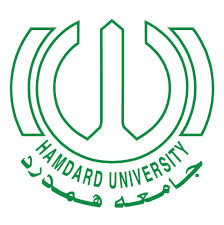
Hamdard University

Department of Computing

Final Year Project



**PLAGIARISM DETECTION SYSTEM FOR URDU LANGUAGE**

**(FYP-027/FL24)**

**Software Design Specifications**

Submitted by

SYED AHMED ALI (2579-2021)

SYED SHAHEER Ul HAQUE (2197-2021)

Supervisor(s)

Mr. AAMIR HUSSAIN

**Fall 2025**

**Document Sign off Sheet**

### Document Information

|  |  |
| --- | --- |
| **Project Title** | PLAGIARISM DETECTION SYSTEM FOR URDU LANGUAGE |
| **Project Code** | FYP-027/FL24 |
| **Document Name** | Software Design Specifications |
| **Document Version** | <1.0> |
| **Document Identifier** | FYP-027/FL24-SDS |
| **Document Status** | Final |
| **Author(s)** | SYED AHMED ALI  SYED SHAHEER Ul HAQUE |
| **Approver(s)** | Mr AAMIR HUSSAIN |
| **Issue Date** | 22/MAY/2024 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Signature** | **Date** |
| SYED AHMED ALI | Team Lead | C:\Users\DELL\Downloads\FINAL FYP\Plagiarism mom\Syed Ahmed.png | 6-July-25 |
| SYED SHAHEER Ul HAQUE | Team Member 2 | C:\Users\DELL\Downloads\FINAL FYP\Plagiarism mom\Syed Shaheer.png | 6-July-25 |
| Mr. AAMIR HUSSAIN | Supervisor | **C:\Users\DELL\Downloads\FINAL FYP\Plagiarism mom\Sir Aamir.png** | 6-July-25 |
|  | Co-Supervisor |  |  |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 11-JAN-25 | 1.0 | Final Draft of the SRS Document for Submission | Syed Shaheer/ Ahmed Ali |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Definition of Terms, Acronyms, and Abbreviations**

|  |  |
| --- | --- |
| **Term** | **Description** |
| Report ID | A unique identifier assigned to each plagiarism report generated by the system. |
| Document ID | A unique identifier assigned to each document uploaded by the user. |
| User ID | A unique identifier for each user registered in the system. |
| PK (Primary Key) | A database key used to uniquely identify a record in a table. |
| FK (Foreign Key) | A database key that links a record to another table’s primary key. |
| Similarity Score | The percentage result indicating similarity between uploaded and reference content. |
| Admin | A system role with higher privileges, responsible for monitoring and management. |
| Source Database | A collection of files or online references used for comparison in plagiarism detection. |

# Table of Contents

[**1 Introduction** 5](#_Toc187589659)

[1.1 Purpose of Document 5](#_Toc187589660)

[1.2 Intended Audience 5](#_Toc187589661)

[1.3 Document Convention 5](#_Toc187589662)

[1.4 Project Overview 5](#_Toc187589663)

[1.5 Scope 6](#_Toc187589664)

[2 Design Considerations 7](#_Toc187589665)

[2.1 Assumptions and Dependencies 7](#_Toc187589666)

[2.2 Risks and Volatile Areas 7](#_Toc187589667)

[3 System Architecture 8](#_Toc187589668)

[3.1 System Level Architecture 8](#_Toc187589669)

[3.2 Software Architecture 8](#_Toc187589670)

[4 Design Strategy 9](#_Toc187589671)

[5 Detailed System Design 10](#_Toc187589672)

[5.1 Design Class Diagram 10](#_Toc187589673)

[5.2 Database Design 11](#_Toc187589674)

[5.2.1 ER Diagram 11](#_Toc187589675)

[5.2.2 Data Dictionary 12](#_Toc187589676)

[5.2.2.1 User 12](#_Toc187589677)

[5.2.2.2 Document 13](#_Toc187589678)

[5.2.2.3 Plagiarism\_Report 14](#_Toc187589679)

[5.2.2.4 Database\_Source 15](#_Toc187589680)

[5.3 Application Design 16](#_Toc187589681)

[5.3.1 Sequence Diagram 16](#_Toc187589682)

[5.3.1.1 Sequence Diagram 1: User Login and Plagiarism Check 16](#_Toc187589683)

[5.3.1.2 Sequence Diagram 2: Document Upload and Report Generation 17](#_Toc187589684)

[5.3.2 State Diagram 18](#_Toc187589685)

[5.3.2.1 State Diagram 1: User Session 18](#_Toc187589686)

[5.3.2.2 State Diagram 2: Document Processing 19](#_Toc187589687)

[5.4 GUI Design 20](#_Toc187589688)

[5.4.1 <Use Case Name - Mock Screen 1> 20](#_Toc187589689)

[5.4.2 <Use Case Name - Mock Screen 2> 20](#_Toc187589690)

[5.4.3 <Use Case Name - Mock Screen 3> 20](#_Toc187589691)

[6 References 21](#_Toc187589692)

[7 Appendices 22](#_Toc187589693)

# Introduction

The project aims to develop a plagiarism detection system specifically for the Urdu language. This system will be accessible through mobile applications, providing a comprehensive tool for educators, students, and professionals to ensure the originality of their Urdu texts. The project addresses the gap in existing plagiarism detection tools, which predominantly focus on widely spoken languages and often overlook regional languages like Urdu.

## Purpose of Document

The purpose of the Software Design Specification (SDS) document is to provide a comprehensive blueprint for the development of the Urdu Plagiarism Detection System. It outlines the architectural design, system components, and detailed functionalities required to build the software effectively. This document serves as a guide for developers, project stakeholders, and supervisors, ensuring a shared understanding of the project’s design objectives.

## Intended Audience

This document is intended for the following stakeholders:

* Development Team
* Supervisors (Mr. Aamir Hussain, Dr. Shahid Muneer)
* End Users (educators, students, and professionals working with Urdu content)
* Academic and administrative decision-makers

## Document Convention

This document uses **Times New Roman 12pt** font for body text and **Arial** **14pt bold** for headings, with 1.5 line spacing for paragraphs. Sections are numbered hierarchically (e.g., 1, 1.1) for easy navigation, and figures/tables are labeled and referenced clearly. Acronyms and technical terms are defined in the **Definitions** section, and all updates are tracked in the **Revision History**. Formal language is used to ensure clarity and consistency for all readers.

This document uses the following conventions and definitions, as described in   
[**Appendix A: Glossary of Terms**](#_Appendix_A:_Glossary).

## Project Overview

The **Urdu Plagiarism Detection System** ensures content originality by analyzing Urdu text for similarities using advanced Natural Language Processing (NLP) techniques. Its core functionality includes text preprocessing, similarity detection, and database comparison. The system adopts a modular design approach with a Python-based backend and a user interface. This ensures scalability, efficiency, and an intuitive user experience.

## Scope

**In Scope**:

* Detect plagiarism in Urdu text documents.
* Focus exclusively on Urdu language.
* Utilize NLP and machine learning techniques for high accuracy.
* Provide a mobile-friendly interface for user accessibility.

**Not In Scope**:

* Cross-language plagiarism detection (e.g., between Urdu and English texts).
* Support for plagiarism detection in languages other than Urdu.
* Offline functionality without stable internet access for database queries.

# Design Considerations

This section identifies the foundational issues that must be resolved to ensure a robust system design for the Urdu Plagiarism Detection System.

## Assumptions and Dependencies

The design assumes stable internet connectivity for database access, a comprehensive and regularly updated Urdu text corpus, and reliance on NLP tools like UrduHack and TensorFlow for text processing and algorithm implementation. Dependencies include the availability of computational resources and user-provided text in a machine-readable Urdu script.

## Risks and Volatile Areas

Key risks include challenges with the quality and completeness of the Urdu text database, the complexity of handling linguistic nuances in Urdu, and evolving user requirements or technologies. The system will employ modular architecture and iterative updates to address these risks, ensuring scalability and adaptability to changes.

The system's hardware and software requirements are outlined in   
[**Appendix B: System Requirements**](#_Appendix_B:_System).

# System Architecture

This section provides an overview of the system's structure, showing how the responsibilities are divided among components and how they interact to deliver functionality.

## System Level Architecture

The system is divided into the following high-level components:

1. **System Decomposition:**
   * **User Interface Layer:** Handles user interactions through a mobile application.
   * **Backend Module:** Processes text input, detects plagiarism, and manages application logic.
   * **Database Layer:** Stores the Urdu text corpus and manages queries.
2. **Relationships Between Elements:**
   * The User Interface sends text input to the Backend Module.
   * The Backend processes data and communicates with the Database Layer to retrieve matching results.
3. **Interfaces to External Systems:**
   * Internet connectivity for querying remote databases and NLP library APIs.
4. **Major Physical Design Issues:**
   * The Backend executes on cloud servers for scalability, while the User Interface operates on mobile devices.
5. **Global Design Strategies:**
   * Centralized error handling within the Backend ensures stability and clear feedback to the user.

## Software Architecture

The software design is structured into the following layers:

1. **User Interface Layer:**
   * Developed in VS Code using HTML, CSS, JavaScript, this layer facilitates user input and displays plagiarism detection results.
2. **Middle Tier:**
   * A Python-based service layer that performs preprocessing (tokenization, stemming) and compares text using NLP algorithms.
3. **Data Access Layer:**
   * A SQL-based layer that interacts with the Urdu text database for data retrieval and updates.

The database schema is described in detail in [**Appendix C: Database Schema**](#_Appendix_C:_Database).

**Interaction Overview:**

The User Interface Layer sends input data to the Middle Tier via APIs. The Middle Tier processes the data, queries the Data Access Layer for matching results, and sends the analysis back to the User Interface for display.

# Design Strategy

The design strategy of the **Urdu Plagiarism Detection System** focuses on modularity, scalability, and maintainability to support current functionality while allowing for future enhancements.

1. **Future System Extension or Enhancement:**
   * The modular architecture ensures that additional features, such as cross-language detection or offline functionality, can be integrated without major restructuring.
2. **System Reuse:**
   * Core components like the NLP models and similarity algorithms are designed for reuse in other language detection systems with minimal adjustments.
3. **User Interface Paradigms:**
   * The mobile application follows a user-centered design to provide an intuitive interface, leveraging clear workflows for text input and plagiarism analysis.
4. **Data Management:**
   * A centralized database ensures efficient storage and retrieval of the Urdu text corpus, with provisions for periodic updates to maintain relevance.
5. **Concurrency and Synchronization:**
   * The system uses asynchronous processing for handling multiple user requests simultaneously, ensuring responsive and consistent performance across concurrent sessions.

This approach balances flexibility for growth with performance and usability, meeting both current and future user needs.

# Detailed System Design

## Design Class Diagram

* **Purpose**: Represents the structure of the plagiarism detection system. It defines the main classes and their relationships.

1. **Main Classes**:
   * **User**: Handles user data such as login credentials.
   * **Plagiarism Checker**: Core functionality for text processing and similarity detection.
   * **Database Manager**: Manages database interactions.
   * **Text Processor**: Handles text preprocessing, tokenization, and stemming.
   * **Similarity Algorithm**: Implements cosine similarity and sequence matching algorithms and others.
2. **Attributes and Methods**:
   * **User**:
     + Attributes: userID, name, password
     + Methods: register(), login(), updateProfile()
   * **PlagiarismChecker**:
     + Attributes: inputText, result
     + Methods: checkPlagiarism(), generateReport()
   * **DatabaseManager**:
     + Attributes: databaseConnection
     + Methods: connect(), fetchData(), storeData()
   * **TextProcessor**:
     + Attributes: textData
     + Methods: tokenize(), stem(), lemmatize()
   * **SimilarityAlgorithm**:
     + Attributes: algorithmType
     + Methods: calculateSimilarity()

**Logical Data Model (E/R Model)**

* **Entities**: User, Document, SimilarityResult
* **Relationships**:
  + User uploads Document.
  + Document has SimilarityResult.

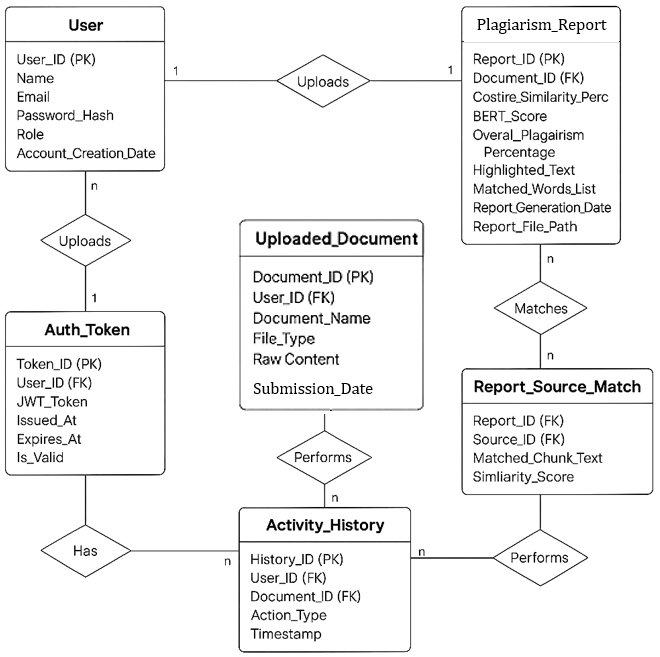
**Detailed GUIs**

* **Login Page**: Allows users to authenticate.
* **Home Screen**: Main dashboard with options for uploading and viewing reports.
* **Plagiarism Results**: Displays percentage of similarity and plagiarized sections.

## Database Design

### ER Diagram

**Entities and Attributes**



**1. User**

* **User\_ID (PK) – Primary Key**
* **Name**
* **Email**
* **Password\_Hash**
* **Role**
* **Account\_Creation\_Date**

**2. Auth\_Token**

* **Token\_ID (PK) – Primary Key**
* **User\_ID (FK) – Foreign Key referencing User**
* **JWT\_Token**
* **Issued\_At**
* **Expires\_At**
* **Is\_Valid**

**3. Uploaded\_Document**

* **Document\_ID (PK) – Primary Key**
* **User\_ID (FK) – Foreign Key referencing User**
* **Document\_Name**
* **File\_Type**
* **Raw\_Content**
* **Submission\_Date**

**4. Plagiarism\_Report**

* **Report\_ID (PK) – Primary Key**
* **Document\_ID (FK) – Foreign Key referencing Uploaded\_Document**
* **Costire\_Similarity\_Perc**
* **BERT\_Score**
* **Overal\_Plagiarism\_Percentage**
* **Highlighted\_Text**
* **Matched\_Words\_List**
* **Report\_Generation\_Date**
* **Report\_File\_Path**

**5. Report\_Source\_Match**

* **Report\_ID (FK) – Foreign Key referencing Plagiarism\_Report**
* **Source\_ID (FK) – Foreign Key (the source table/entity is not shown in the diagram)**
* **Matched\_Chunk\_Text**
* **Similarity\_Score**

**6. Activity\_History**

* **History\_ID (PK) – Primary Key**
* **User\_ID (FK) – Foreign Key referencing User**
* **Document\_ID (FK) – Foreign Key referencing Uploaded\_Document**
* **Action\_Type**
* **Timestamp**

**Relationships**

**1. User ↔ Uploaded\_Document**

* **Relationship:** One-to-Many (1:n)
* **Explanation**: A single user can upload multiple documents, but each document belongs to only one user.
* **Keys Involved**:
  + User.User\_ID (PK)
  + Uploaded\_Document.User\_ID (FK)

**2.** **User ↔ Auth\_Token**

* **Relationship**: One-to-Many (1:n)
* **Explanation**: A user can have multiple authentication tokens (e.g., multiple logins).
* **Keys Involved**:
  + User.User\_ID (PK)
  + Auth\_Token.User\_ID (FK)

**3.** **Uploaded\_Document ↔ Plagiarism\_Report**

* **Relationship**: One-to-One (1:1)
* **Explanation**: Each uploaded document generates one plagiarism report.
* **Keys Involved**:
  + Uploaded\_Document.Document\_ID (PK)
  + Plagiarism\_Report.Document\_ID (FK)

**4.** **Plagiarism\_Report ↔ Report\_Source\_Match**

* **Relationship**: One-to-Many (1:n)
* **Explanation**: A single plagiarism report can be matched against multiple source chunks.
* **Keys Involved**:
  + Plagiarism\_Report.Report\_ID (PK)
  + Report\_Source\_Match.Report\_ID (FK)

**5.** **Uploaded\_Document ↔ Activity\_History**

* **Relationship**: One-to-Many (1:n)
* **Explanation**: One document can have multiple user actions associated with it (uploads, checks, downloads, etc.).
* **Keys Involved**:
  + Uploaded\_Document.Document\_ID (PK)
  + Activity\_History.Document\_ID (FK)

**6.** **User ↔ Activity\_History**

* **Relationship**: One-to-Many (1:n)
* **Explanation**: A user can perform many actions across various documents.
* **Keys Involved**:
  + User.User\_ID (PK)
  + Activity\_History.User\_ID (FK)

**7.** **Report\_Source\_Match ↔ (Source\_ID)**

* **Relationship**: Many-to-One or Many-to-Many depending on implementation
* **Explanation**: Each matched chunk in the report refers to a source (not defined in the diagram).
* **Note**: Source\_ID is a foreign key, but its reference table is **not included** in the current ERD. You may want to define a Source\_Document or similar table.

The API documentation for the plagiarism detection system is provided in   
[**Appendix D: API Documentation**](#_Appendix_D:_API).

### Data Dictionary

**1. Name: User**

**Alias**: UserAccount  
**Where-used/how-used**:  
• Input to processes such as user authentication, uploading documents.  
• Output as user details for activity logs and administration.  
• Stored in the database.

**Content description**:  
The user data is composed of the following attributes:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Type** | **Length** | **Nullable** | **Default Value** | **Key Type** |
| User\_ID | Unique identifier for the user | Int | 10 | No | AUTO\_INCREMENT | PK |
| Name | Full name of the user | Varchar | 100 | No | NULL |  |
| Email | User’s email address | Varchar | 150 | No | NULL |  |
| Password\_Hash | Hashed password for secure login | Varchar | 255 | No | NULL |  |
| Role | User’s role (Student, Teacher, Admin) | Varchar | 50 | No | NULL |  |
| Account\_Creation\_Date | Date the account was created | DateTime | - | No | CURRENT\_TIMESTAMP |  |

**2. Name: Auth\_Token**

**Alias**: SessionToken  
**Where-used/how-used**:  
• Used to manage login sessions using JWT.  
• Stored for validating user sessions.

**Content description**:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Type** | **Length** | **Nullable** | **Default Value** | **Key Type** |
| Token\_ID | Unique ID for the token | Int | 10 | No | AUTO\_INCREMENT | PK |
| User\_ID | Foreign key from User table | Int | 10 | No |  | FK |
| JWT\_Token | JWT session token | Varchar | 500 | No | NULL |  |
| Issued\_At | Time the token was generated | DateTime | - | No | CURRENT\_TIMESTAMP |  |
| Expires\_At | Expiry time of the token | DateTime | - | No |  |  |
| Is\_Valid | Status of the token (valid/invalid) | Boolean | - | No | TRUE |  |

**3. Name: Uploaded\_Document**

**Alias**: UserFile  
**Where-used/how-used**:  
• Input to plagiarism detection system.  
• Stored for analysis history.

**Content description**:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Type** | **Length** | **Nullable** | **Default Value** | **Key Type** |
| Document\_ID | Unique document ID | Int | 10 | No | AUTO\_INCREMENT | PK |
| User\_ID | Foreign key to User | Int | 10 | No |  | FK |
| Document\_Name | Name of the uploaded document | Varchar | 255 | No | NULL |  |
| File\_Type | Type of document (pdf, docx, txt) | Varchar | 50 | No | NULL |  |
| Raw\_Content | Extracted text from the document | Text | - | No | NULL |  |
| Submission\_Date | Date document was uploaded | DateTime | - | No | CURRENT\_TIMESTAMP |  |

**4. Name: Plagiarism\_Report**

**Alias**: ReportData  
**Where-used/how-used**:  
• Output after running plagiarism analysis.  
• Stored and referenced in report generation.

**Content description**:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Type** | **Length** | **Nullable** | **Default Value** | **Key Type** |
| Report\_ID | Unique ID of the report | Int | 10 | No | AUTO\_INCREMENT | PK |
| Document\_ID | Foreign key to Uploaded\_Document | Int | 10 | No |  | FK |
| Cosine\_Similarity\_Perc | Cosine similarity percentage | Float | - | No | 0.0 |  |
| BERT\_Score | Semantic similarity score from BERT | Float | - | No | 0.0 |  |
| Overall\_Plagiarism\_Percentage | Combined plagiarism score | Float | - | No | 0.0 |  |
| Highlighted\_Text | Sections marked as plagiarized | Text | - | Yes | NULL |  |
| Matched\_Words\_List | List of matched words or phrases | Text | - | Yes | NULL |  |
| Report\_Generation\_Date | Date the report was generated | DateTime | - | No | CURRENT\_TIMESTAMP |  |
| Report\_File\_Path | Path to the generated PDF report | Varchar | 255 | No | NULL |  |

**5. Name: Report\_Source\_Match**

**Alias**: MatchedChunk  
**Where-used/how-used**:  
• Stores matched content from the report source.  
• Referenced in report generation for highlighting.

**Content description**:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Type** | **Length** | **Nullable** | **Default Value** | **Key Type** |
| Report\_ID | Foreign key to Plagiarism\_Report | Int | 10 | No |  | FK |
| Source\_ID | Foreign key to source document (TBD) | Int | 10 | No |  | FK |
| Matched\_Chunk\_Text | Text chunk matched with user document | Text | - | Yes | NULL |  |
| Similarity\_Score | Similarity score of this match | Float | - | No | 0.0 |  |

**6. Name: Activity\_History**

**Alias**: UserActionLog  
**Where-used/how-used**:  
• Logs actions like uploads, checks, downloads.  
• Useful for user tracking and analysis.

**Content description**:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Type** | **Length** | **Nullable** | **Default Value** | **Key Type** |
| History\_ID | Unique identifier for the action log | Int | 10 | No | AUTO\_INCREMENT | PK |
| User\_ID | Foreign key referencing User | Int | 10 | No |  | FK |
| Document\_ID | Foreign key referencing document | Int | 10 | No |  | FK |
| Action\_Type | Type of action performed | Varchar | 100 | No | NULL |  |
| Timestamp | Date and time of the action | DateTime | - | No | CURRENT\_TIMESTAMP |  |

The notation to develop content description is given below:

|  |  |  |
| --- | --- | --- |
| **Data construct** | **Notation** | **Meaning** |
| Sequence | + | And |
| Selection | `[ | ]` |
| Repetition | { }n | n repetitions |
| Optional Data | ( ) | Optional data |
| Comment | \* … \* | Delimits comments |

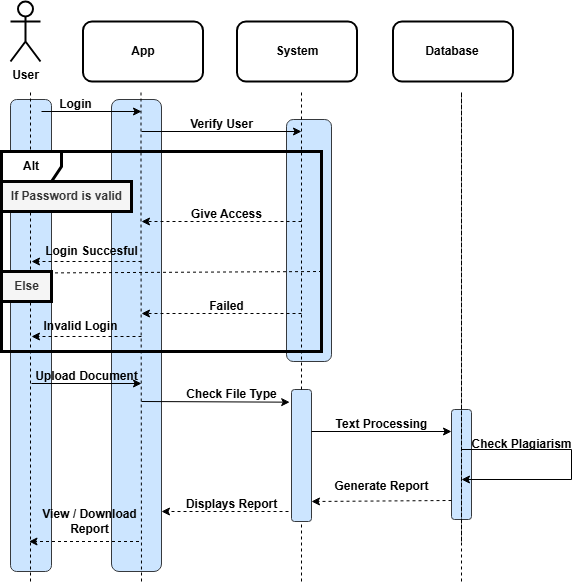
## Application Design

The application design section outlines the dynamic interactions and state transitions within the system, focusing on the flow of user interactions and system responses.

### Sequence Diagram

Sequence diagrams illustrate the order of interactions between objects or components over time. Below are the diagrams that depict the interactions in various user scenarios.

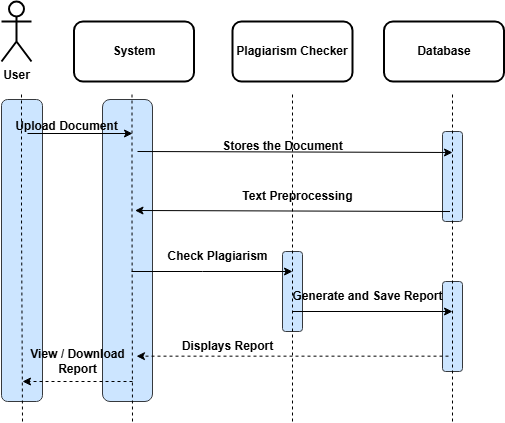
#### Sequence Diagram 1: User Login and Plagiarism Check



**Explanation:**  
This diagram demonstrates the interaction flow when a user logs in and uploads a document for plagiarism checking.

* The user enters their credentials and submits the login form.
* The system verifies the credentials.
* If the login is successful, the user accesses the home screen. Else Return error.
* The user uploads a document for plagiarism detection.
* The backend processes the document and performs text preprocessing.
* The plagiarism checker compares the document against the database.
* The system generates a plagiarism report and displays the results on the home screen which the user can either view and download.

#### Sequence Diagram 2: Document Upload and Report Generation



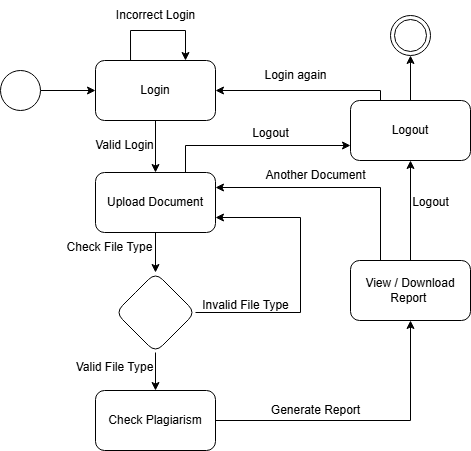
**Explanation:**  
This diagram illustrates the interaction when a user uploads a document and a plagiarism report is generated.

* The user selects and uploads a document.
* The system stores the document in the database.
* The system runs the text preprocessing functions (tokenization, stemming).
* The plagiarism checker compares the document with existing sources in the database.
* A plagiarism report is generated and saved in the database.
* The report is displayed on the user interface with the percentage of plagiarism detected.

### State Diagram

State diagrams are used to model the different states an object can be in throughout its lifecycle.

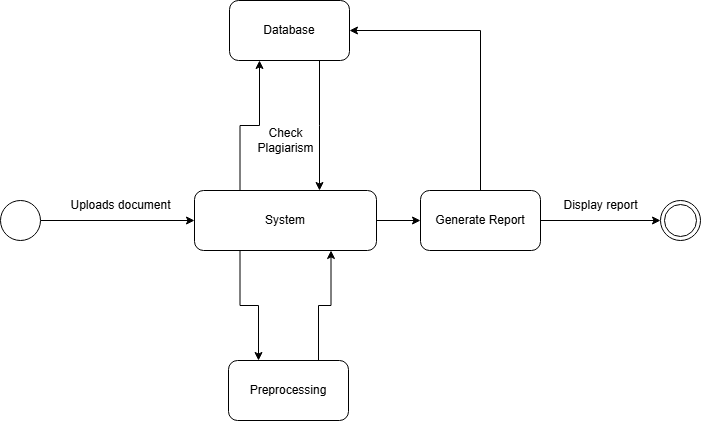
#### State Diagram 1: User Session



**Explanation:**  
This diagram shows the states that a user can experience during their interaction with the application.

* **Initial State:** The user opens the application.
* **State 1:** The user is prompted to log in.
* **State 2:** After successful login, the user is redirected to the home screen. Else return to login with an error.
* **State 3:** The user uploads a document.
* **State 4:** The system will check the file type. If valid then proceed with next state else return to home screen with an error.
* **State 5:** The document is processed for plagiarism.
* **State 6:** Once the report is generated, the user can view or download the result.
* **Final State:** The user logs out or closes the application.

#### State Diagram 2: Document Processing

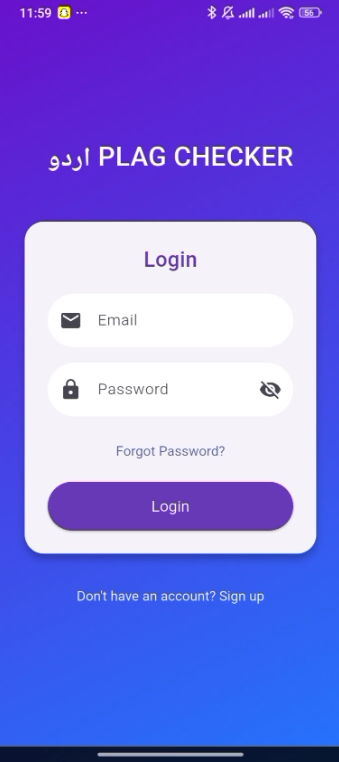


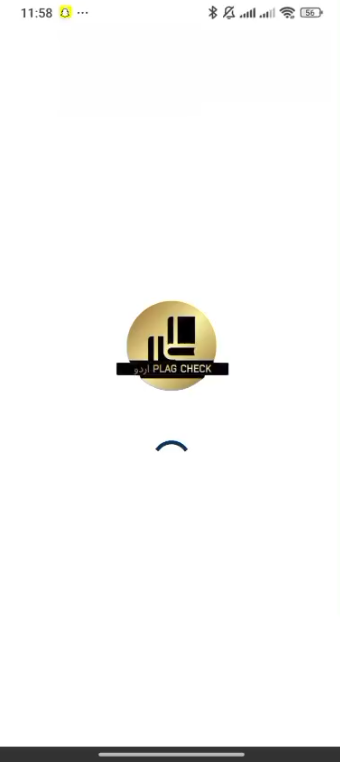
**Explanation:**  
This diagram illustrates the state transitions of the document during processing.

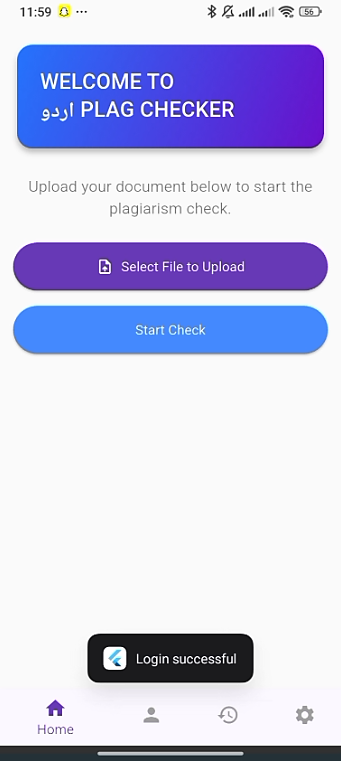
* **Initial State:** A document is uploaded by the user.
* **State 1:** The document undergoes preprocessing (tokenization and stemming).
* **State 2:** The system compares the document with existing data in the database.
* **State 3:** Whether plagiarism is found or not, the report will be generated.
* **State 4:** The plagiarism report is saved in the database.
* **Final State:** The user is notified with the report results.

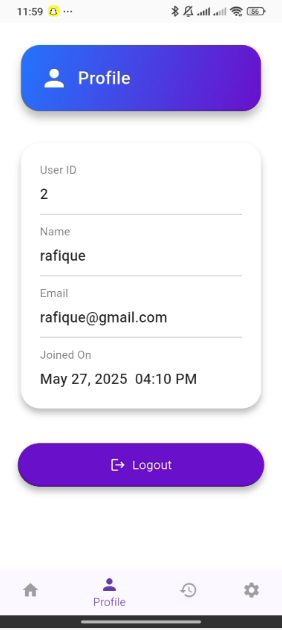
## GUI Design

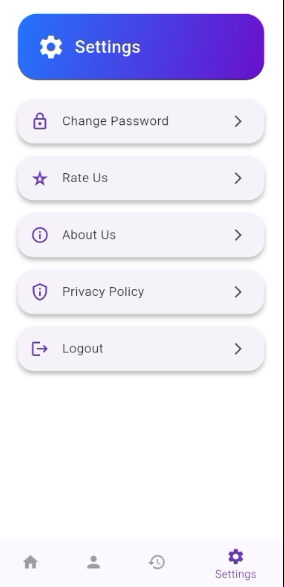
Following are the GUI designs for Home, Profile, Settings, Login, Sign up, Plagiarism Checker, History, Plagiarism Report and Rate us.

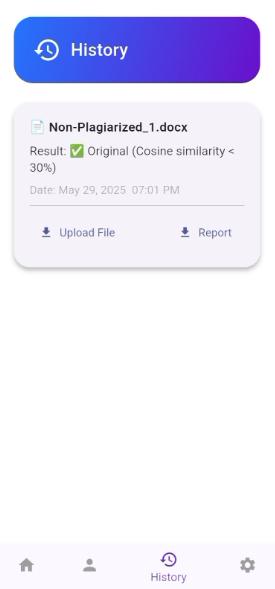
**Start Page**  **Login Page**



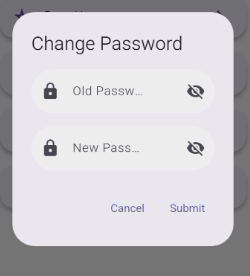
**Home Page Profile Page**

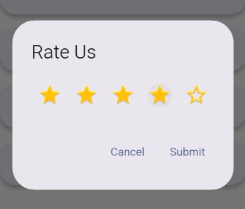


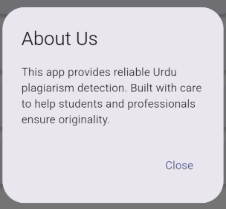
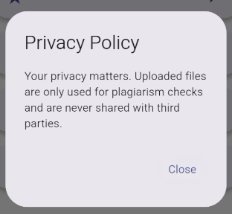
**History Page Settings**

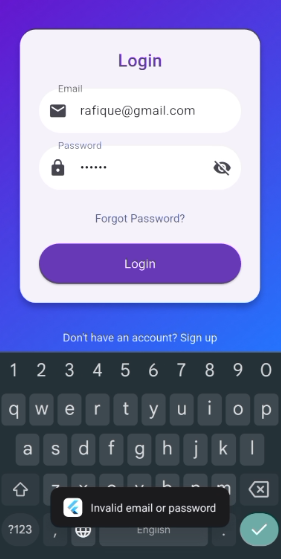
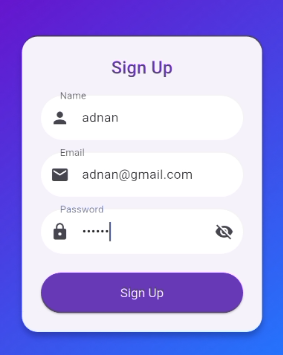
****

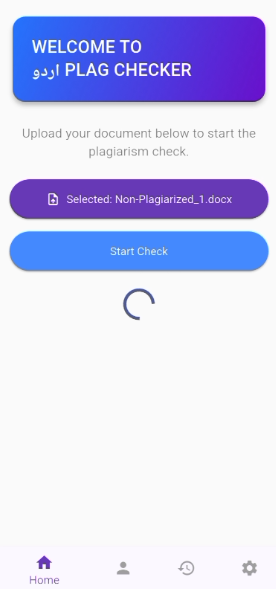
**Change Password Rate Us**

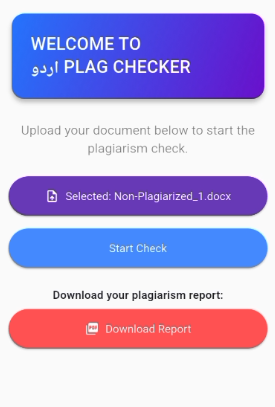
****

****

**About Us Privacy Policy**

**Invalid Login Sign Up**

**Upload File Download PDF Report**

****

# References

* "Plagiarism Detection in Urdu Documents using Sentence Structure Analysis" by S. M. Akram Shah et al.
* "Urdu Plagiarism Detection using Statistical Features" by M. Naveed Iqbal et al.
* "Plagiarism Detection in Urdu Language Documents Using Shallow Semantic Parsing" by M. Yasir Khan et al.

# Appendices

## Appendix A: Glossary of Terms

* **Plagiarism:** The act of passing off someone else's work as one's own.
* **Natural Language Processing (NLP):** A subfield of artificial intelligence that deals with the interaction between computers and humans in natural language.
* **Tokenization:** The process of breaking down text into individual words or tokens.
* **Stemming:** The process of reducing words to their base or root form.
* **Cosine Similarity:** A measure of similarity between two vectors that is often used in text similarity analysis.
* **Sequence Matching:** A technique used to compare two sequences of text to determine their similarity.

## Appendix B: System Requirements

* Hardware Requirements:
  + Mobile device with Android 10+ or iOS 14+
  + Minimum 2GB RAM
  + Minimum 16GB storage
* Software Requirements:
  + Mobile application developed using Java or Swift
  + Integration with NLP libraries and APIs

## Appendix C: Database Schema

The database schema for the plagiarism detection system consists of the following tables:

1. Users Table:

- User ID (primary key)

- Username

- Password

- Email

2. Documents Table:

- Document ID (primary key)

- User ID (foreign key)

- Document Name

- Document Content

- Submission Date

3. Plagiarism Reports Table:

- Report ID (primary key)

- Document ID (foreign key)

- Plagiarism Percentage

- Similarity Details

- Generated Date

## Appendix D: API Documentation

The plagiarism detection system provides the following APIs:

1. Upload Document API:

- Endpoint: /upload-document

- Method: POST

- Parameters: document\_name, document\_content

- Response: document\_id

2. Check Plagiarism API:

- Endpoint: /check-plagiarism

- Method: POST

- Parameters: document\_id

- Response: plagiarism\_percentage, similarity\_details