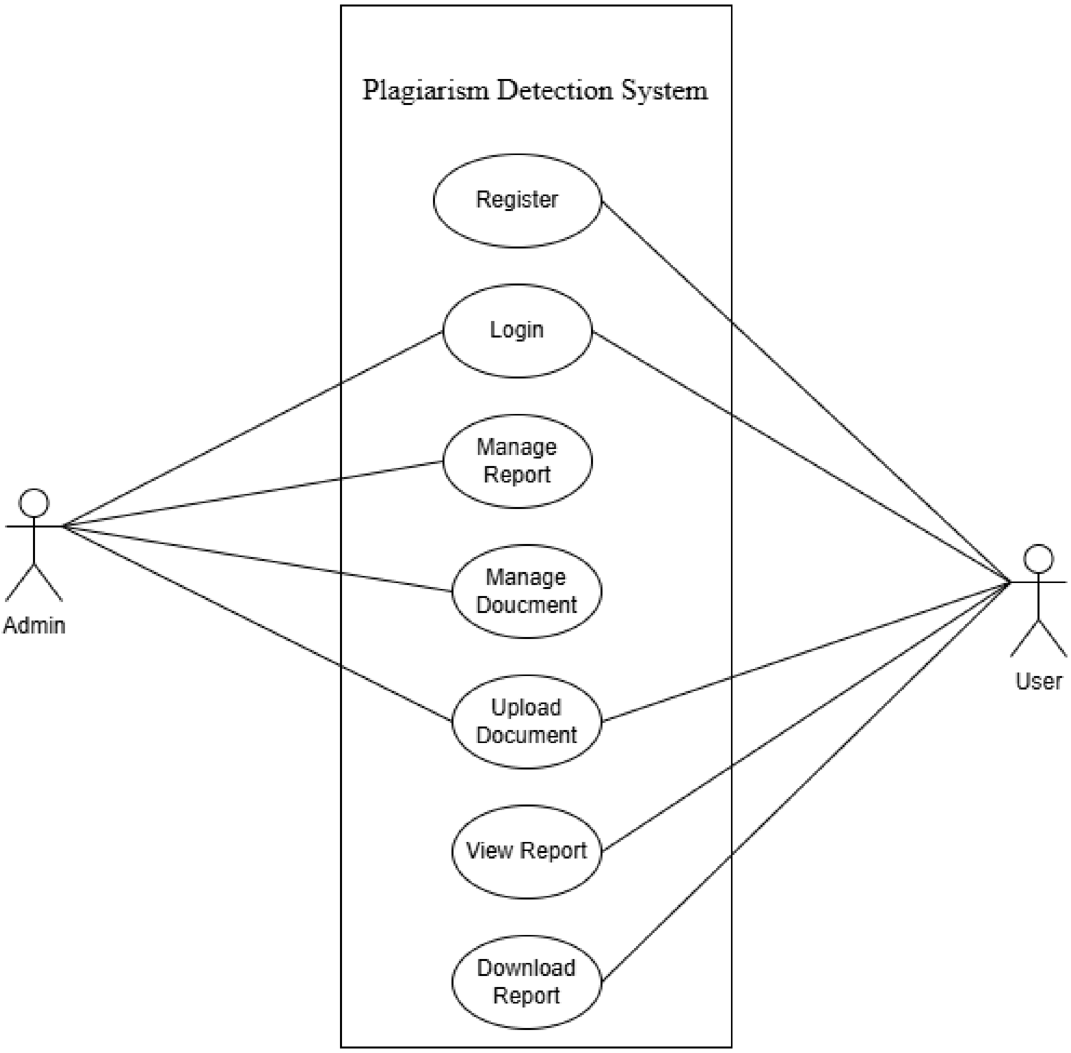


Figure 3.2 Use-Case Diagram for Plagiarism Detection System

The Plagiarism Detection System's use case diagram shows how the two main actors the administrator and the user interact with one another. Users have the ability to sign up, log in, upload documents for plagiarism detection, and examine reports that are created. Admins are also responsible for monitoring system performance, maintaining user accounts, and supervising document uploads. All of these interactions are managed by the central Plagiarism Detection System, which also processes document uploads, does plagiarism checks, and produces comprehensive results. In order to guarantee effective and precise plagiarism detection, this diagram offers a thorough overview of the system's functionality and the information flow between various components by precisely describing the roles and interactions of users and administrators.

3.1.2 Feasibility Analysis

a) Economic Feasibility

The Plagiarism Detection System's economic feasibility focuses on evaluating the project's financial sustainability. By utilizing open-source technologies like Python and the Natural Language Toolkit (NLTK), the solution eliminates the requirement for a substantial initial investment in proprietary software. The only required cost will be cost of maintaining a server which will not be problem as we will be hosting this system locally. Because the advantages of putting this system in place like increased academic integrity and fewer cases of plagiarism outweigh the disadvantages, the project is financially viable.

b) Technical Feasibility

The Plagiarism Detection System's technical feasibility study looks at how feasible it is to execute the project using the technology now in use. In order to construct the system, Python, NLTK, HTML, CSS, PHP, and SQLite will be used. In web development and natural language processing, these are popular and well supported technologies. Development tools and servers are among the easily accessible hardware and software requirements. It is also technically possible because we have the technological know-how needed to design, implement, and maintain the system.

Table 3.1: Technical Feasibility Study Table

Technologies Used	Hardware Used	Software Used
HTML	Laptop	Python Libraries
CSS	-	Visual Studio Code
JavaScript	-	Django server
SQLite	-	Draw.io
Python	-	-

c) Operational Feasibility

Operational viability evaluates how well the Plagiarism Detection System integrates with current processes and how well users accept it. Teachers, students, and institutions can upload and check documents for plagiarism with ease because to the system's user-friendly design. Users save time and effort by using the automated procedure to expedite manual plagiarism checks. Operationally, the system is practicable due to its user-centric design and the observable advantages it provides.

d) Schedule Feasibility

Table 3.2: Gantt Chart

Task/Date	16th February	21st April	26th May	3rd June	20th July
Planning	■				
Requirement gathering	■	■			
Sprint 1-Basic UI and DB Setup		■	■		
Testing Sprint 1		■	■		
Sprint 2- Text Comparison Module			■		
Testing Sprint 2			■	■	
Sprint 3-Reports and UI Enhancement			■	■	
Testing Sprint 3				■	■
Deployment					■
Documentation	■	■	■	■	■

A Gantt chart is a bar chart that illustrates a project schedule. In this Gantt chart, each row represents a task involved in the development of the Plagiarism Detection System. The start date and end date of each task are shown, as well as the duration. The Gantt chart visualizes the timeline for each task and shows the overlap between the separate phases of the project.

3.1.3 Data Modeling (ER-Diagram)

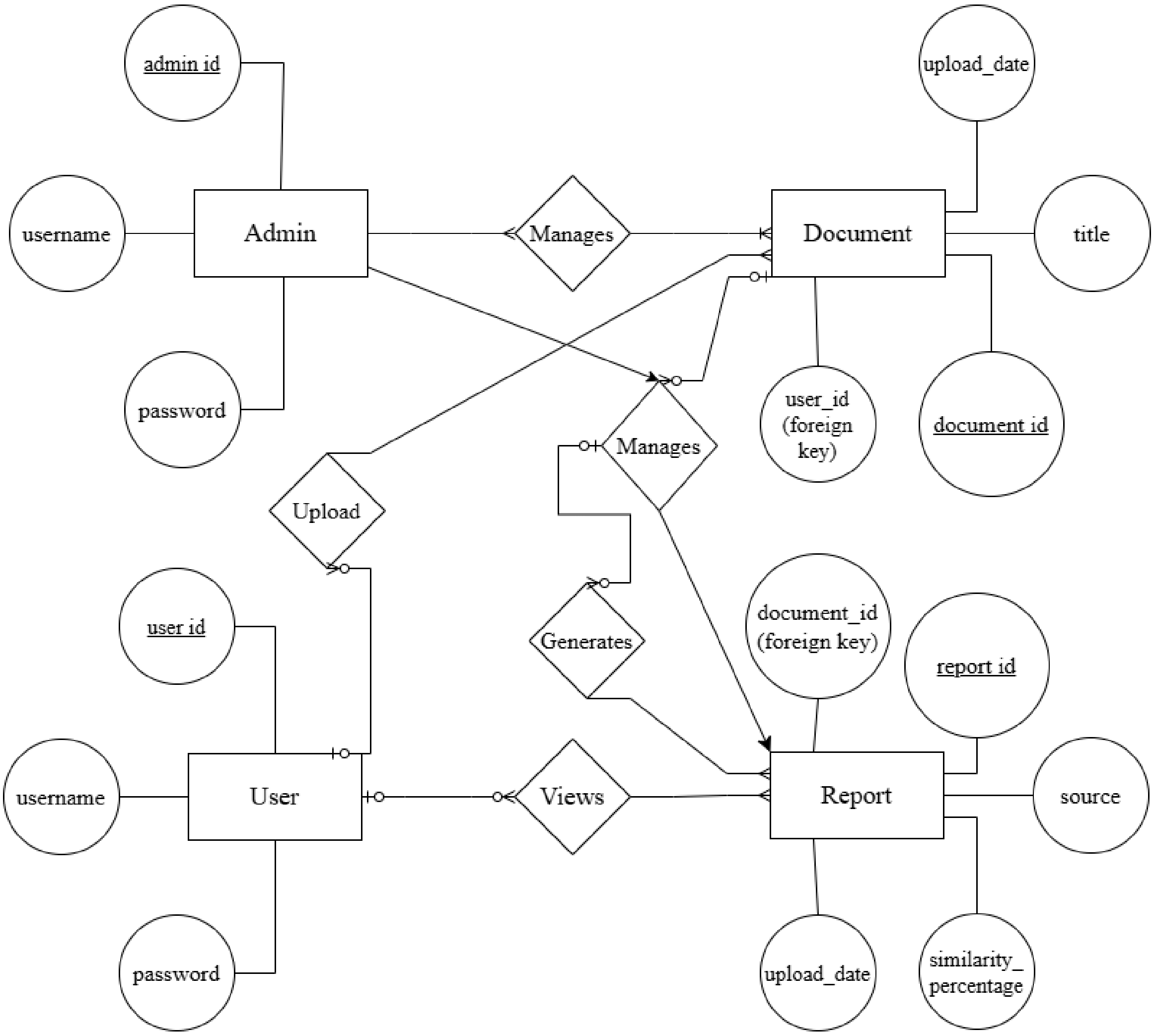


Figure 3.3 ER Diagram for Plagiarism Detection System

This ER diagram shows a system in which users, administrators, documents, and reports are all related by a number of relationships. Admins have hierarchical control over both users and documents. In turn, users can view reports and upload documents, demonstrating their interactive role in the system. Any document that a user upload can produce several reports, which users can then read. The database structure and the interactions between various entities in the system are clearly understood thanks to this visual depiction, which emphasizes the characteristics of each entity and their relationships.

3.1.4 Process Modeling

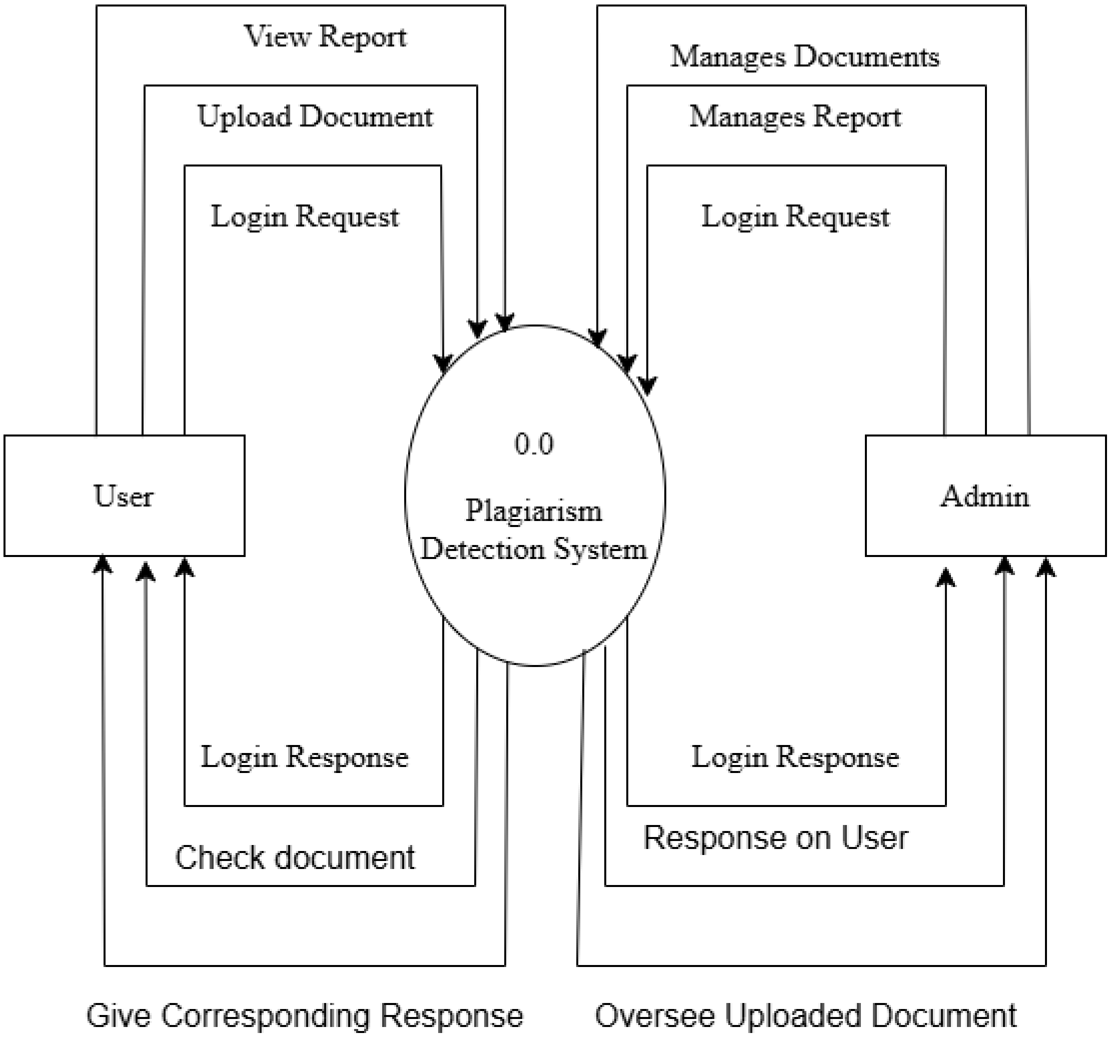


Figure 3.4 DFD (Level-0) for Plagiarism Detection System

The Plagiarism Detection System's Level 0 Data Flow Diagram (DFD) shows how administrators and users communicate with the main system. While administrators log in, control user accounts, supervise document uploads, and keep an eye on system performance, users log in, upload documents, and view plagiarism reports. After processing these actions, the system provides pertinent answers, such as document status updates, login confirmations, and thorough plagiarism reports. The information flow between users, administrators, and the system is depicted in this high-level overview to guarantee effective and accurate plagiarism detection.

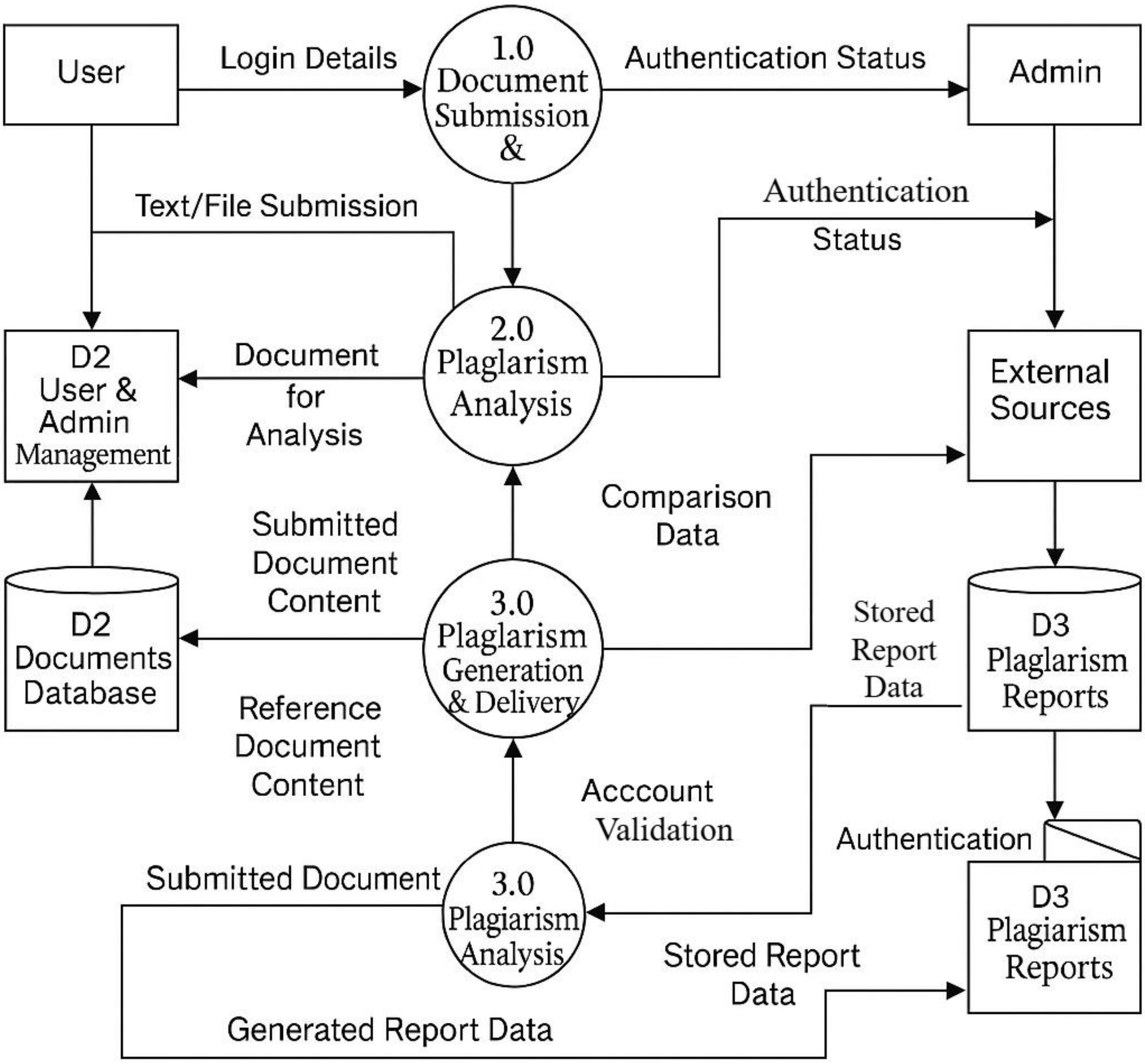


Figure 3.5 DFD (Level-1) for Plagiarism Detection System

This Level 1 Data Flow Diagram provides a high-level overview of the major processes, external entities, and data stores involved in the Plagiarism Detection System, along with the key data flows that connect them. The system interacts with three External Entities: the User, who submits documents and retrieves reports; the Admin, who manages system configurations and oversees user accounts; and External Sources, representing external databases and web content used for plagiarism comparison. Within the system, there are four key Processes. 1.0 Document Submission & Management handles all incoming documents from Users and Admins, storing them in the D2 Documents Database and preparing them for analysis. 2.0 Plagiarism Analysis Core then takes these documents, retrieves Reference Documents from the D2 Documents Database and Comparison Data from External Sources, performing the actual plagiarism detection and generating Raw Plagiarism Results. These results are passed to 3.0 Report Generation & Delivery, which formats, stores, and delivers the final Plagiarism Report to Users and Admins, utilizing and updating the D3 Plagiarism Reports data store. Finally, 4.0 User & Admin Management manages all user and administrator accounts, handling Login Details, Registration Details, and Account Updates and interacting with the D1 User Accounts data store to manage user credentials and authentication status. This interconnected system ensures a comprehensive workflow from document submission to plagiarism report delivery and user administration.

3.2 System Design

To realize the different functional requirement of the system in graphical form, different design diagram of the system has been prepared which are as follow:

3.2.1. Architecture Design

For the system, three tier architecture is used which includes user interface, web server and database. In architectural design, basic structure of the system is shown:

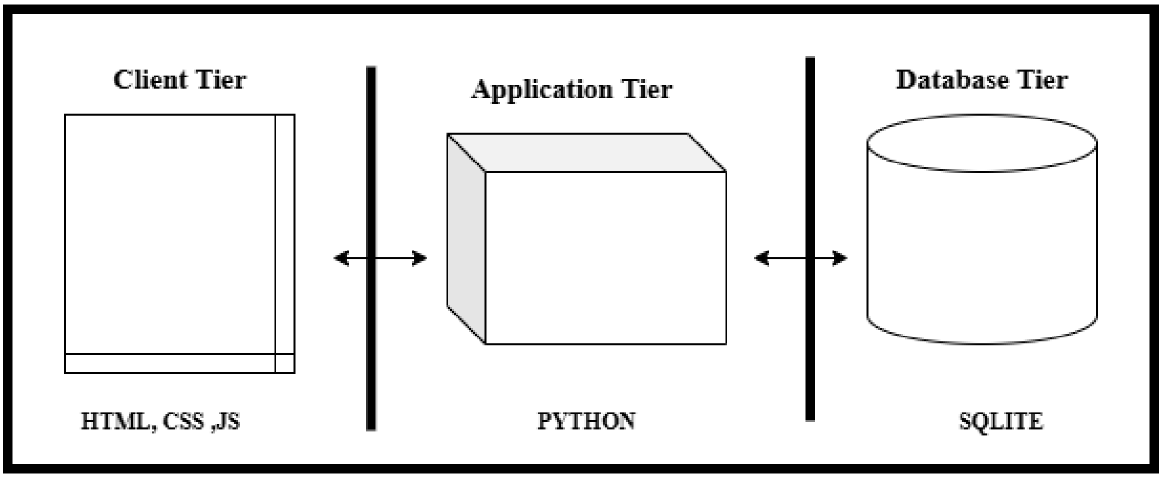


Figure 3.6 Architecture Design

3.2.2. Flow Chart

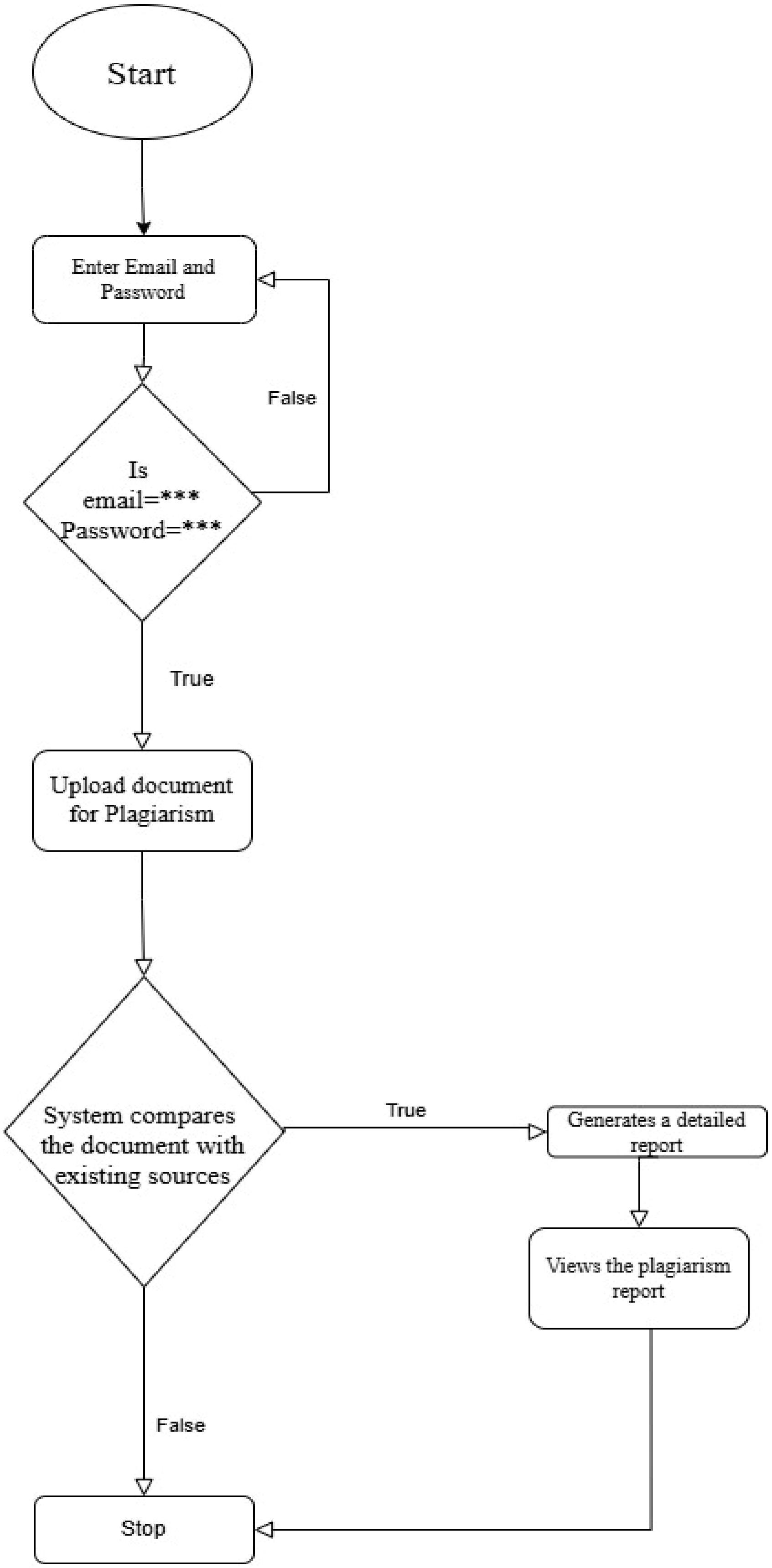

Start
Enter Email and Password
False
Is email=*** Password=***
True
Upload document for Plagiarism
System compares the document with existing sources
True
Generates a detailed report
Views the plagiarism report
False
Stop

Figure 3.7 Flow Chart for User

Starting with the user providing their email address and password, the flowchart explains how to check a paper for plagiarism. In the event that the credentials are accurate, the user uploads the document for verification. In order to identify plagiarism, the algorithm then analyzes the document with already published sources. The user can examine the comprehensive report that the system generates if plagiarism is detected. If plagiarism is not found, the procedure is over. This flowchart ensures that users understand the necessary procedures involved by providing a clear and simple sequence of activities and decisions required to check a document for plagiarism.

1. Start: The process begins.
2. Enter Email and Password: The user is prompted to enter an email and password.
3. Is email=* Password=*: The system checks if the entered email and password are correct.

- If the email and password are incorrect (False), the user is prompted to re-enter the email and password.
- If the email and password are correct (True), the process proceeds to the next step.

4. Upload document for Plagiarism: The user uploads the document that needs to be checked for plagiarism.
5. System compares the document with existing sources: The system compares the uploaded document with existing sources to check for plagiarism.

- If plagiarism is detected (True), the system generates a detailed report.
- If no plagiarism is detected (False), the process stops.

6. Generates a detailed report: The system generates a detailed plagiarism report.
7. Views the plagiarism report: The user views the generated plagiarism report.
8. Stop: The process ends.

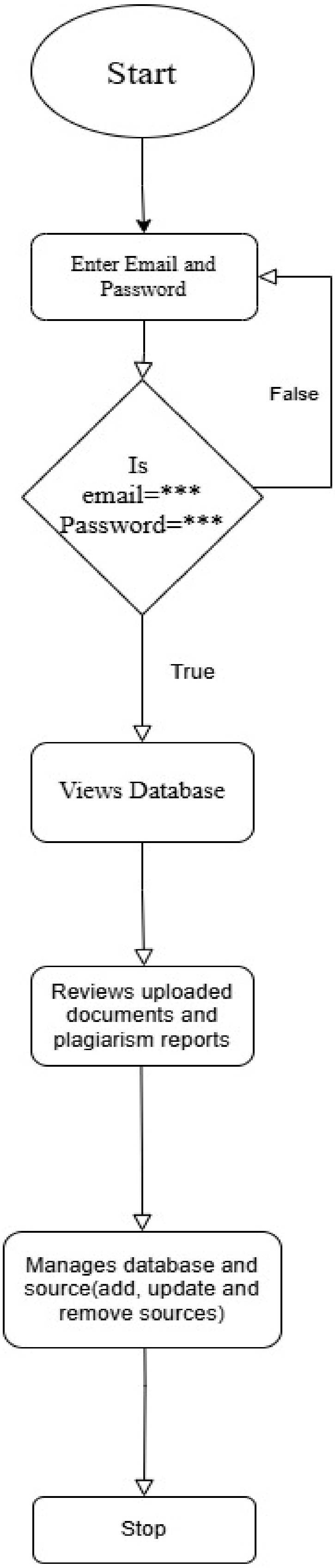

Start
Enter Email and Password
Is email=*** Password=***
False
True
Views Database
Reviews uploaded documents and plagiarism reports
Manages database and source(add, update and remove sources)
Stop

Figure 3.8 Flow Chart for admin

The steps involved in managing and gaining access to a database system are shown in this flowchart. The administrator inputs their password and email address to start the procedure. If the credentials are wrong, the process goes back and asks the administrator to input their password and email address again. The administrator is able to view the database after entering the necessary credentials. The administrator can then examine reports of plagiarism and uploaded documents from there. Lastly, the administrator can add, edit, or remove sources to manage the database and sources. This methodical procedure offers a clear workflow for database administration and authentication while guaranteeing data security and integrity.

1.Start: The process begins.

2.Enter Email and Password: The admin is prompted to enter an email and password.

3.Check Credentials: The system verifies if the entered email and password are correct.

- Incorrect Credentials (False): The admin is prompted to re-enter the email and password.
- Correct Credentials (True): The admin gains access to the system.

4.View Database: The admin is granted access to view the database.

5.Review Uploaded Documents and Plagiarism Reports: The admin can review the documents that have been uploaded and check plagiarism reports.

6.Manage Database and Sources: The admin can manage the database by adding, updating, or removing sources.

7.Stop: The process ends.

3.2.3. Database Schema

In the figure below, my project's database acts as a highly organized digital archive, where each distinct table serves a specific purpose: the User table meticulously manages all account information, including unique IDs, usernames, securely stored passwords, contact details, and activity timestamps, ensuring controlled access to the system; the Reference Document table functions as the project's core library, diligently storing all documents intended for plagiarism checks with their content, metadata, and associated user, forming the essential dataset for comparisons; and finally, the Plagiarism Check Result table serves as a comprehensive history log, recording every detail of each plagiarism analysis from who performed it and when, to the type of check, the input texts, the calculated similarity percentages, word counts, and even specific highlighted sections and detected sources providing a complete audit trail of all detection activities

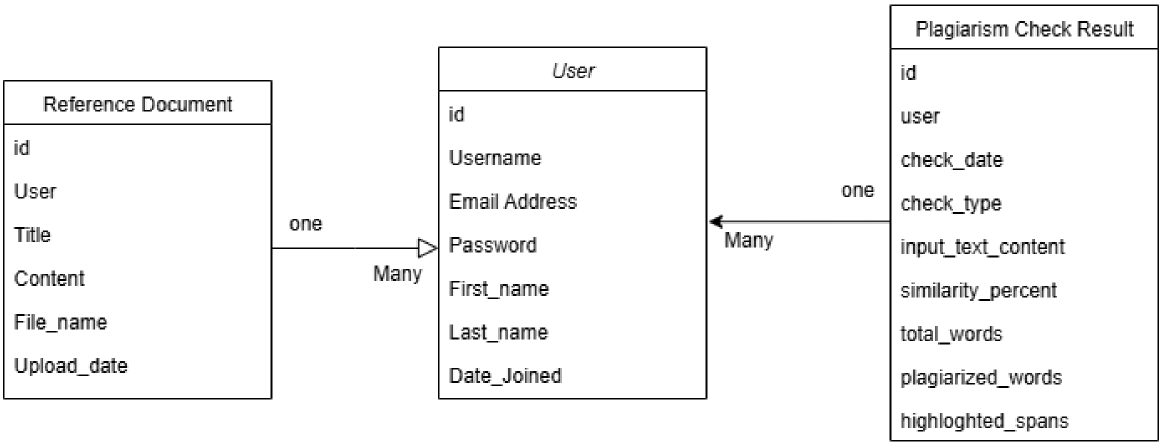


Figure 3.9 Database Schema

3.2.4 Wireframing

Wireframing is the process of developing a visual blueprint or guide for a software interface, app, or website. It helps in organizing the structure, functionality, and layout earlier than the actual design and development process.

Login Page:

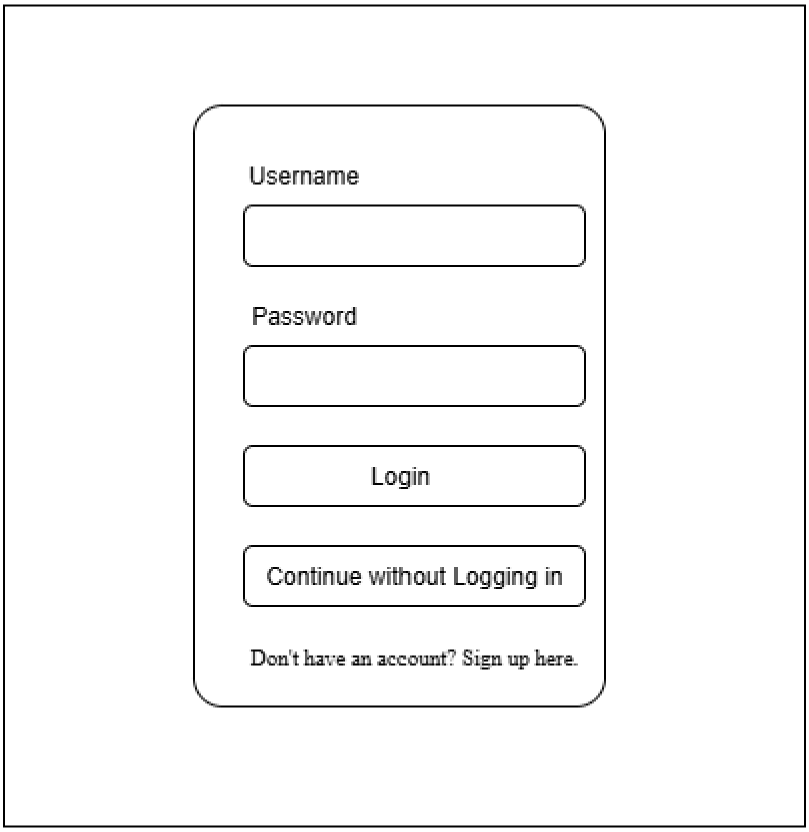

Username
Password
Login
Continue without Logging in
Don't have an account? Sign up here.

Figure 3.10 Wireframing Design of Login Page

Profile Page:

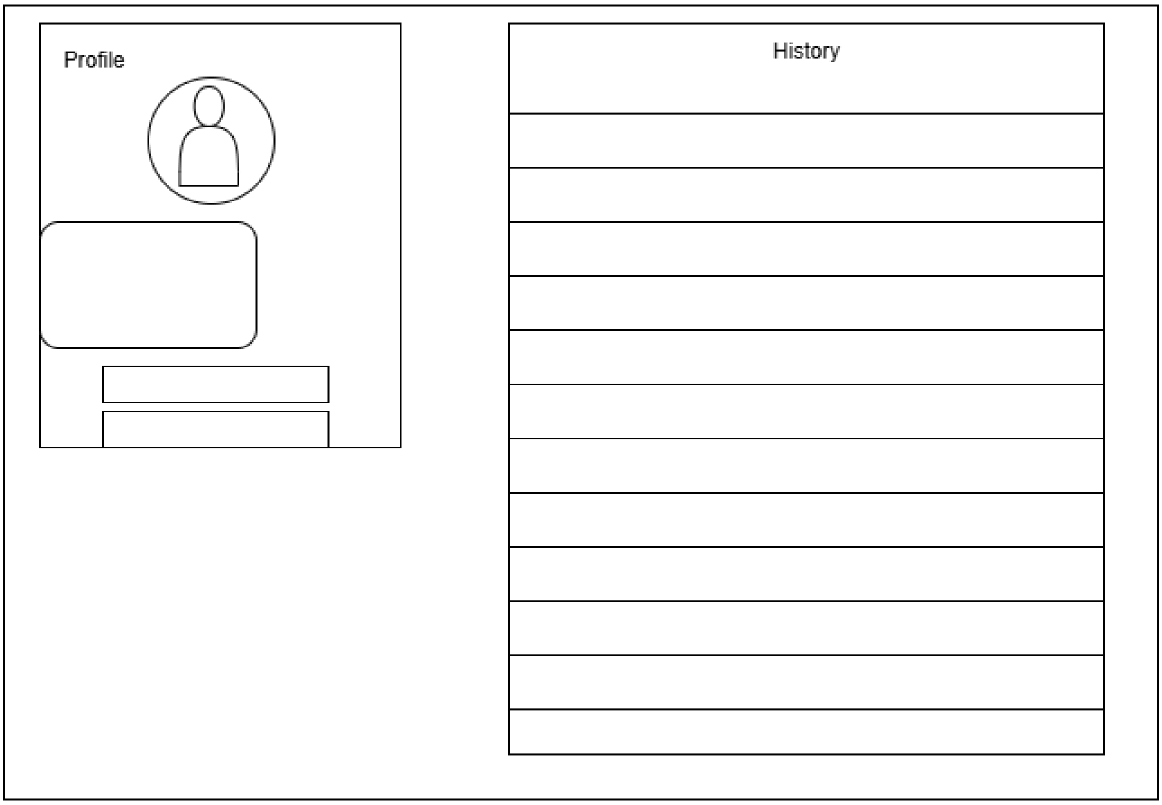


Figure 3.11 Wireframing Design of Profile Page

Home Page:

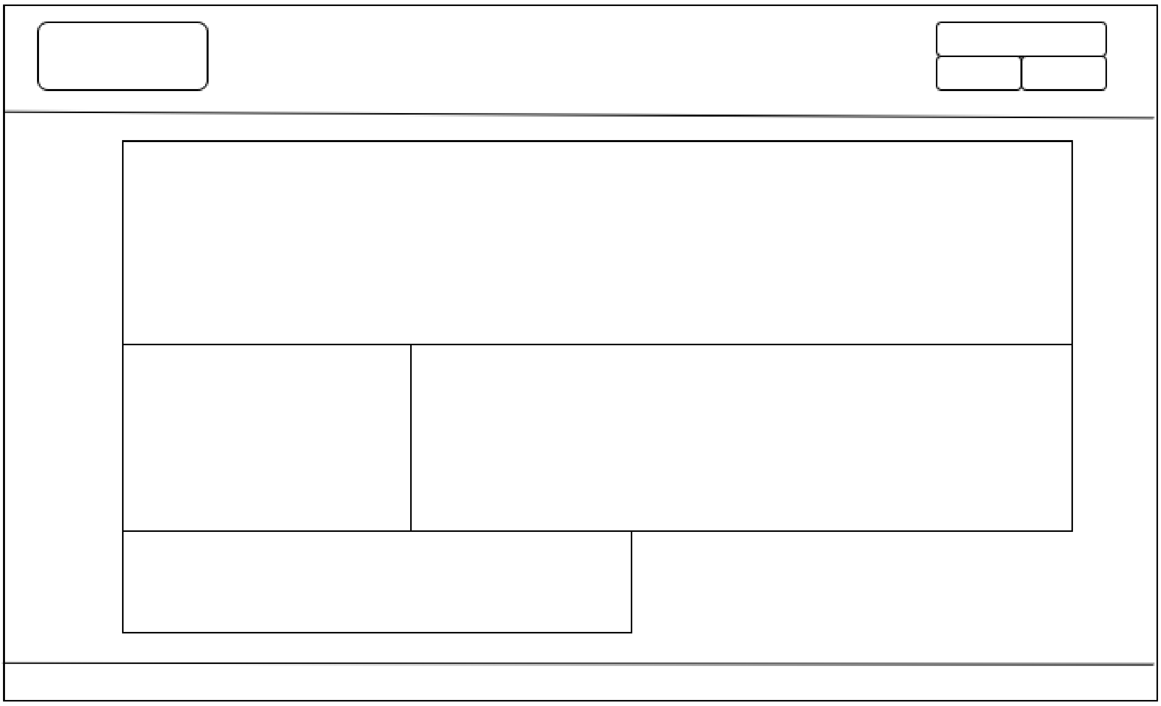


Figure 3.12 Wireframing Design of Home Page

User Page:

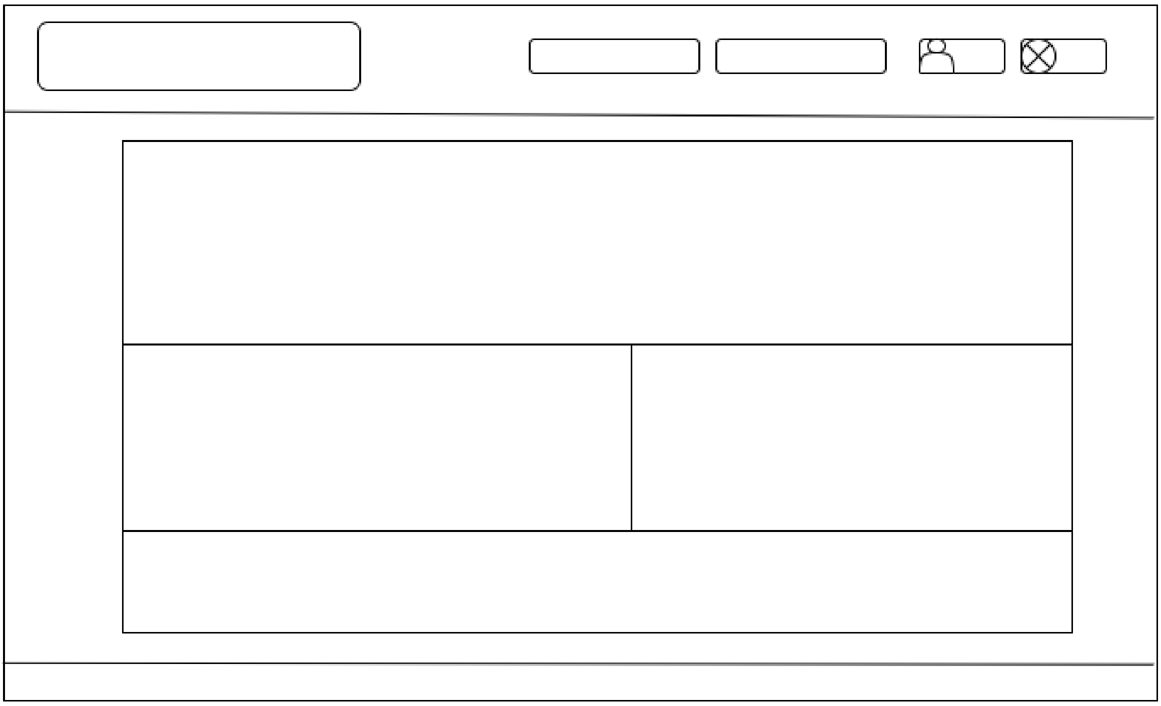


Figure 3.13 Wireframing Design of User Page

Plagiarism Detection Page:

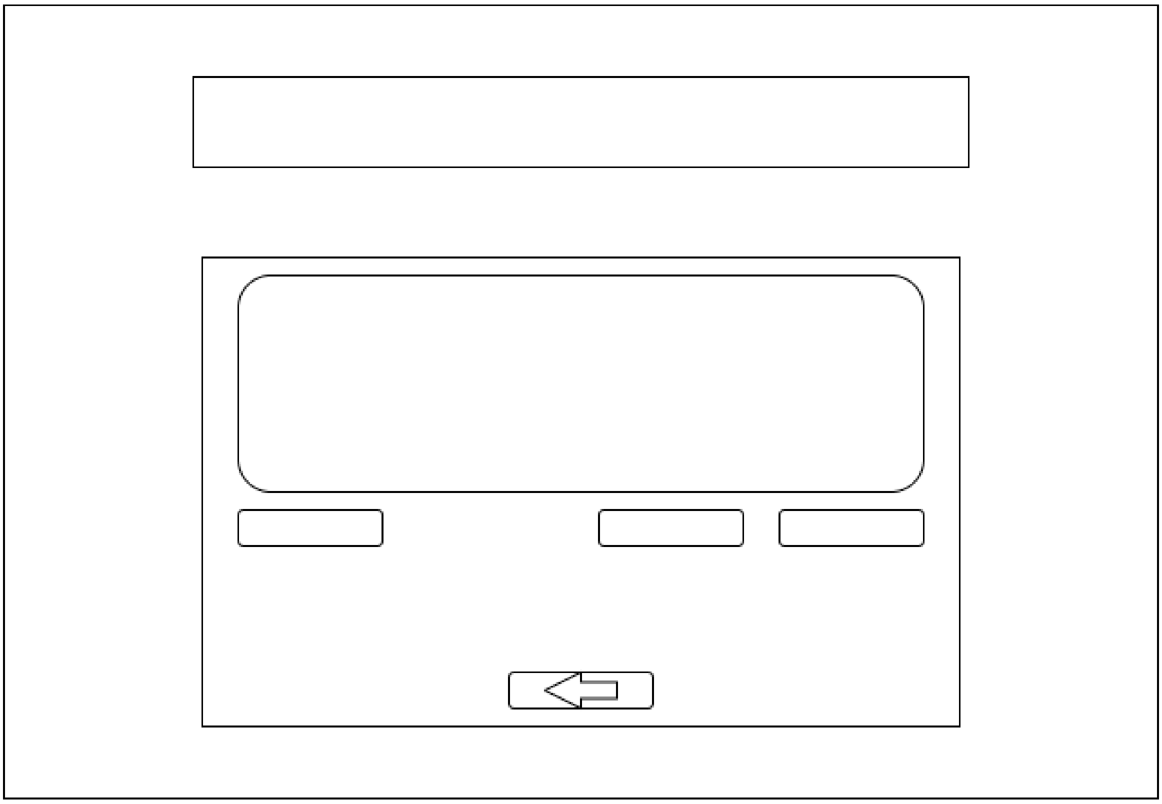


Figure 3.14 Wireframing Design of Plagiarism Detection Page

Compare Page:

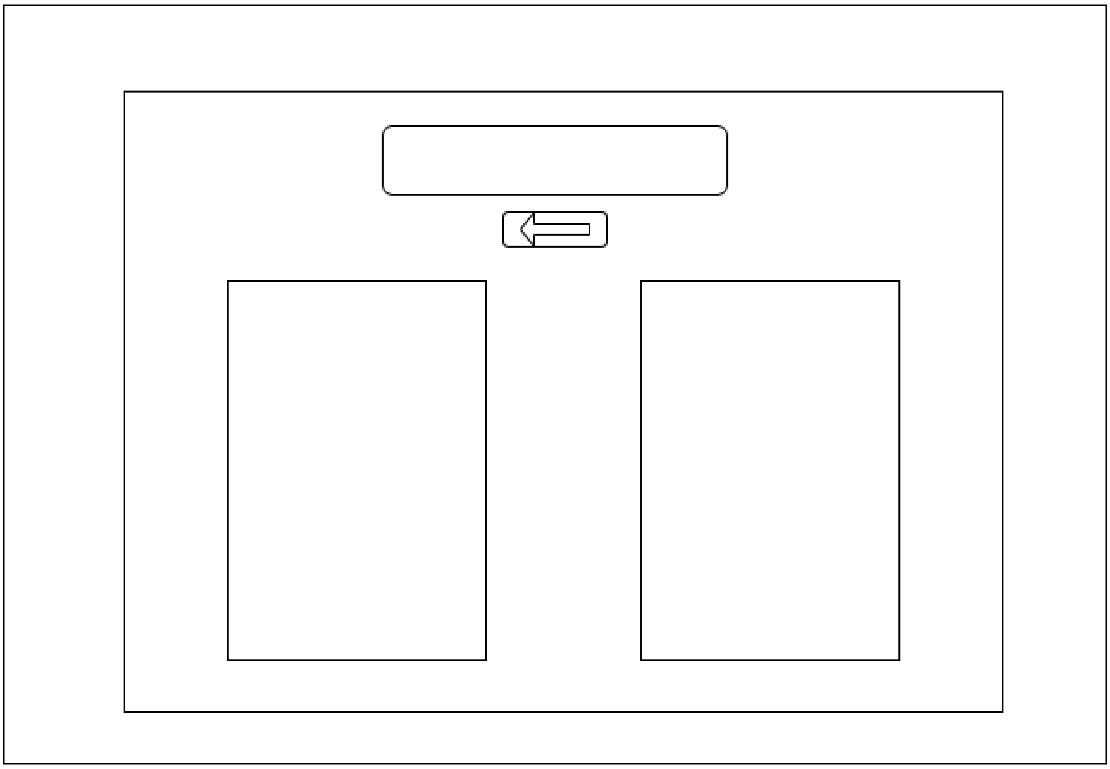
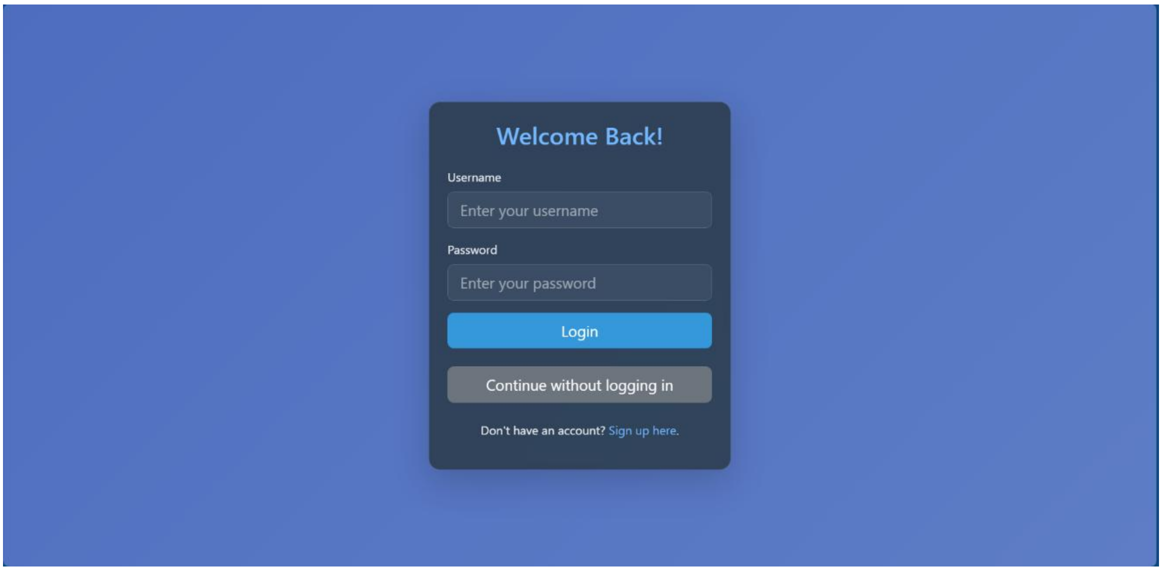


Figure 3.15 Wireframing Design of Compare Page

3.2.5 Interface Design



Welcome Back!

Username

Enter your username

Password

Enter your password

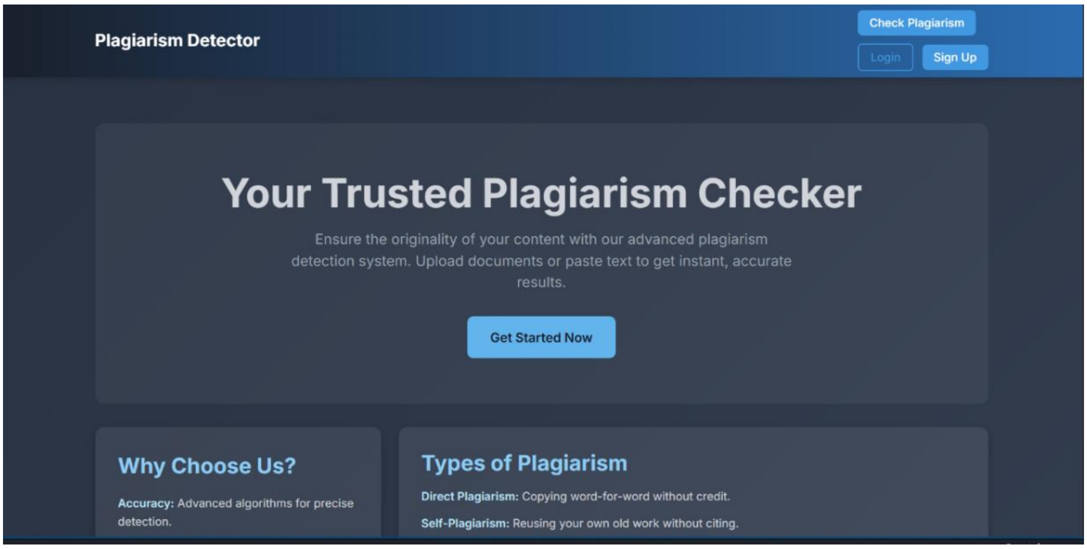
Login

Continue without logging in

Don't have an account? Sign up here.

Figure 3.16 Design of Login Page

The image displays a login form for Plagiarism Detection System. The form, centered on a blue-purple gradient background, has a title that says "Welcome Back!". It includes input fields for "Username" and "Password," a blue "Login" button, and a grey "Continue without logging in" button. Below the buttons, there is a link to "Sign up here" for new users.



Your Trusted Plagiarism Checker

Ensure the originality of your content with our advanced plagiarism detection system. Upload documents or paste text to get instant, accurate results.

Why Choose Us?

Accuracy: Advanced algorithms for precise detection.

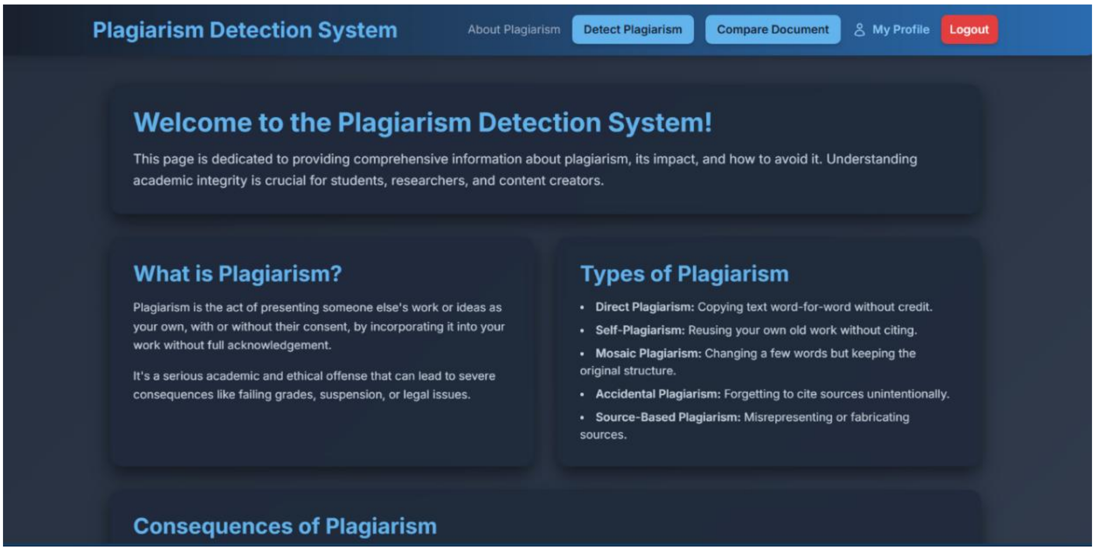
Types of Plagiarism

Direct Plagiarism: Copying word-for-word without credit.

Self-Plagiarism: Reusing your own old work without citing.

Figure 3.17 Design of Home Page

The image shows the homepage. The main section features a prominent title "Your Trusted Plagiarism Checker" with a brief description and a "Get Started Now" button. Below this are two cards with headings "Why Choose Us?" and "Types of Plagiarism," providing information on the service and plagiarism definitions. The header of the page includes the application's name and buttons for "Check Plagiarism," "Login," and "Sign Up."



Welcome to the Plagiarism Detection System!

This page is dedicated to providing comprehensive information about plagiarism, its impact, and how to avoid it. Understanding academic integrity is crucial for students, researchers, and content creators.

What is Plagiarism?

Plagiarism is the act of presenting someone else's work or ideas as your own, with or without their consent, by incorporating it into your work without full acknowledgement.

It's a serious academic and ethical offense that can lead to severe consequences like failing grades, suspension, or legal issues.

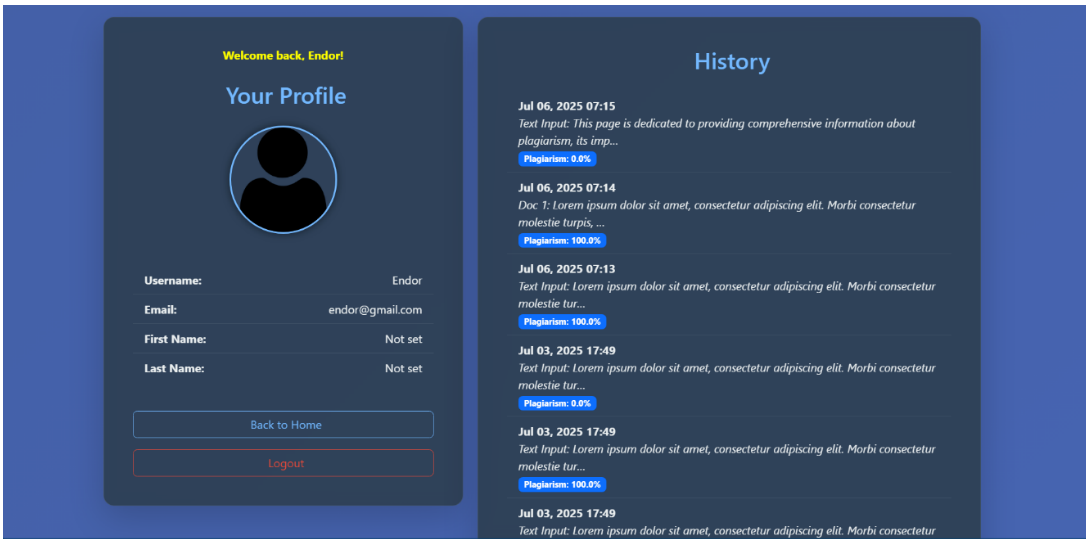
Types of Plagiarism

- **Direct Plagiarism:** Copying text word-for-word without credit.
- **Self-Plagiarism:** Reusing your own old work without citing.
- **Mosaic Plagiarism:** Changing a few words but keeping the original structure.
- **Accidental Plagiarism:** Forgetting to cite sources unintentionally.
- **Source-Based Plagiarism:** Misrepresenting or fabricating sources.

Consequences of Plagiarism

Figure 3.18 Design of User Home Page

The image shows the page after Login (User Home Page). The page is titled "Welcome to the Plagiarism Detection System!" and provides information about plagiarism. It features two prominent cards: one defining "What is Plagiarism?" and another listing and describing "Types of Plagiarism," such as Direct and Self-Plagiarism. The header of the page includes navigation links like "About Plagiarism," "Detect Plagiarism," "Compare Document," "My Profile," and "Logout."


Welcome back, Endor!

Your Profile

Username: Endor

Email: endor@gmail.com

First Name: Not set

Last Name: Not set

History

Jul 06, 2025 07:15
Text Input: This page is dedicated to providing comprehensive information about plagiarism, its imp...
Plagiarism: 0.0%

Jul 06, 2025 07:14
Doc 1: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi consectetur molestie turpis, ...
Plagiarism: 100.0%

Jul 06, 2025 07:13
Text Input: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi consectetur molestie tur...
Plagiarism: 100.0%

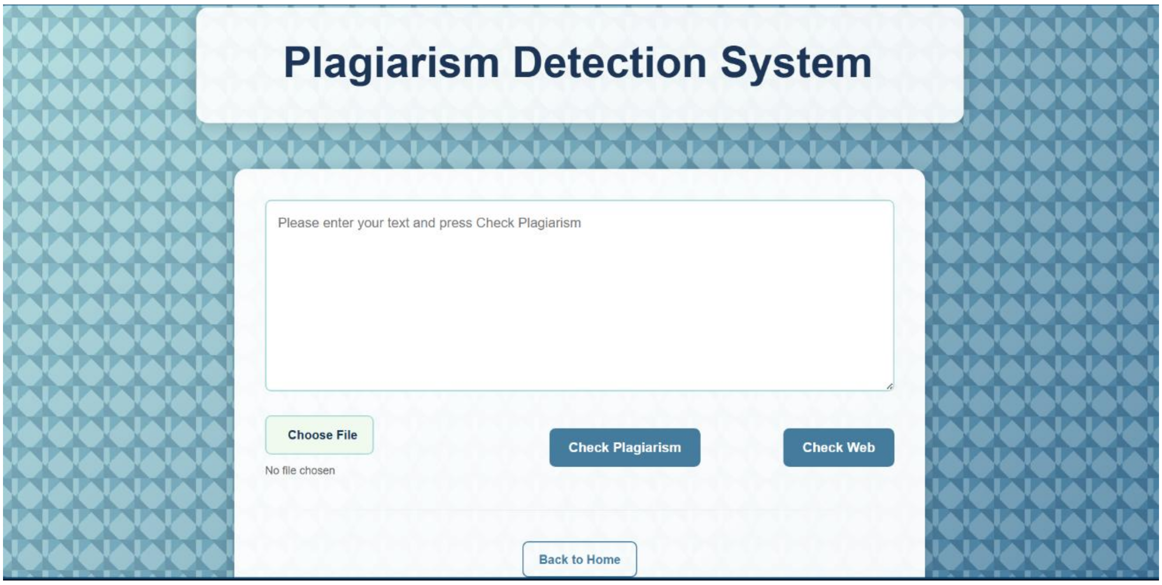
Jul 03, 2025 17:49
Text Input: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi consectetur molestie tur...
Plagiarism: 0.0%

Jul 03, 2025 17:49
Text Input: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi consectetur molestie tur...
Plagiarism: 100.0%

Jul 03, 2025 17:49
Text Input: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi consectetur

Figure 3.19 Design of User Profile Page

The image shows a user's profile and history page. The left side, labeled "Your Profile," displays a user avatar, username ("Endor"), and other details like email. The right side, titled "History," lists several past plagiarism checks. Each entry includes the date and time, a preview of the text input, and a colored bar indicating the plagiarism percentage.


Plagiarism Detection System

Please enter your text and press Check Plagiarism

No file chosen

Figure 3.20 Design of Plagiarism Detection Page

The image shows the core functionality page of the "Plagiarism Detection System." It features a large text area for a user to paste content, a "Choose File" button for document uploads, and two primary action buttons: "Check Plagiarism" and "Check Web." The interface is clean and straightforward, focusing on the main task of submitting text or a file for analysis.

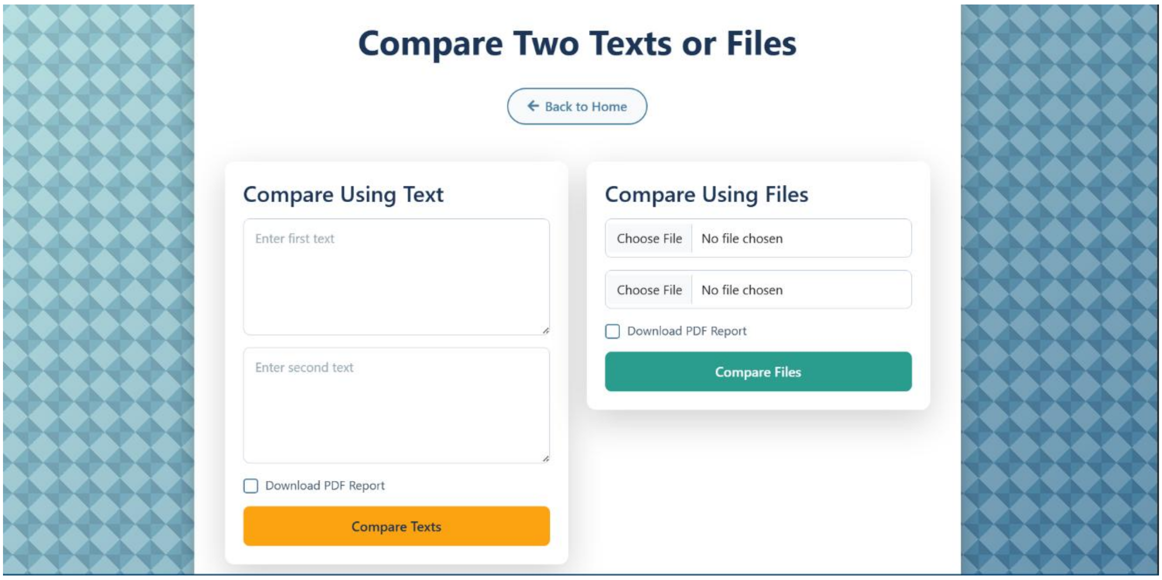


Figure 3.21 Design of Compare Page

The image displays comparison page, designed to check plagiarism between two sources. The page is divided into two main sections: one for comparing text by pasting it into two separate text areas, and another for comparing two files uploaded by the user. Both sections offer the option to download a PDF report and have a button to initiate the comparison, with the "Compare Texts" button being orange and the "Compare Files" button being a greenish-blue.

3.3 Algorithm (Fuzzy Matching Algorithm)

1.Preprocessing:

- Tokenization: Dividing the supplied document's text into smaller chunks known as tokens (words or phrases?). Libraries such as NLTK will be used for this purpose.
- Normalization: To make the text consistent, change all of the tokens to lowercase, eliminate punctuation, and use stemming or lemmatization. By doing this, variations brought on by various word forms are lessened.

2.Similarity Calculation:

- Cosine Similarity: Find the cosine similarity between the provided document's vectors and those of the database's existing documents. There are two possible values for the cosine similarity score: 0 (no resemblance) and 1 (identical).

3.Thresholding:

- Set a Similarity Threshold: Establish a threshold for similarity (0.7, for example). A document pair is marked as possibly plagiarized if its cosine similarity score is higher than this limit.

4.Post processing:

- Highlighting: Identify the text's specific passages that are identical to those in other texts and highlight them. This makes it easier for users to spot potentially plagiarized information.
- Reporting: Create a report that contains the matched portions, the sources of the matched material, and the similarity score. Send the user or administrator this report so they can review it further.

5.Formula Used:

Similarity Percentage = (2 × Number of Matching Characters) ÷ (Total Characters in Text 1 + Total Characters in Text 2)

By following this algorithm, we will effectively implement the fuzzy matching algorithm in plagiarism detection system, making it more robust and accurate in identifying both exact matches and paraphrased content.

Chapter-4: Implementation and Testing

4.1 Implementation

To develop the software, some standard tools that are very essential and common are used, which are mentioned below.

4.1.1 Tools Used

For visualizing system components and creating user interfaces, we rely on draw.io and Figma. draw.io aids in system design with clear diagrams, while Figma enables interactive and visually appealing UI designs. We optimize coding efficiency using Visual Studio Code (VSCode), an IDE with features like syntax highlighting and code completion.

4.1.1.1 Programming Languages

- HTML, CSS, and JavaScript are used for creating interactive user interfaces.
- Python is used as the server-side scripting language to generate dynamic content and interact with the database, supporting the complex logic of plagiarism detection.

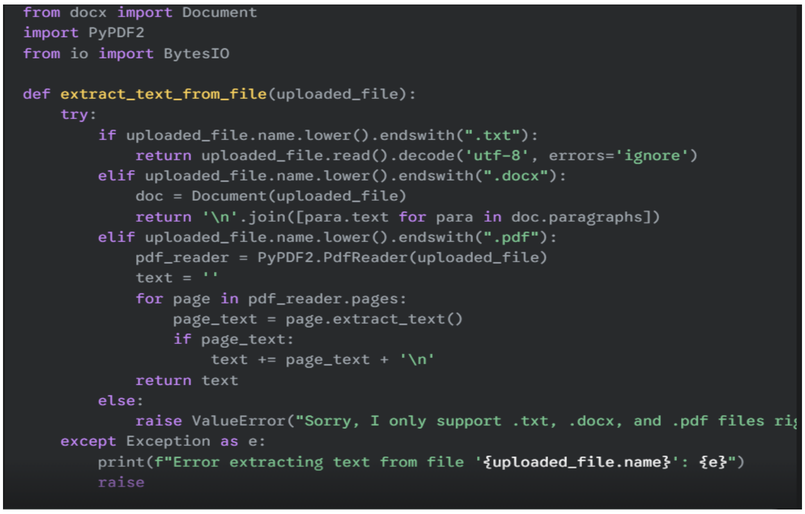
4.1.1.2 Database Platforms

- SQLite serves as the database management system for effective data storage and retrieval of submitted documents, reference materials, user accounts, and plagiarism reports.

The integration of HTML, CSS, JavaScript, and Python version, combined with the SQLite database, ensures the efficient implementation of the Plagiarism Detection System by providing core functionalities such as document analysis, report generation, and user management.

4.1.2 Code Snippet

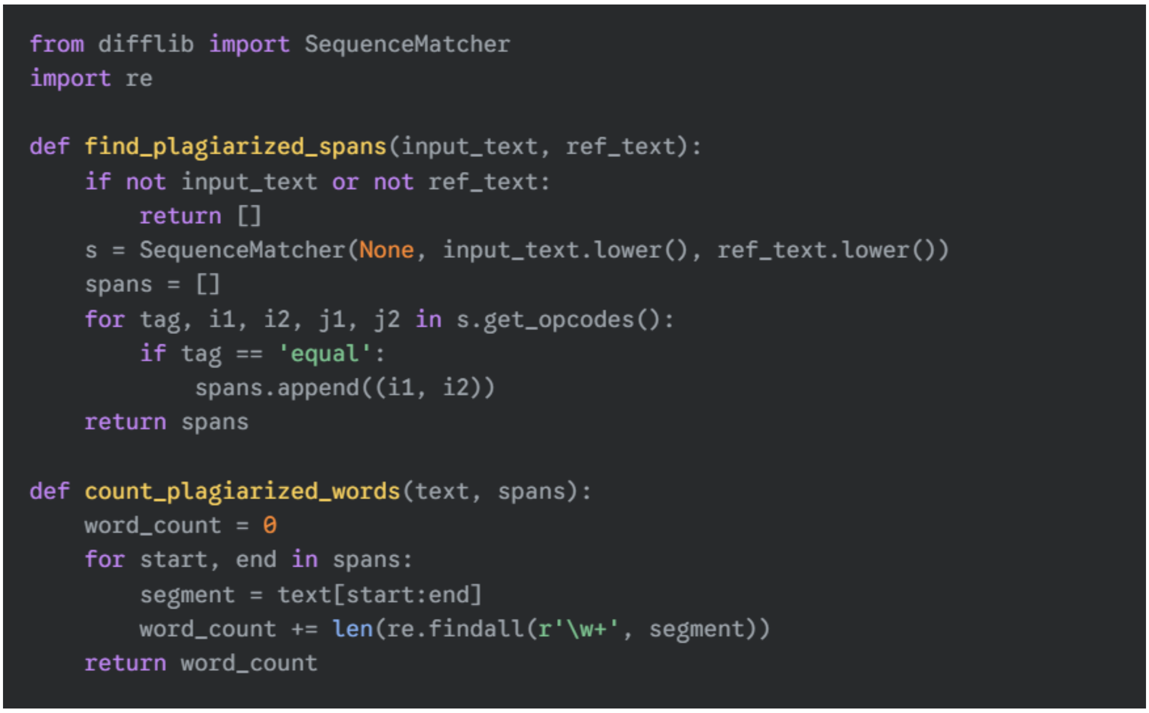
i. Extracting Text from Different File Types



```
from docx import Document
import PyPDF2
from io import BytesIO

def extract_text_from_file(uploaded_file):
    try:
        if uploaded_file.name.lower().endswith(".txt"):
            return uploaded_file.read().decode('utf-8', errors='ignore')
        elif uploaded_file.name.lower().endswith(".docx"):
            doc = Document(uploaded_file)
            return '\n'.join([para.text for para in doc.paragraphs])
        elif uploaded_file.name.lower().endswith(".pdf"):
            pdf_reader = PyPDF2.PdfReader(uploaded_file)
            text = ''
            for page in pdf_reader.pages:
                page_text = page.extract_text()
                if page_text:
                    text += page_text + '\n'
            return text
        else:
            raise ValueError("Sorry, I only support .txt, .docx, and .pdf files ri
    except Exception as e:
        print(f"Error extracting text from file '{uploaded_file.name}': {e}")
        raise
```

Listing 4.1 Code Snippet of Extracting Text from Different File Types

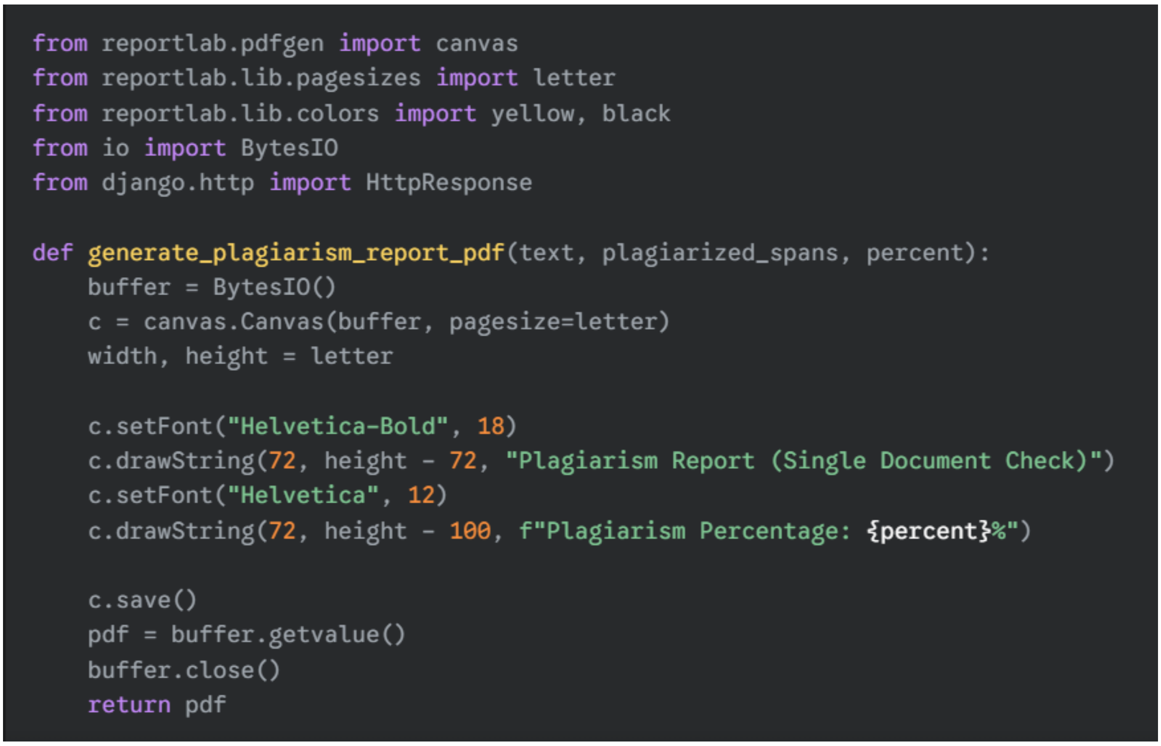
ii. Core Plagiarism Detection Logic (Similarity & Spans)



```python
from difflib import SequenceMatcher
import re

def find_plagiarized_spans(input_text, ref_text):
    if not input_text or not ref_text:
        return []
    s = SequenceMatcher(None, input_text.lower(), ref_text.lower())
    spans = []
    for tag, i1, i2, j1, j2 in s.get_opcodes():
        if tag == 'equal':
            spans.append((i1, i2))
    return spans

def count_plagiarized_words(text, spans):
    word_count = 0
    for start, end in spans:
        segment = text[start:end]
        word_count += len(re.findall(r'\w+', segment))
    return word_count
```

Listing 4.2 Code Snippet of Core Plagiarism Detection Logic

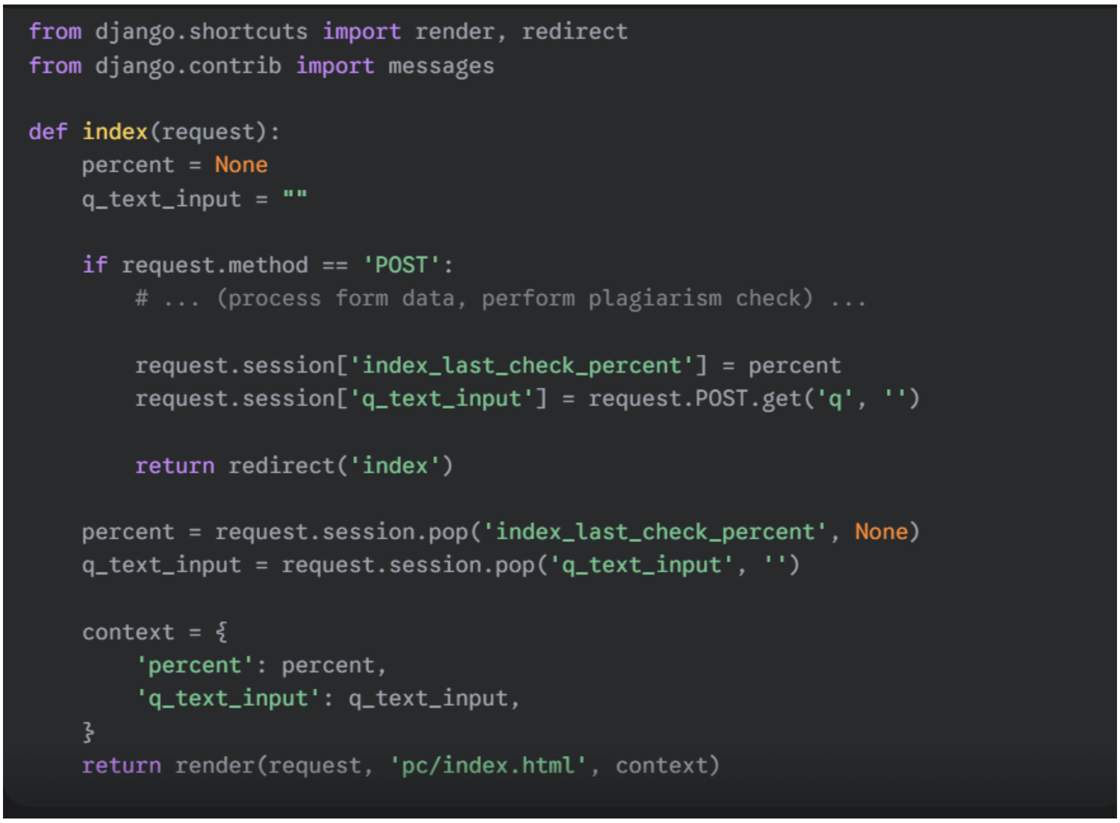
iii.Generating and Serving a PDF Report



```python
from reportlab.pdfgen import canvas
from reportlab.lib.pagesizes import letter
from reportlab.lib.colors import yellow, black
from io import BytesIO
from django.http import HttpResponse

def generate_plagiarism_report_pdf(text, plagiarized_spans, percent):
    buffer = BytesIO()
    c = canvas.Canvas(buffer, pagesize=letter)
    width, height = letter

    c.setFont("Helvetica-Bold", 18)
    c.drawString(72, height - 72, "Plagiarism Report (Single Document Check)")
    c.setFont("Helvetica", 12)
    c.drawString(72, height - 100, f"Plagiarism Percentage: {percent}%")

    c.save()
    pdf = buffer.getvalue()
    buffer.close()
    return pdf
```

Listing 4.3 Code Snippet of Generating and Serving a PDF report

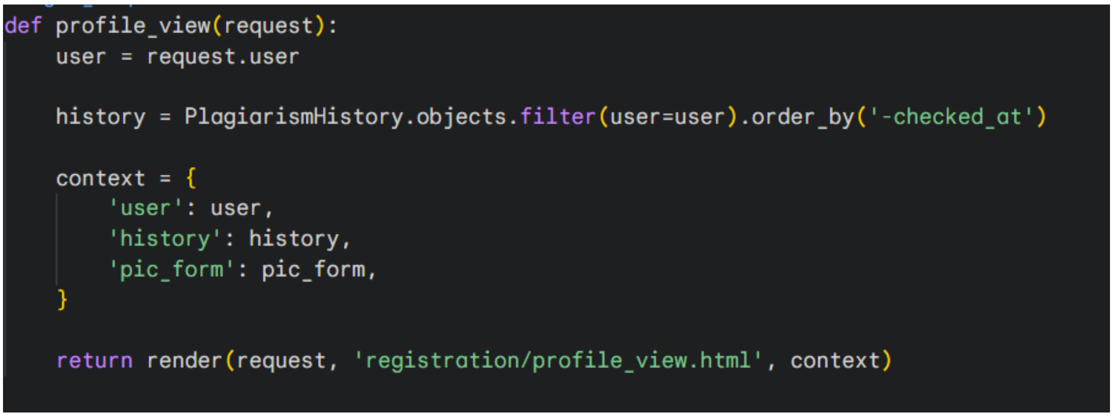
iv. Post-Redirect-Get (PRG) Pattern with Session Management



```python
from django.shortcuts import render, redirect
from django.contrib import messages

def index(request):
    percent = None
    q_text_input = ""

    if request.method == 'POST':
        # ... (process form data, perform plagiarism check) ...

        request.session['index_last_check_percent'] = percent
        request.session['q_text_input'] = request.POST.get('q', '')

        return redirect('index')

    percent = request.session.pop('index_last_check_percent', None)
    q_text_input = request.session.pop('q_text_input', '')

    context = {
        'percent': percent,
        'q_text_input': q_text_input,
    }
    return render(request, 'pc/index.html', context)
```

Listing 4.4 Code Snippet of Post-Redirect-Get Pattern with Session Management

v. Database Connectivity



```python
def profile_view(request):
    user = request.user

    history = PlagiarismHistory.objects.filter(user=user).order_by('-checked_at')

    context = {
        'user': user,
        'history': history,
        'pic_form': pic_form,
    }

    return render(request, 'registration/profile_view.html', context)
```

Listing 4.5 Code Snippet of Database Connectivity

4.2 Testing

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or services under test. It is the process of executing software with the intent of finding bugs and to ensure that it satisfies the specified requirements. After the development of the system, we have tested whether it meets the requirements or not in different phases. The functionality has also been tested taking some test cases.

4.2.1 Unit Testing

Unit testing in the Plagiarism Detection System is a vital step to ensure the correctness and reliability of individual components. This process involves designing and executing test cases for specific modules such as user authentication, document submission, plagiarism analysis algorithms, and report generation functionalities. By conducting unit tests, we validate that each module operates as intended in isolation. Comprehensive unit testing helps identify and resolve bugs at an early stage, ensuring a stable foundation for the entire system.

4.2.2 System Testing

System testing plays a key role in verifying the proper integration and overall functionality of all modules within the Plagiarism Detection System. This phase involves testing the system as a whole to ensure seamless interaction between components like the admin panel, user dashboard, plagiarism detection workflows, and database operations. The objective is to confirm that the entire system works cohesively to deliver accurate and efficient performance under real-world conditions.

4.2.3 Test Cases

Test cases for system testing cover a broad range of functionalities essential to the plagiarism detection system. Examples include verifying login and registration processes to ensure secure user access, checking the functionality and accuracy of plagiarism detection algorithms against various text samples, ensuring that document uploads and storage are handled correctly, and validating that plagiarism reports are accurately generated and displayed in the dashboard. System testing was performed to guarantee a reliable, user-friendly experience for both students and administrators, enabling efficient plagiarism checking and management. The software was tested under various scenarios to validate its robustness. Some of the key test cases designed to assess the correctness and reliability of the system are listed and discussed in detail in the following test table.

User Registration

Table 4.1 Test Cases for User Registration for PDS

S.N	Test Name	Input	Expected outcome	Actual Outcome	Test Result
1.	Open Application	http://127.0.0.1:8000/	At front page	At front page	Pass
2.	Open SignUp	http://127.0.0.1:8000/signup	At front page	At front page	Pass
3.	Enter Invalid Username, email, address, choose user, password, confirm password and click register button	Username = Ananta E-mail= abc@gmail.com Password = asdfgh123 Confirm password =	Enter Confirm password	Registration Failed	Pass
4.	Enter Valid Username, email, address, choose user, password, confirm	Username = Ananta E-mail= abc@gmail.com Password = asdfgh123 Confirm password = asdfgh123	Registration successful and redirect to index page	Registration successful and redirect to index page	Pass

	password and click register button				

User Login

Table 4.2 Test case of User login for PDS

S. N	Test Name	Input	Expected Output	Actual Outcome	Test Result
1.	Open Application	http://127.0.0.1:8000	At Form Page	At Form page	Pass
2.	Enter Username and Invalid password and click login button	UserName: Ananta Password: 123	Login Failed And refresh the fields in same page	Login failed	Pass
3.	Enter Valid Username and Password and click login button	UserName: Ananta Password: asdfgh123	Login Successful and redirect to home page	Redirect to home page	Pass

Admin Login

Table 4.3 Test case of Admin Login for PDS

S. N	Test Name	Input	Expected Output	Actual Outcome	Test Result

1.	Open Application	http://127.0.0.1:8000/admin	At Form Page	At Form page	Pass
2.	Enter Username and Invalid password and click login button	UserName: endor Password: 123	Login Failed And refresh the fields in same page	Login failed	Pass
3.	Enter Valid Username and Password and click login button	UserName: endor Password: admin@123	Login Successful and redirect to home page	Redirect to admin dashboard	Pass

4.3 Result and Analysis

The Plagiarism Detection System has been successfully designed and implemented as a comprehensive platform for automating the process of checking academic integrity. The system uses an intelligent text comparison mechanism to identify similarities between a submitted document and a vast repository of web content and local reference files, significantly reducing the manual effort and potential for errors in plagiarism review.

The system is role-based, allowing users to submit documents, view their history, and generate detailed reports, while administrators can oversee the entire platform. The front-end is built with HTML, CSS, and JavaScript, with a Python/Django back-end and SQLite database, ensuring the system is secure, modular, and scalable.

The most impactful feature is its dual-layered checking mechanism, which provides a thorough analysis. The system's strong performance and user-friendly interface are key strengths. Looking ahead, there is plan to enhance the core algorithm to detect paraphrasing, integrate advanced analytics, and develop a mobile application to improve accessibility and functionality

Chapter-5: Conclusion and Future Recommendation

5.1. Conclusion

In Closing, The Plagiarism Detection System project has the potential to significantly influence the maintenance of professional and academic integrity. The system provides a strong and dependable way to identify several types of plagiarism by utilizing Python and the Natural Language Toolkit (NLTK), in addition to HTML, CSS, Python, JavaScript, and SQLite. By means of meticulous system analysis, requirement analysis, and feasibility assessment, we have proven the technical, financial, and operational feasibility of the project. The key goals of the system are precise detection, an intuitive user interface, effective processing, thorough analysis, and data protection. By meeting these goals and following both functional and non-functional specifications, the Plagiarism Detection System seeks to give consumers an easy-to-use and trustworthy tool for confirming the uniqueness of their material. As we continue to create and carry out this project, we are dedicated to supporting ethical writing habits, maintaining academic integrity, and raising the standard of professional and scholarly work in general.

5.2. Outcome

- The system offers a fast and precise way to identify instances of plagiarism, significantly improving academic and professional integrity.
- A natural interface makes it easy for users, especially instructors and students, to upload documents and access detailed plagiarism reports quickly.
- The system provides detailed detection reports that show the similarity percentage and the sources of suspected plagiarism, helping users resolve any issues effectively.
- Administrators will have robust tools to manage document uploads, oversee user report, and monitor system performance, ensuring seamless and efficient operations.

  By achieving these goals, the Plagiarism Detection System aims to foster a culture of honesty and creativity, enhancing the overall quality and integrity of academic and professional work.

5.3. Future Recommendation

- Add AI detections.
- Add Humanization for the plagiarized text and documents.
- Add high search capability for web searching and similarity checking.
- Add high efficiency detection capability with more accurate source detection.

References

[1] N. M. G. TOMÁŠ FOLTÝNEK, "Academic Plagiarism Detection: A Systematic Literature Review Academic Plagiarism Detection: A Systematic Literature Review," *Academic Plagiarism Detection,* vol. 52, p. 42, 2019.

[2] M. Agrawal, "IEEE Xplore," 16 March 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7877421. [Accessed March 2025].

[3] D. K. Sharma, "Aclanthology," Coling 2010 Organizing Committee, August 2010. [Online]. Available: https://aclanthology.org/C10-2115. [Accessed March 2025].

[4] Sven Meyer zu Eissen, Benno Stein , "springer," Lecture Notes in Computer Science, 2006. [Online]. Available: https://link.springer.com/chapter/10.1007/11735106_66#Abs1. [Accessed March 2025].

[5] E. E. Mohammad Khalil, "springer," 09 June 2023. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-34411-4_32. [Accessed March 2025].

[6] A. S. Y. Bin-Habtoor, "researchgate," January 2012. [Online]. Available: https://www.researchgate.net/publication/271302675_A_Survey_on_Plagiarism_Detection_Systems. [Accessed March 2025].

[7] S. S. Patil, "Researchgate," Shram Sadhana Bombay Trust's College of Engineering and Technology, Feburary 2019. [Online]. Available: https://www.researchgate.net/publication/331062965_Overview_of_Plagiarism_Checkers_and_Plagiarism_Detection_Tools_A_Study. [Accessed March 2025].

[8] H. Chowdhury, "Plagiarism: Concept and Tools for Detection," *Plagiarism,* vol. Volume V, p. 14, 2020.

[9] M. A. Zaher, "MGCUB," 22 May 2020. [Online]. Available: https://mgcub.ac.in/f. [Accessed March 2025].

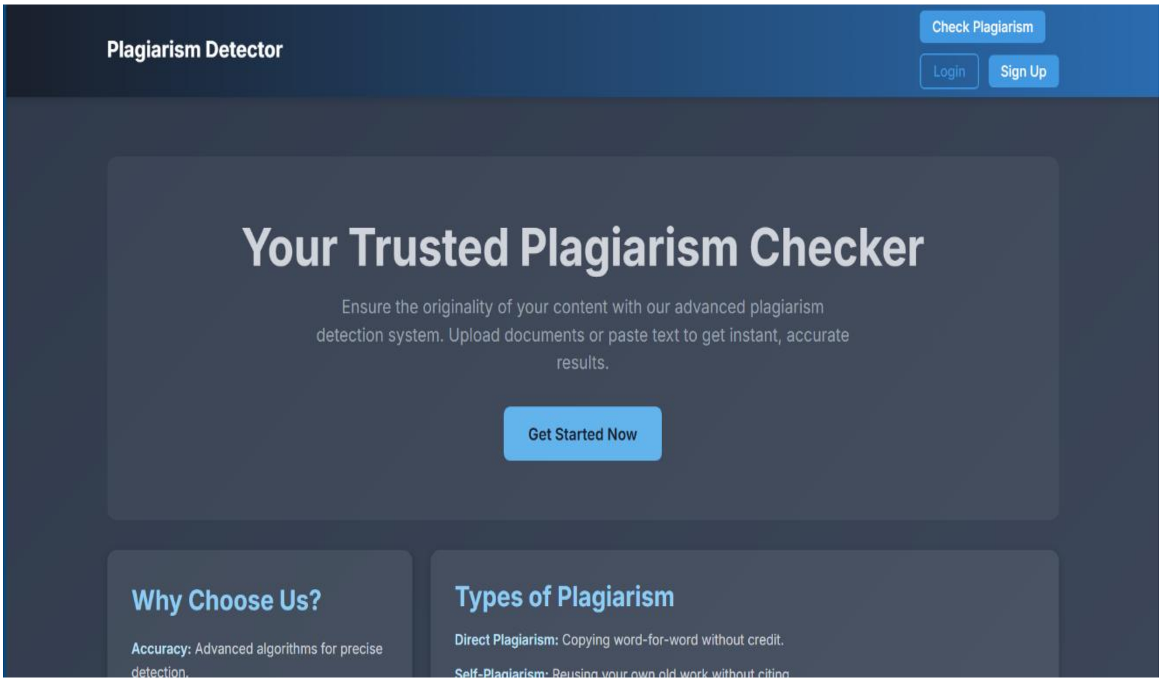
[10] M. Abdelhamid, "Arxiv," May 2020. [Online]. Available: https://arxiv.org/pdf/2201.03423. [Accessed March 2025].

[11] S. Gupta, "ResearchGate," Prateeksha Publications, January 2071. [Online]. Available: https://www.researchgate.net/publication/335319583_Plagiarism_Detection_Software_an_Overview. [Accessed March 2025].

[12] D. Trivedi, "Research Gate," Bowie State University, December 2021. [Online]. Available: https://www.researchgate.net/publication/356924683_Agile_Methodologies. [Accessed March 2025].

Appendices

Screenshots



Your Trusted Plagiarism Checker

Ensure the originality of your content with our advanced plagiarism detection system. Upload documents or paste text to get instant, accurate results.

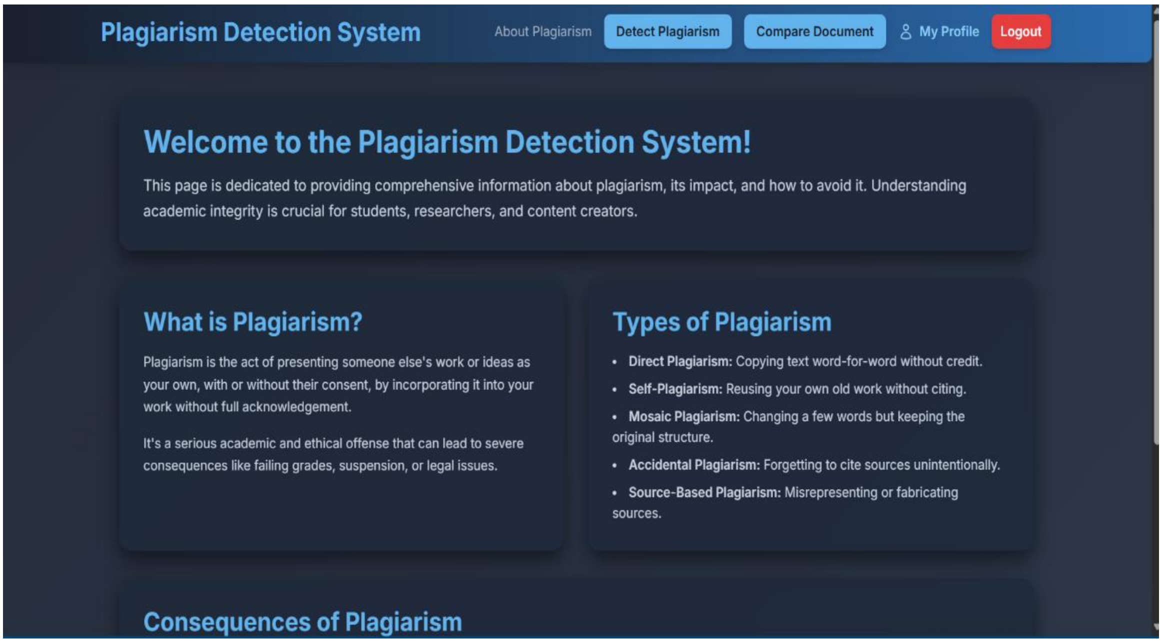
Why Choose Us?

Accuracy: Advanced algorithms for precise detection.

Types of Plagiarism

Direct Plagiarism: Copying word-for-word without credit.

Self-Plagiarism: Reusing your own old work without citing



Welcome to the Plagiarism Detection System!

This page is dedicated to providing comprehensive information about plagiarism, its impact, and how to avoid it. Understanding academic integrity is crucial for students, researchers, and content creators.

What is Plagiarism?

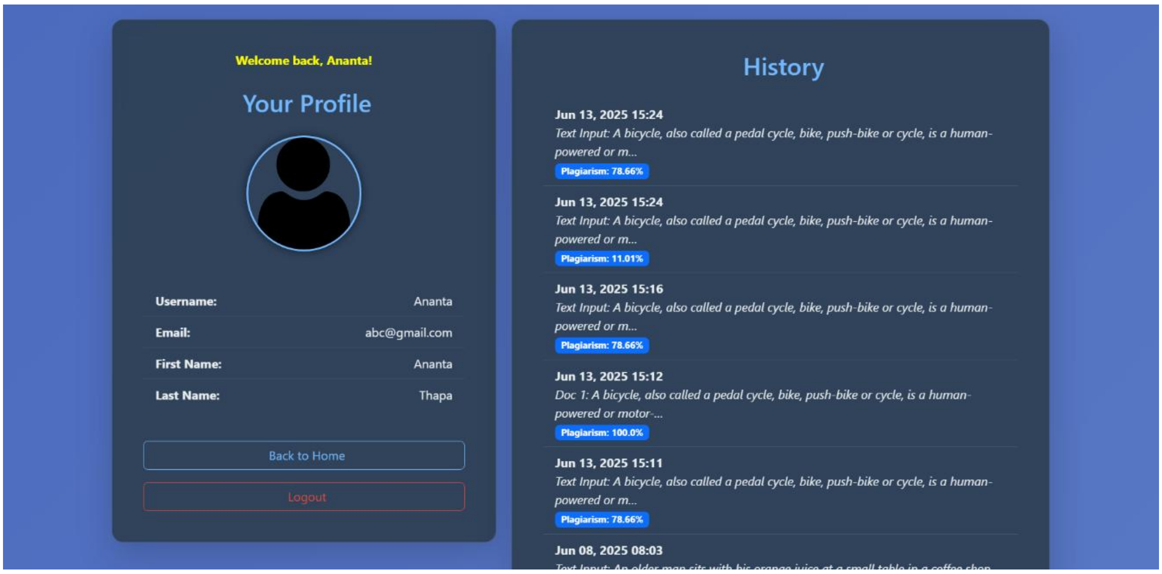
Plagiarism is the act of presenting someone else's work or ideas as your own, with or without their consent, by incorporating it into your work without full acknowledgement.

It's a serious academic and ethical offense that can lead to severe consequences like failing grades, suspension, or legal issues.

Types of Plagiarism

- **Direct Plagiarism:** Copying text word-for-word without credit.
- **Self-Plagiarism:** Reusing your own old work without citing.
- **Mosaic Plagiarism:** Changing a few words but keeping the original structure.
- **Accidental Plagiarism:** Forgetting to cite sources unintentionally.
- **Source-Based Plagiarism:** Misrepresenting or fabricating sources.

Consequences of Plagiarism



Welcome back, Ananta!

Your Profile

Username:	Ananta
Email:	abc@gmail.com
First Name:	Ananta
Last Name:	Thapa

History

Jun 13, 2025 15:24
Text Input: A bicycle, also called a pedal cycle, bike, push-bike or cycle, is a human-powered or m...
Plagiarism: 78.66%

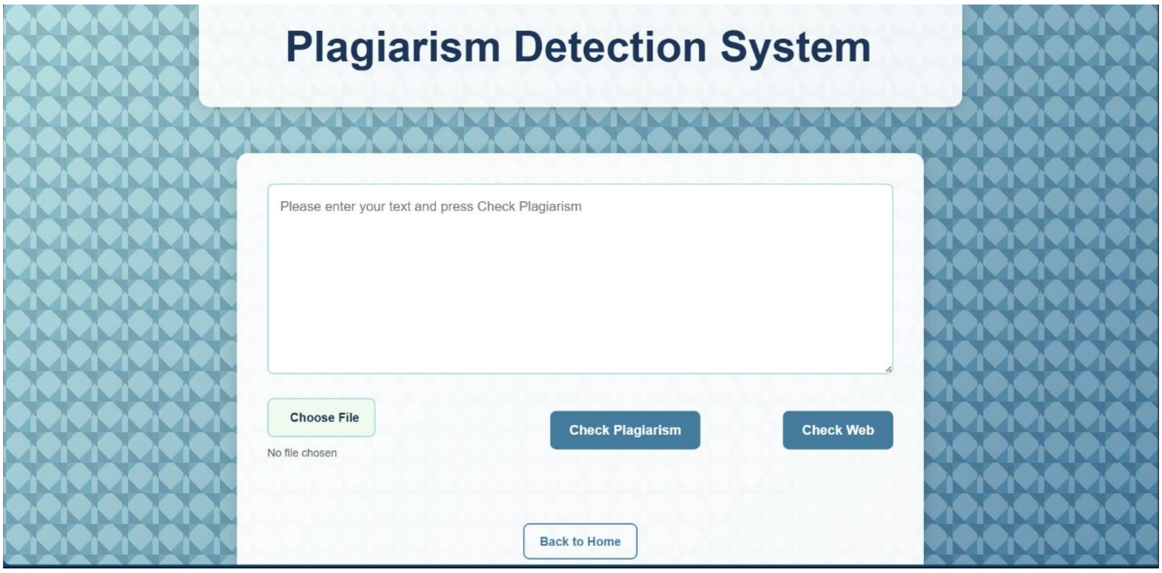
Jun 13, 2025 15:24
Text Input: A bicycle, also called a pedal cycle, bike, push-bike or cycle, is a human-powered or m...
Plagiarism: 11.01%

Jun 13, 2025 15:16
Text Input: A bicycle, also called a pedal cycle, bike, push-bike or cycle, is a human-powered or m...
Plagiarism: 78.66%

Jun 13, 2025 15:12
Doc 1: A bicycle, also called a pedal cycle, bike, push-bike or cycle, is a human-powered or motor-...
Plagiarism: 100.0%

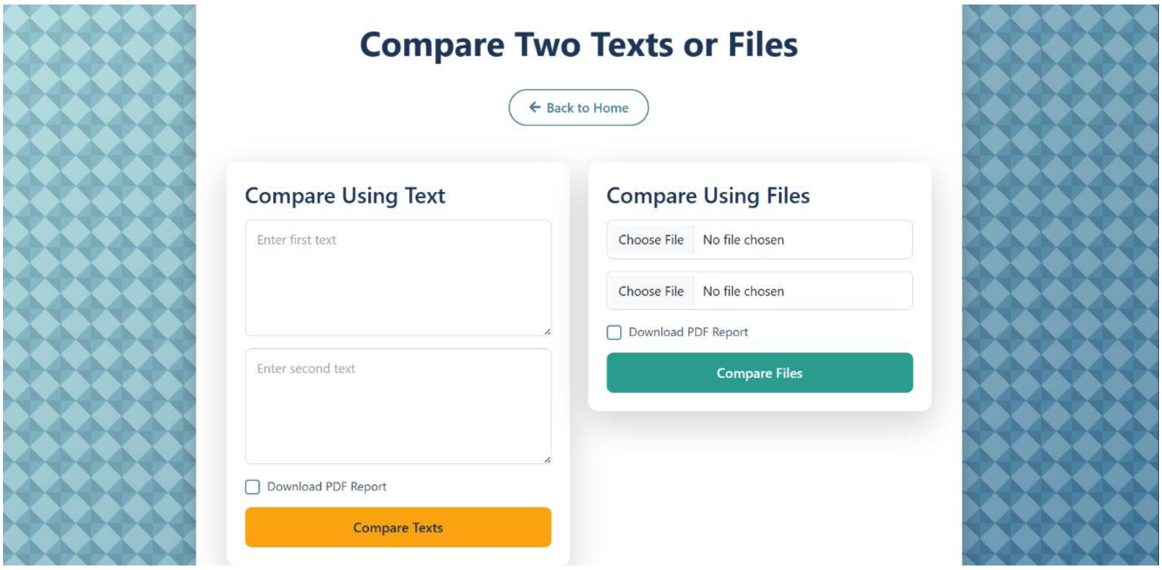
Jun 13, 2025 15:11
Text Input: A bicycle, also called a pedal cycle, bike, push-bike or cycle, is a human-powered or m...
Plagiarism: 78.66%

Jun 08, 2025 08:03
Text Input: An older man sits with his orange juice at a small table in a coffee shop



Plagiarism Detection System

Please enter your text and press Check Plagiarism



Compare Two Texts or Files

Compare Using Text

Enter first text

Enter second text

□ Download PDF Report

Compare Using Files

□ Download PDF Report

Source Code

```
{% extends "pc/base.html" %}{% load static %}{% load custom_filters %}
{% block title %}Home - Plagiarism Checker{% endblock %}
{% block head %}
    {{ block.super }}
    <style>
        body {
            background:
                linear-gradient(135deg, rgba(168, 218, 220, 0.8), rgba(69, 123, 157, 0.8)), /* Base gradient, slightly transparent */
                linear-gradient(45deg, #1d3557 25%, transparent 25%, transparent 75%, #1d3557 75%, #1d3557), /* Darker diagonal lines */
                linear-gradient(-45deg, #457b9d 25%, transparent 25%, transparent 75%, #457b9d 75%, #457b9d); /* Lighter diagonal lines */
            background-size:
                cover, /* Cover the entire area with the main gradient */
                40px 40px, /* Size of the diagonal line pattern */
                40px 40px; /* Size of the second diagonal line pattern */
            background-position:
                0 0, /* Position for the main gradient */
                0 0, /* Initial position for the first pattern */
                0 0; /* Initial position for the second pattern */
            background-attachment: fixed; /* Keep background fixed while scrolling */
            animation: moveBackground 60s linear infinite; /* Slow, smooth animation */
            font-family: 'Arial', sans-serif;
            color: #333;
            margin: 0;
            padding: 0;
            display: flex;
            flex-direction: column;
            min-height: 100vh;}
        @keyframes moveBackground {
            from {
                background-position: 0 0, 0 0, 0 0;}
            to {
                background-position: 0 0, 40px 40px, -40px 40px;}}
        .animated-title-container {
            text-align: center;
            margin-top: 50px;
            margin-bottom: 30px;
            padding: 30px; /* This was increased to make the box taller */
            background-color: rgba(255, 255, 255, 0.9);
            border-radius: 15px;
            box-shadow: 0 4px 20px rgba(0, 0, 0, 0.15);
            max-width: 1000px; /* This was increased to make the box wider */
            width: 95%; /* Added for responsiveness */
            margin-left: auto;
            margin-right: auto;
            box-sizing: border-box; /* Include padding in the total width/height */}
        .animated-title {
            font-size: 3.5em; /* Changed from 4.5em back to 3.5em */
            font-weight: bold;
    color: #1d3557; /* Dark blue from your palette */
    display: inline-block;
    overflow: hidden; /* Ensures the typing effect is visible */
    white-space: nowrap; /* Keeps the text on a single line */
    border-right: 3px solid #e63946; /* Blinking cursor effect */
    animation: typing 3s steps(20, end) forwards,
               blink-caret .75s step-end infinite;
    padding-right: 5px; /* Space for the caret */}
@keyframes typing {
    from { width: 0 }
    to { width: 100% }}
@keyframes blink-caret {
    from, to { border-color: transparent }
    50% { border-color: #e63946; }}
.main-content {
    background-color: rgba(255, 255, 255, 0.95);
    padding: 40px;
    border-radius: 15px;
    box-shadow: 0 4px 20px rgba(0, 0, 0, 0.15);
    max-width: 900px;
    width: 90%;
    margin: 30px auto; /* Centering with margin */
    text-align: center; /* Center buttons and text area */}
.text-input-area {
    width: 100%;
    height: 250px;
    padding: 15px;
    border: 2px solid #a8dadc; /* Light blue border */
    border-radius: 8px;
    font-size: 1.1em;
    line-height: 1.6;
    resize: vertical;
    margin-bottom: 25px;
    box-sizing: border-box; /* Include padding and border in the element's total width and height */
    transition: border-color 0.3s ease, box-shadow 0.3s ease;}
.text-input-area:focus {
    border-color: #457b9d; /* Darker blue on focus */
    box-shadow: 0 0 10px rgba(69, 123, 157, 0.3);
    outline: none;}
.bottom-buttons {
    display: flex;
    justify-content: space-between; /* Distribute items evenly */
    align-items: center;
    flex-wrap: wrap; /* Allow wrapping on smaller screens */
    margin-bottom: 25px;
    gap: 15px; /* Space between items */}
.btn-choose-file {
    background-color: #f1faee; /* Light background */
    color: #1d3557; /* Dark blue text */
    border: 2px solid #a8dadc; /* Light blue border */
    padding: 12px 20px;
    border-radius: 8px;
    cursor: pointer;
    font-size: 1em;
    font-weight: bold;
    display: inline-flex;
    align-items: center;
    gap: 8px;
    transition: background-color 0.3s ease, border-color 0.3s ease, color 0.3s ease;}
.btn-choose-file:hover {
    background-color: #e0f2f4; /* Slightly darker on hover */
    border-color: #457b9d;
    color: #2a9d8f;}
#selected-file-name {
    margin-top: 10px;
    font-size: 0.9em;
    color: #666;
    text-align: left; /* Align text to the left below the button */}
.delete-icon {
    background: none;
    border: none;
    color: #e63946; /* Red for delete */
    font-size: 1.8em;
    cursor: pointer;
    transition: color 0.3s ease;}
.delete-icon:hover {
    color: #c0392b;}
.btn-primary {
    background-color: #457b9d; /* Blue */
    color: white;
    border: none;
    padding: 12px 25px;
    border-radius: 8px;
    cursor: pointer;
    font-size: 1.1em;
    font-weight: bold;
    transition: background-color 0.3s ease, transform 0.2s ease;}
.btn-primary:hover {
    background-color: #1d3557; /* Darker blue on hover */
    transform: translateY(-2px);}
.btn-warning {
    background-color: #f4a261; /* Orange/Yellow */
    color: #1d3557; /* Dark blue text for contrast */
    border: none;
    padding: 12px 25px;
    border-radius: 8px;
    cursor: pointer;
    font-size: 1.1em;
    font-weight: bold;
    transition: background-color 0.3s ease, transform 0.2s ease;}
.btn-warning:hover {
    background-color: #e76f51; /* Reddish orange on hover */
    transform: translateY(-2px);
    color: white;}
```

```
.alert-info {
   background-color: #d1ecf1;
   color: #0c5460;
   padding: 15px;
   border-radius: 8px;
   margin-top: 25px;
   border: 1px solid #bee5eb;
   font-size: 1.1em;}
.alert-danger {
   background-color: #f8d7da;
   color: #721c24;
   padding: 15px;
   border-radius: 8px;
   margin-top: 25px;
   border: 1px solid #f5c6cb;
   font-size: 1.1em;}
.plagiarized-text {
   background-color: yellow;
   font-weight: bold;
   padding: 2px 0;}
hr {
   border: 0;
   height: 1px;
   background-image: linear-gradient(to right, rgba(0, 0, 0, 0), rgba(0, 0, 0, 0.2), rgba(0, 0, 0, 0));
   margin: 40px 0;}
.btn-outline-secondary {
   border: 2px solid #457b9d;
   color: #457b9d;
   background-color: transparent;
   padding: 10px 20px;
   border-radius: 8px;
   text-decoration: none;
   font-weight: bold;
   transition: background-color 0.3s ease, color 0.3s ease;}
.btn-outline-secondary:hover {
   background-color: #457b9d;
   color: white;}
.web-results-container {
   margin-top: 30px;
   text-align: left;
   background-color: #f0f8ff; /* Light blue background */
   padding: 20px;
   border-radius: 10px;
   box-shadow: 0 2px 10px rgba(0,0,0,0.1);}
.web-results-container h5 {
   color: #1d3557;
   margin-bottom: 15px;
   font-size: 1.4em;}
.web-results-list {
   list-style-type: none;
   padding: 0;
   margin: 0;}
```

```
    .web-results-item {
      margin-bottom: 15px;
      padding-bottom: 15px;
      border-bottom: 1px dashed #c0d0e0;}
    .web-results-item:last-child {
      border-bottom: none;
      margin-bottom: 0;
      padding-bottom: 0;}
    .web-results-item a {
      color: #457b9d;
      text-decoration: none;
      font-weight: bold;
      display: block;
      margin-bottom: 5px;
      word-break: break-all; /* Ensures long URLs wrap */}
    .web-results-item a:hover {
      text-decoration: underline;
      color: #2a9d8f;}
    .web-results-item p {
      color: #3b5066;
      font-size: 0.95em;
      margin: 0;}
  </style>{% endblock %}{% block content %}
<div class="animated-title-container">
  <div class="animated-title">Plagiarism Detection System</div></div>
<div class="main-content">
  <form method="POST" action="{% url 'detect' %}" enctype="multipart/form-data" id="plagiarism-
form">{% csrf_token %}
    <textarea id="q" name="q" class="text-input-area" placeholder="Please enter your text and press
Check Plagiarism">{{ q_text_input|default:"" }}</textarea>
    <div class="bottom-buttons"><div>
        <label for="docfile" class="btn-choose-file">
          <i class="fas fa-cloud-upload-alt"></i> Choose File
          <input type="file" id="docfile" name="docfile" accept=".txt,.docx,.pdf" hidden></label>
        <div id="selected-file-name">No file chosen</div></div>
      <button type="button" class="delete-icon" id="clear-text-button"><i class="fas fa-trash-alt"></i>
      </button>
      <button type="button" class="btn btn-primary" id="check-text-button">
        Check Plagiarism
      </button><button type="button" class="btn btn-primary" id="check-web-button">
        Check Web</button>
    </div>
    {# HIDDEN INPUT FOR ACTION TYPE #}
    <input type="hidden" name="action" id="submit_action" value="check_text">
  </form>
  {% if percent is not None %}
  <div class="alert alert-info text-center">
    <p>Similarity with dataset: <strong>{{ percent }}%</strong></p>
    {% if total_words and plag_words is not None %}
    <p>Total Words: <strong>{{ total_words }}</strong></p>
    <p>Plagiarized Words: <strong>{{ plag_words }}</strong></p>
    {% endif %}
    </div>{% if web_results %}
    <div class="web-results-container">
        <h5>Web Plagiarism Sources:</h5>
        {% if web_results.similarities %}
        <ul class="web-results-list">
            {% for url, similarity_score in web_results.similarities.items %}
                <li class="web-results-item">
                    <a href="{{ url }}" target="_blank" rel="noopener noreferrer">{{ url }}</a>
                    <p>Similarity: {{ similarity_score|floatformat:2 }}%</p>
                    {# You can add more details here if 'output' contained relevant snippets based on 'url' #}
                    {# Example: If output[url] contains a snippet field #}
                        {% with web_results.output|get_item:url as result %}
                            {% if result and result.snippet %}
                                <p>Snippet: {{ result.snippet }}</p>
                            {% endif %}
                        {% endwith %}
                </li>
            {% endfor %}</ul>{% else %}<p>No specific web sources found with significant similarity.</p>
{% endif %} </div>
    {% endif %}
    <form method="POST" action="{% url 'detect' %}" class="mt-2">
        {% csrf_token %}
        <textarea name="q" hidden>{{ q_text_input|default:"" }}</textarea>
        <input type="hidden" name="action" value="{{ last_action|default:"check_text" }}">
        <button type="submit" name="download_pdf" value="1" class="btn btn-warning w-100">
            Download PDF Report
        </button>
    </form>{% endif %}{% if error %}
    <div class="alert alert-danger text-center mt-4">{{ error }}</div>
    {% endif %}<hr>
    <div style="margin-top: 30px; text-align: center;">
    {% if request.user.is_authenticated %}
        <a href="{% url 'webpage' %}" class="btn btn-outline-secondary" style="max-width: 250px; margin: 0 auto;">Back to Home</a>{% else %}
        <a href="{% url 'home' %}" class="btn btn-outline-secondary" style="max-width: 250px; margin: 0 auto;">Back to Home</a>{% endif %}</div>
</div>{% endblock %}{% block extra_js %}
<script>
    document.addEventListener('DOMContentLoaded', function() {
        const textArea = document.getElementById('q');
        const docFileInput = document.getElementById('docfile');
        const plagiarismForm = document.getElementById('plagiarism-form');
        const clearTextButton = document.getElementById('clear-text-button');
        const checkTextButton = document.getElementById('check-text-button');
        const checkWebButton = document.getElementById('check-web-button');
        const animatedTitle = document.querySelector('.animated-title');
        const submitActionHiddenInput = document.getElementById('submit_action');
        const selectedFileNameDisplay = document.getElementById('selected-file-name');
        const typingDurationMs = 3 * 1000;
        setTimeout(() => {
            animatedTitle.style.borderRight = 'none';
        }, typingDurationMs);
        docFileInput.addEventListener('change', function() {
            if (this.files.length > 0) {
                selectedFileNameDisplay.textContent = `File chosen: ${this.files[0].name}`;
                textArea.value = ''; // Clear textarea if a file is uploaded
            } else {
                selectedFileNameDisplay.textContent = 'No file chosen';
            }});
        checkTextButton.addEventListener('click', function(event) {
            if (docFileInput.files.length > 0) {
                submitActionHiddenInput.value = 'check_file';
                plagiarismForm.submit();
            } else if (textArea.value.trim() !== '') {
                submitActionHiddenInput.value = 'check_text';
                plagiarismForm.submit();
            } else {
                event.preventDefault();
                alert('Please enter text or choose a file to check for plagiarism.');
            }});
        checkWebButton.addEventListener('click', function(event) {
            if (textArea.value.trim() !== '') {
                submitActionHiddenInput.value = 'check_web';
                plagiarismForm.submit();
            } else {
                event.preventDefault();
                alert('Please enter text to check against the web.');
            }});
        clearTextButton.addEventListener('click', function() {
            textArea.value = '';
            docFileInput.value = '';
            selectedFileNameDisplay.textContent = 'No file chosen';
            submitActionHiddenInput.value = 'check_text';
        });
    });
</script>
{% endblock %
```