

Information Verification System.

Web Based Application



By:

Muhammad Taha.

Supervised By:

Dr. Ayyaz Hussain.

**Department of Computer Science,
Quaid-i-Azam University Islamabad.
Batch (2019-23).**

ACKNOWLEDGEMENT

First and foremost, I wish to express my profound appreciation to **Allah Almighty** for bestowing upon me the resilience, wisdom, skills, and liberty to embark on and successfully conclude this venture. Without His grace, this accomplishment would have remained elusive. I extend my genuine thanks to my respected supervisor, **Dr. Ayyaz Hussain**. His counsel and mentorship have been indispensable to me throughout this odyssey.

My gratitude would be incomplete without conveying sincere thanks to my steadfast pillar of support, my family, and the distinguished faculty members of the Department of Computer Science at Quaid-i-Azam University (QAU). I am deeply beholden to them for their priceless contributions. I also want to convey my appreciation to my fellow classmates and friends, whose unwavering encouragement and aid have played a pivotal role in my pursuits. I am sincerely thankful to all those who have contributed significantly to my academic and personal development during this journey.

ABSTRACT

In response to persistent challenges associated with discrepancies in academic information during the admission process at QAU, we present an innovative Information Verification System. This software introduces an advanced verification mechanism that seamlessly compares details provided by students in the admission form with information extracted from uploaded documents.

The verification process is initiated by clicking the 'verify' button, prompting the system to employ advanced data extraction techniques from the uploaded documents. The extracted data is meticulously compared with the information in the admission form. In the case of a successful match, the form is promptly submitted. However, if any disparities are detected, the system generates an immediate notification, highlighting specific field misalignments with the data in the mark sheet or document.

This solution aims to streamline the admission process at QAU, ensuring accuracy and transparency in verifying academic information. It specifically addresses historical issues related to mismatched marks, providing a robust and efficient system for validating data during admissions. This innovative Information Verification System serves as a pivotal tool in enhancing the overall reliability and efficiency of the academic admission process at QAU.

Table of Contents

ACKNOWLEDGEMENT 2

ABSTRACT:..... 3

Chapter 1: SOFTWARE PROJECT MANAGEMENT PLAN (SPMP) 7

1.1- Introduction: 8

1.1-1. Project Overview:..... 8

1.1-2. Project Deliverables: 8

1.2- Project Organization: 8

1.2-1. Software Process Model: 8

1.2-2. Roles and Responsibilities:..... 10

1.2-3. Tools and Techniques: 10

1.3- Project Management Plan: 10

1.3-1. Tasks: 10

1.3-2. Description: 11

1.4- Summary: 12

CHAPTER 2: SOFTWARE REQUIREMENTS SPECIFICATION (SRS) 13

2.1-Introduction: 14

Purpose: 14

Scope: 14

Objectives:..... 14

2.1-3. Definitions, Acronyms and Abbreviations:..... 15

2.2- Overall Description: 15

2.2-1. Product perspective: 15

2.2-2. Product Functions:..... 16

2.2-3. User Characteristics: 16

2.2-4. Constraints:..... 16

2.3- Specific Requirements:	16
2.3-1. External Interfaces:.....	16
2.3-2. Functions:	16
2.3-3. Performance Requirements:	17
2.3-4. Software Quality Attributes:.....	17
2.4- Functional Requirements:	17
2.4-1. Use cases List:	17
2.4-2. Use case Diagram.....	18
2.4-3. Use case Description.	19
2.5- System Sequence Diagrams:	27
2.5-1. Data Entry.....	28
2.5-2. File Upload.	29
2.5-3. Text Extraction.	30
2.5-4. Data Validation.	31
2.5-5. User Notification.....	32
2.5-6. Corrective Action.....	33
2.5-7. Feedback Mechanism.	34
2.5-8. Document Type Agnosticism.	35
2.5-9. User Interface.....	36
2.6- Domain Model.	37
CHAPTER 3: SOFTWARE DESIGN DESCRIPTION.....	38
3.1- Introduction:	39
3.1-1. Purpose:.....	39
3.1-2. Design Overview:.....	39
3.1-3. Requirement traceability matrix.....	39
3.2- User Interface Design.	40

3.3- Sequence Diagram.....	41
3.4- Class Diagram.....	42
Explanation:.....	43
3.5- Summary.....	43
CHAPTER 4: IMPLEMENTATION AND TESTING	44
4.1- Implementation:.....	45
4.1.1- Frontend:	45
4.1.2- Backend:	45
4.2- Testing:	47
4.2.1- Introduction:.....	47
4.2.2. Test Strategy:	47
4.2.3. Features to be Tested:	47
4.2.4. Test Cases:.....	47
4.3- Summary:.....	52
CHAPTER 5: CONCLUSION	53
5.1- CONCLUSION	54
5.2- FUTURE WORK	54

Chapter 1:
Software Project Management Plan (SPMP)

This Information verification system is a web-based application designed to verify the data entered by user from the documents he/she uploaded while submitting the application for BS admission. This project management plan outlines the objectives, scope, and deliverables of the application development.

1.1- Introduction:

Information Verification system is a web-based application, it could be run on any operating system and its main purpose is to verify whether the user input data matches with the data in the documents he uploaded.

1.1-1. Project Overview:

While the BS admission process of QAU, most of the applicants receive emails regarding their mark's mismatches with the marks in marksheets. The main purpose of this web-based application is to verify the user entered data from the document images he/she uploaded at the time of submission, if the data matches form application, then will be accepted else a notification will be shown to the user regarding the mismatches of data.

1.1-2. Project Deliverables:

The key deliverables of the “Information Verification System” Project includes.

- Web based Application.
- SQLite Database integration.
- Documentation of the Project.
- Testing and Quality Assurance.

1.2- Project Organization:

It contains the process model, tools and techniques used and the role and responsibilities.

1.2-1. Software Process Model:

Throughout the project documentation and implementation, Spiral model of software designing will be used due to following reasons.

1.2.1.1. Flexibility:

In spiral model we can easily make changes at the later stage and can be managed accurately moreover additional functionalities can be added at the later phases.

1.2.1.2- Risk Handling:

One of the most valuable features of spiral model is that of its risk analysis and handling at every phase of the project.

1.2.1.3- Reliable:

As my project can be extended in near future so keeping in mind the requirements spiral model is chosen as it is compatible for large and complex projects

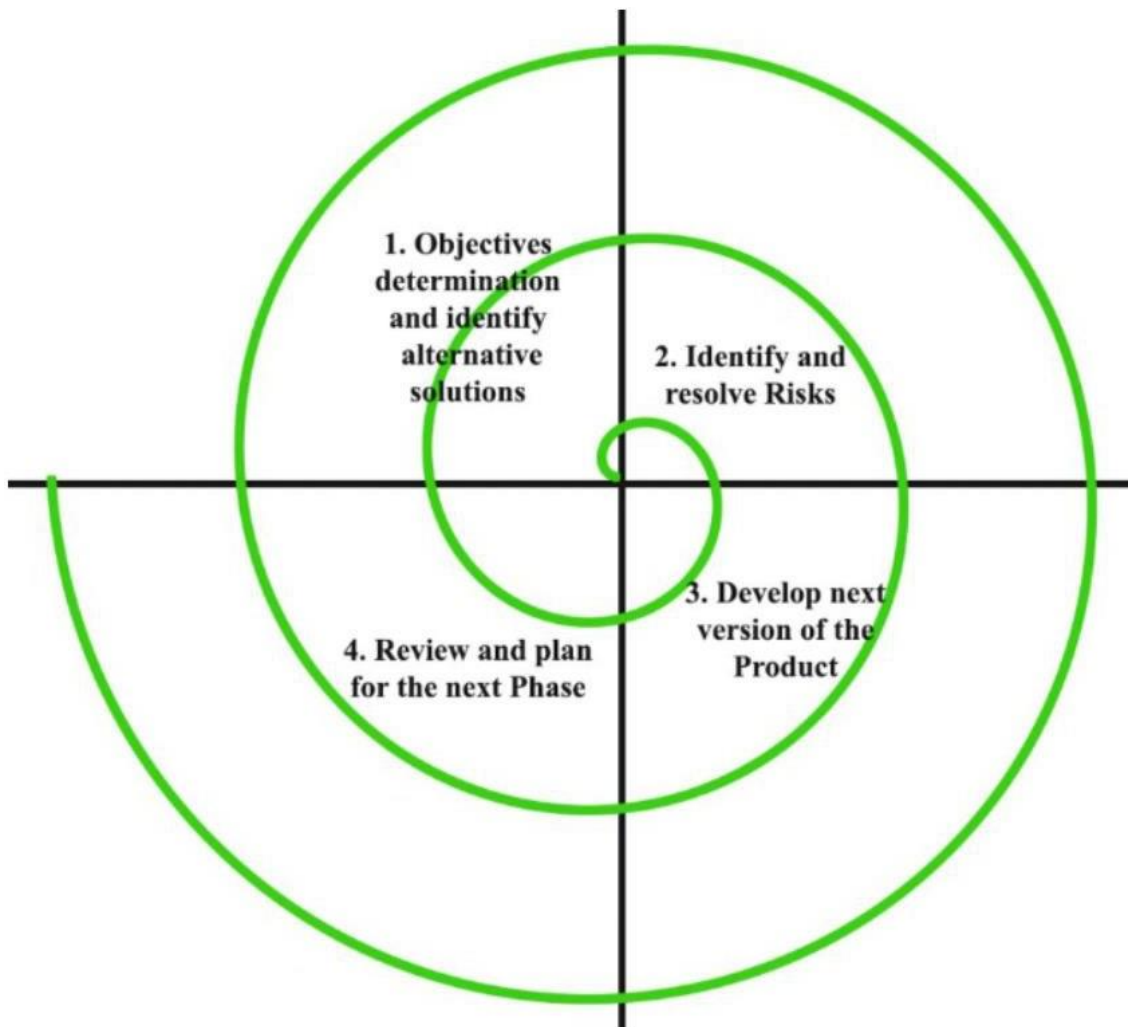


Figure 1.1(Software process model) Spiral Model

1.2-2. Roles and Responsibilities:

Student (Muhammad Taha):

- Development lead.
- Web Developer.
- Data extraction form images specialist.
- Testing and Quality assurance contributor.
- Documentation responsible.

Supervisor (Dr Ayyaz Hussain Abbasi):

- Project Oversight.
- Technical guidance.
- Mentorship.
- Quality assurance support.
- Review and approval.
- Documentation review.

1.2-3. Tools and Techniques:

Tools/ Techniques	Purpose
Microsoft Word	Used for documentation
Draw.io	Used for diagrams designing
Django	For backend development
React.js,	For fronted development
IDE	VS Code
pytesseract	For data extraction from image

1.3- Project Management Plan:

In the management of the project, it describes how the resources and time has been managed throughout the project's lifecycle.

1.3-1. Tasks:

In the projects plan it comprises of two main steps first is the requirement

and analysis and second is the design phase. The major tasks in phase 1 is to identify the scope, requirements, feasibility report, defining the use cases, developing model, develop SRS and review the SRS. While in the second phase of designing the major tasks is to follow the OOP approach to design the model for like input, validate input and evaluate design.

1.3-2. Description:

The description of both major tasks is given below.

1.3-2.1. Requirements and Analysis Phase: Identify requirements:

The main goal is to review case study and define requirements.

Feasibility report:

Goal of this report is to check whether the project is feasible or not.

Define Use cases:

Make use cases Diagrams.

Develop SRS:

It includes all the functional and non-functional requirements and software requirements specification document. All the other details like scope, introduction and purpose etc. are also included in it.

1.3-2.2. Design Phase:

Make user Interfaces:

The front end of the software, the view of the project

Make Sequence Diagram:

Design sequence Diagram to show the sequence of actions taken by the whole system.

Make Class Diagram:

Design class diagram to describe the attributes and operations of entities with the system and constraints imposed on systems.

1.4- Summary:

This chapter introduces the "Information Verification system" and its key deliverables. It highlights the adoption of an iterative and incremental software development process. The chapter details the roles and responsibilities of project, emphasizing the collaboration between a student and a supervisor. It also outlines the essential tools and techniques used throughout the project. This chapter also introduces the division of the project into two phases: Requirements & Analysis Phase and Design Phase, while providing an overview of the tasks involved in each phase.

CHAPTER 2:
**SOFTWARE REQUIREMENTS
SPECIFICATION (SRS)**

2.1-Introduction:

The section mainly describes the SRS of the project including almost everything related to SRS document and scope descriptions, and the purpose of the document is also described in this section.

Purpose:

The purpose of SRS document is to provide a detailed overview of our software product, its parameters, and goals. The main purpose is that the user entered data will be instantly verified from the documents he uploaded.

Scope:

Information verification system play a vital role in the BS admission process of QAU, where most of the students mistakenly entered the incorrect data, and it takes time to the admission office to cross check those data from documents and respond to the student. By using this system, the user will be instantly notified if he/she enters the wrong data. System would accommodate many users to run the app at same time. It doesn't hang or slow down. The system displays appropriate messages to the user whenever the user submits any response.

Objectives:

- ✓ Develop a web-based application using react and Django and create a form to verify and submit data.
- ✓ Making it sure that the application is user friendly and fully functionable so that users don't get any trouble while using it.
- ✓ The data user entered in the application form will be verified instantly by comparing against the data in the documents he/she uploaded alongside.

Major Inputs:

- ✓ User entered data and documents.

Major Outputs:

- ✓ Data validation message to user

Functionality:

- ✓ Enter User details.
- ✓ Upload document image.
- ✓ Extract data from image.
- ✓ Compare against the user entered data.
- ✓ Show notification if not matches else submit to database.

Constraints:

- ✓ Web based application.
- ✓ User must have internet connection.
- ✓ User must be familiar with basic English language.
- ✓ User must fill and submit the form.

2.1-3. Definitions, Acronyms and Abbreviations:

Terms	Definitions
UC	Use case
SSD	System sequence diagram
IVS	Information verification system
SRS	Software requirement systems

2.2- Overall Description:

2.2-1. Product perspective:

DVS is a web-based application which will verify the user entered data from the documents he/she uploaded.

2.2-2. Product Functions:

- ✓ Enter User details.
- ✓ Upload document image.
- ✓ Extract data from image.
- ✓ Compare against the user entered data.
- ✓ Show notification if not matches else submit to database.
- ✓ Store data to database.

2.2-3. User Characteristics:

System expects from user to know basic English language and how to use web browser in a window- based machines.

2.2-4. Constraints:

- ✓ Web based application.
- ✓ User must have internet connection.
- ✓ User must be familiar with basic English language.
- ✓ User must fill and submit the form.

2.3- Specific Requirements:

2.3-1. External Interfaces:

The external interface for this application is the SQLite Database of web application of this system to store and fetch the submitted data.

2.3-2. Functions:

- ✓ Get User details.
- ✓ Get document image.
- ✓ Extract data from image.
- ✓ Compare against the user entered data.
- ✓ Show notification if not matches else submit to database.
- ✓ Store data to database.

2.3-3. Performance Requirements:

Multiple users can use this application at a time.

2.3-4. Software Quality Attributes:

2.3-4.1. Usability

The user interface is intuitive, with a user-friendly design that promotes ease of navigation. Clear and concise labels, notifications, and feedback help users understand the application's functionality.

2.3-4.2. Reliability and Availability

The system will be more than 90% reliable and operational.

2.3-4.3. Performance:

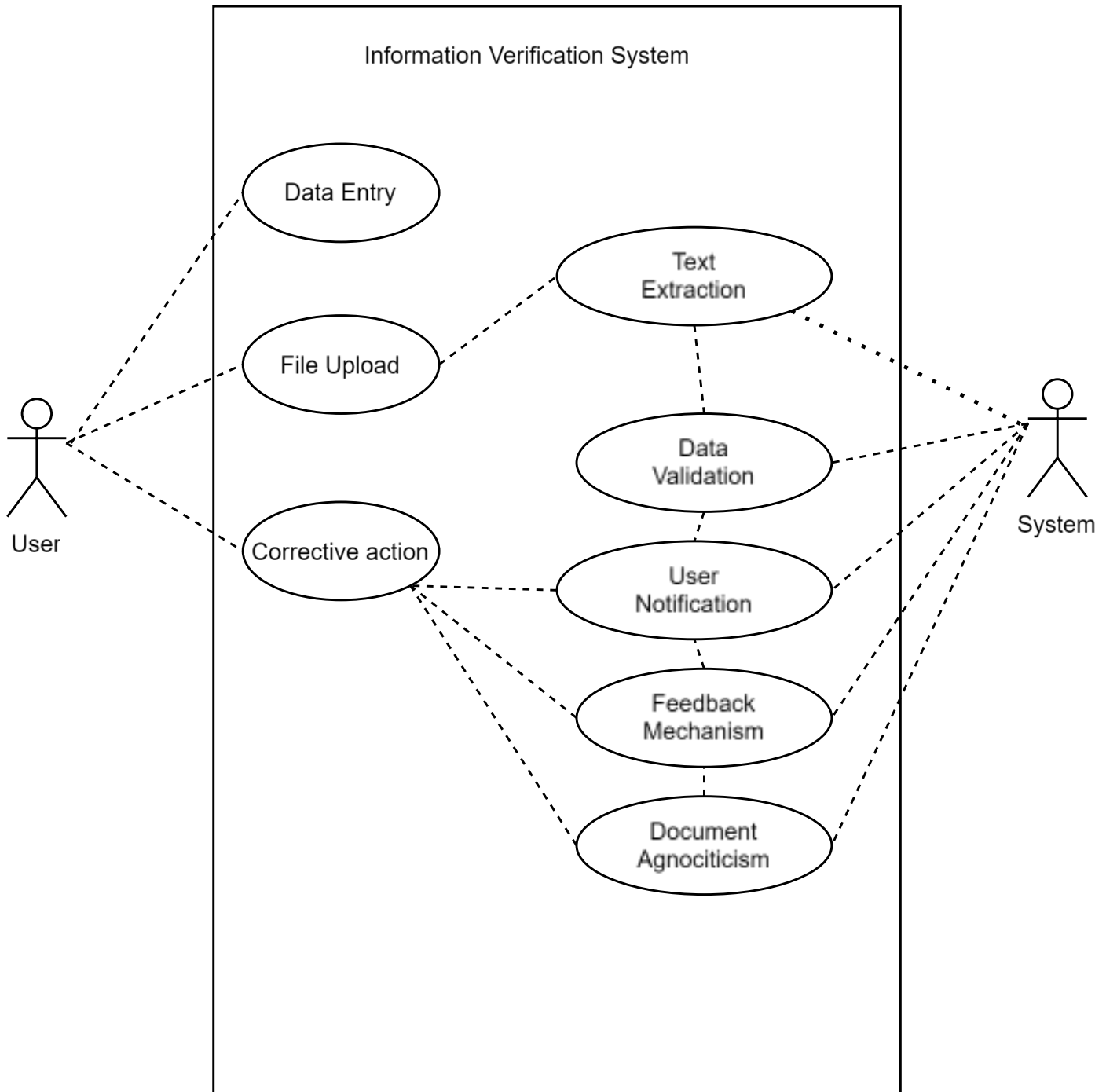
System would accommodate many users to run the app at same time. It doesn't hang or slow down. The system displays appropriate messages to the user whenever the user submits any response.

2.4- Functional Requirements:

2.4-1. Use cases List:

- ✓ User Data entry
- ✓ Upload Document
- ✓ Data Extraction
- ✓ Data Validation
- ✓ Data comparison
- ✓ User Notification
- ✓ Corrective action
- ✓ Feedback mechanisms
- ✓ Document Type Agnosticism
- ✓ User friendly Interface.

2.4-2. Use case Diagram.



2.4-3. Use case Description.

Complete description of use cases is given below.

2.4-3.1. User Data Entry

ID	UC-1
Name	User Data Entry
Primary Actors	User
Stack holders	System
Description	Users must enter specific data through a web form
Pre-condition	User must be authenticated and Registered.
Post-condition	Data is entered
Main success Scenario	User entered the data
Alternate Scenario	Data entry fails due to system issues.
Special Requirements	None
Frequency	Frequent

2.4-3.2. Upload Document.

ID	UC-02
Name	Upload Document
Primary Actors	User
Stack holders	User
Description	Users should be able to upload various document types for analysis
Pre-condition	User must be authenticated and Registered.
Post-condition	Documentis uploaded to the system
Main success Scenario	1. User uploads a document. 2. Document is securely stored.
Alternate Scenario	Document upload fails due to system issue
Frequency	Frequent

2.4-3.3. Text Extraction.

ID	UC-03
Name	Text Extraction
Primary Actors	System
Stack holders	User
Description	Extracting text from uploaded documents for further analysis.
Pre-condition	Document already uploaded to the system
Post-condition	Text is extracted from the document
Main success Scenario	1. Document is uploaded. 2. System extracts text from the document.
Alternate Scenario	Text extraction fails due to document format issues.
Special Requirements	N/A.
Frequency	Frequent

2.4-3.4. Data Validation.

ID	UC-04
Name	Data Validation
Primary Actors	System
Stack holders	User
Description	Validating user-entered data against extracted document text.
Pre-condition	<ol style="list-style-type: none">1. Document must be already uploaded by the user.2. Data must be entered by the user
Post-condition	Validation results are communicated to the user.
Main success Scenario	<ol style="list-style-type: none">1. User enters data.2. System compares entered data with extracted text.3. Validation results are communicated.
Alternate Scenario	Validation fails due to mismatched data.
Special Requirements	N/A
Frequency	Whenever any document is to be verified.

2.4-3.5. User Notification.

ID	UC-05
Name	User Notification
Primary Actors	System
Stack holders	User
Description	Notifying users in case of data discrepancies.
Pre-condition	Data validation results must be available
Post-condition	Notification is sent
Main success Scenario	<ol style="list-style-type: none">1. Data validation results are available.2. System generates notifications for users with mismatched data.3. Notifications prompt users to review and correct data.
Alternate Scenario	Notification fails to be sent due to system issues.
Special Requirements	N/A
Frequency	occasional

2.4-3.6. Corrective Action.

ID	UC-06
Name	Corrective Action
Primary Actors	User
Stack holders	System
Description	Providing users with the ability to correct entered data.
Pre-condition	User received a notification for data discrepancies.
Post-condition	Data is corrected
Main success Scenario	<ol style="list-style-type: none">1. User receives a notification.2. User navigates to the correction interface.3. User edits and resubmits data.4. Changes are reflected in subsequent validations
Alternate Scenario	User encounters issues while correcting data.
Special Requirements	N/A
Frequency	Frequent

2.4-3.7. Feedback Mechanism.

ID	UC-07
Name	Feedback Mechanism
Primary Actors	System
Stack holders	User
Description	Providing feedback on the correctness of user-entered data.
Pre-condition	Data validation results must be available
Post-condition	Feedback is generated
Main success Scenario	<ol style="list-style-type: none">1. Data validation results are available.2. System generates feedback messages indicating the accuracy of user-submitted information.3. Guidance is provided to assist users in making accurate submissions
Alternate Scenario	Feedback generation fails due to system issues.
Special Requirements	N/A
Frequency	Frequent

2.4-3.8. Document type Agnosticism.

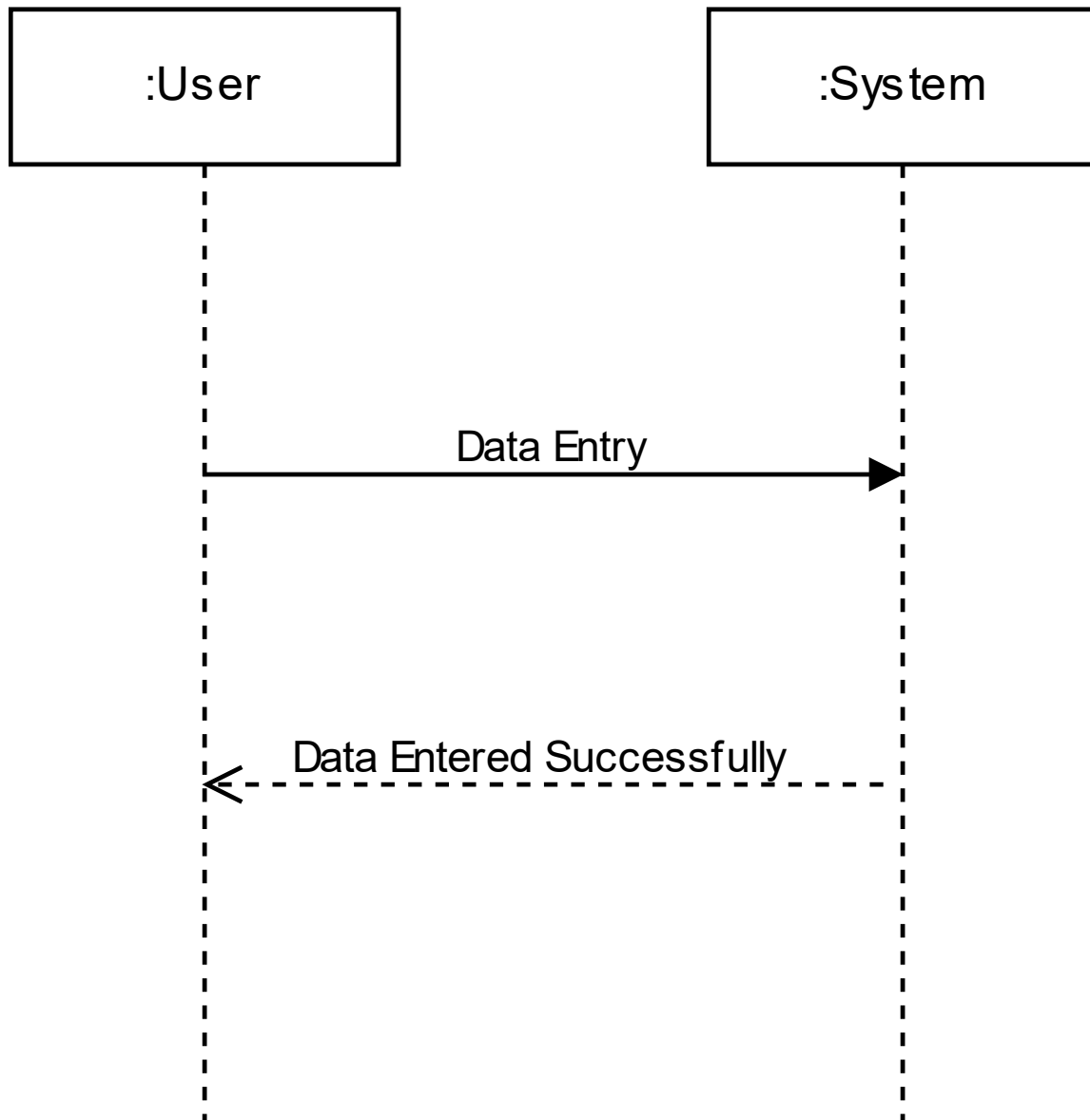
ID	UC-08
Name	Document Type Agnosticism
Primary Actors	System
Stack holders	User
Description	Supporting various document types for analysis.
Pre-condition	System can handle multiple document formats.
Post-condition	Data is corrected
Main success Scenario	1. Users can upload documents of various types. 2. Document analysis modules are adaptable to different document formats.
Alternate Scenario	System encounters issues when handling specific document formats.
Special Requirements	N/A
Frequency	Frequent

2.4-3.9. User Friendly Interface.

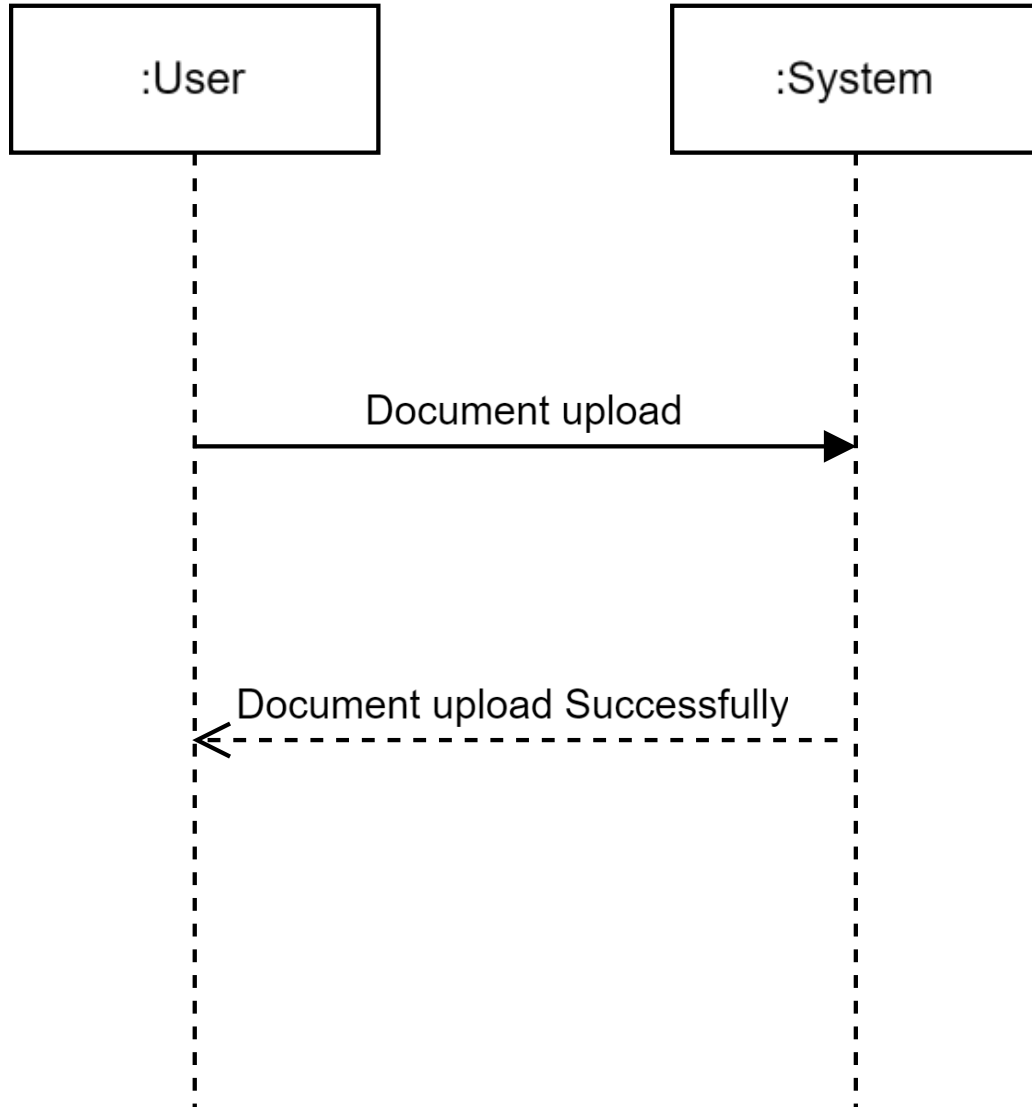
ID	UC-9
Name	User Friendly Interface
Primary Actors	User
Stack holders	System
Description	Designing an intuitive and user-friendly web interface.
Pre-condition	System must be accessible to the user.
Post-condition	Interface is designed
Main success Scenario	1. Users can access the system. 2. Interface is easy to navigate and visually appealing. 3. Data entry forms are well-organized and user-friendly.
Alternate Scenario	Users face difficulties in navigating the interface.
Special Requirements	N/A
Frequency	Frequent

2.5- System Sequence Diagrams:

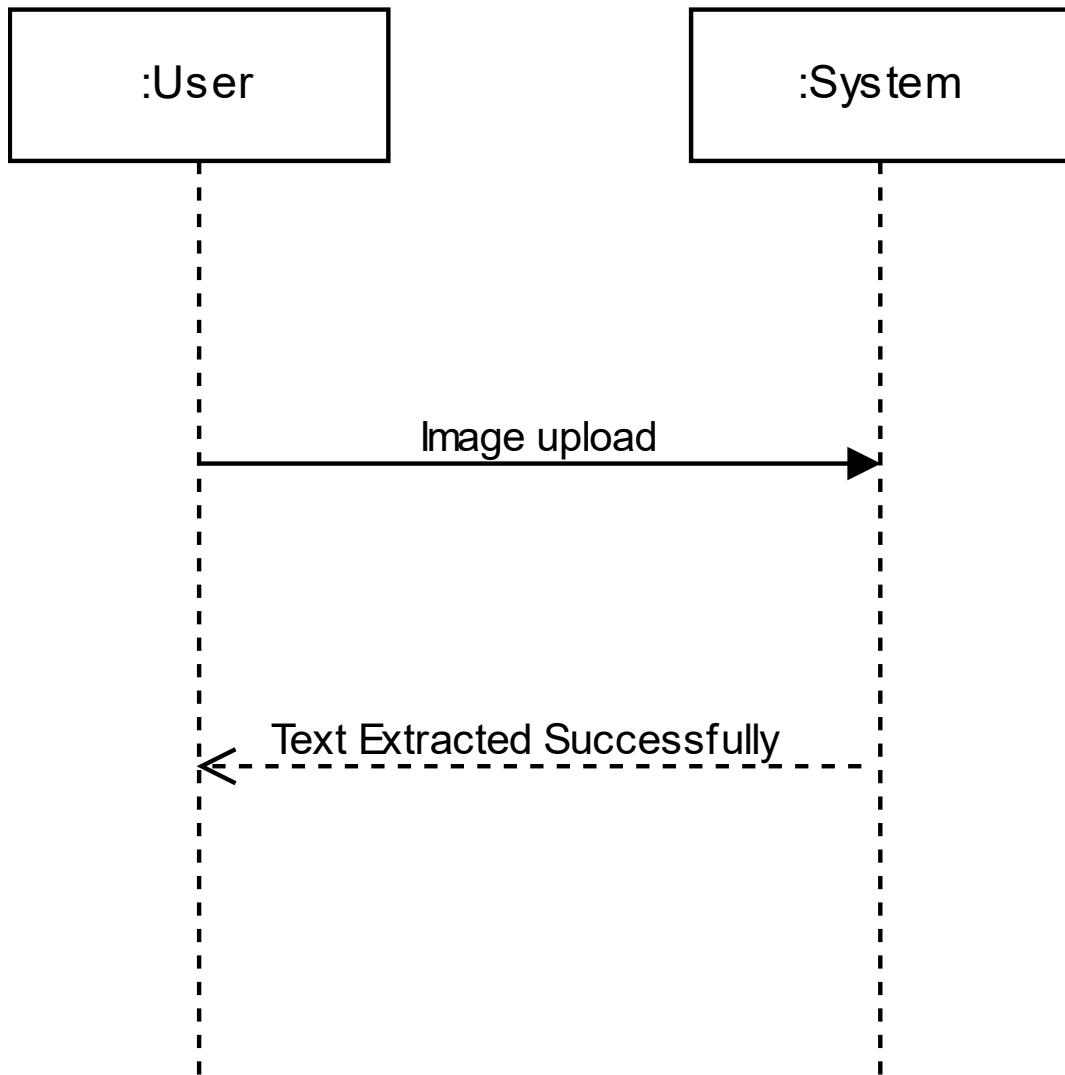
2.5-1. Data Entry.



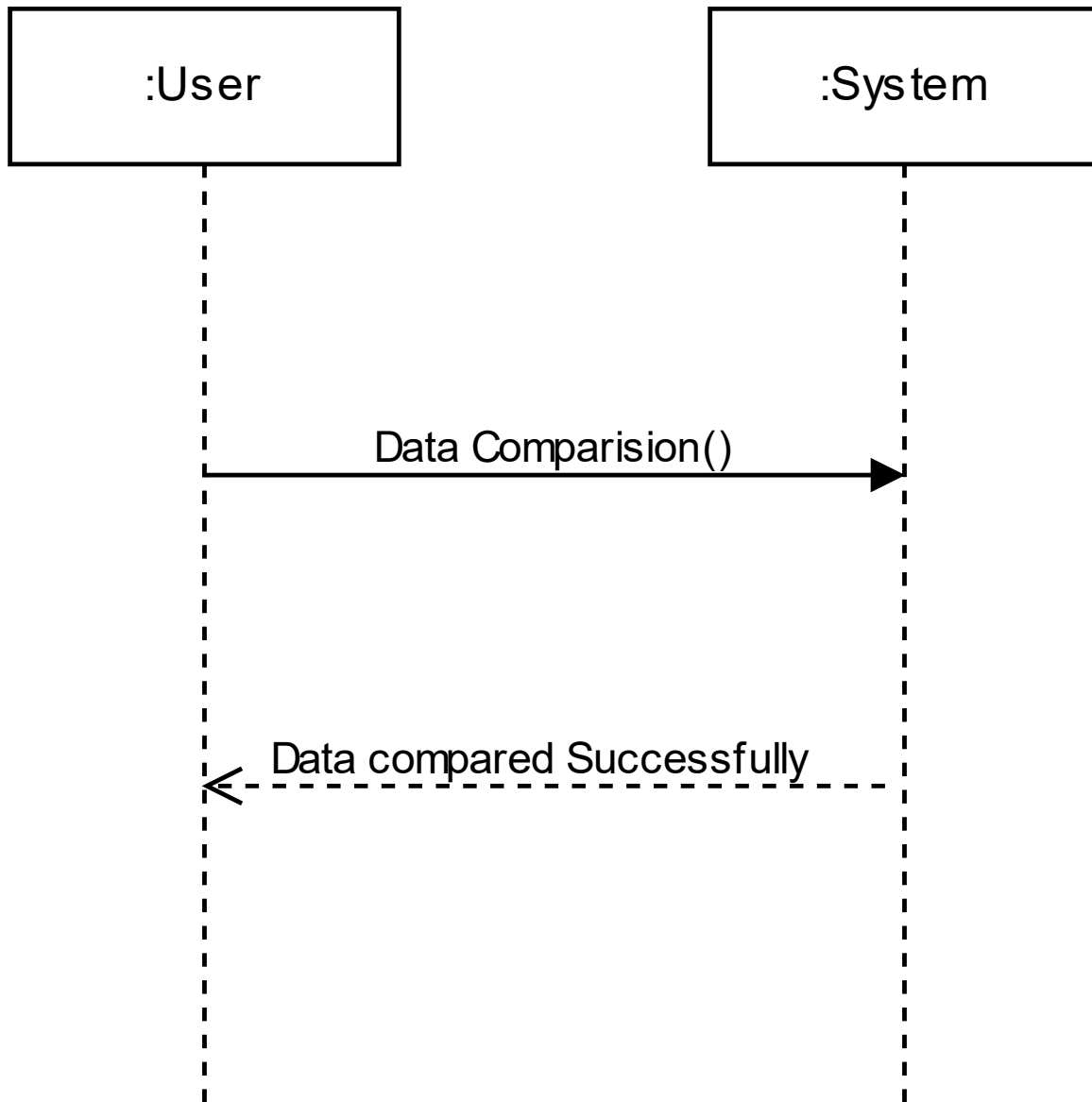
2.5-2. File Upload.



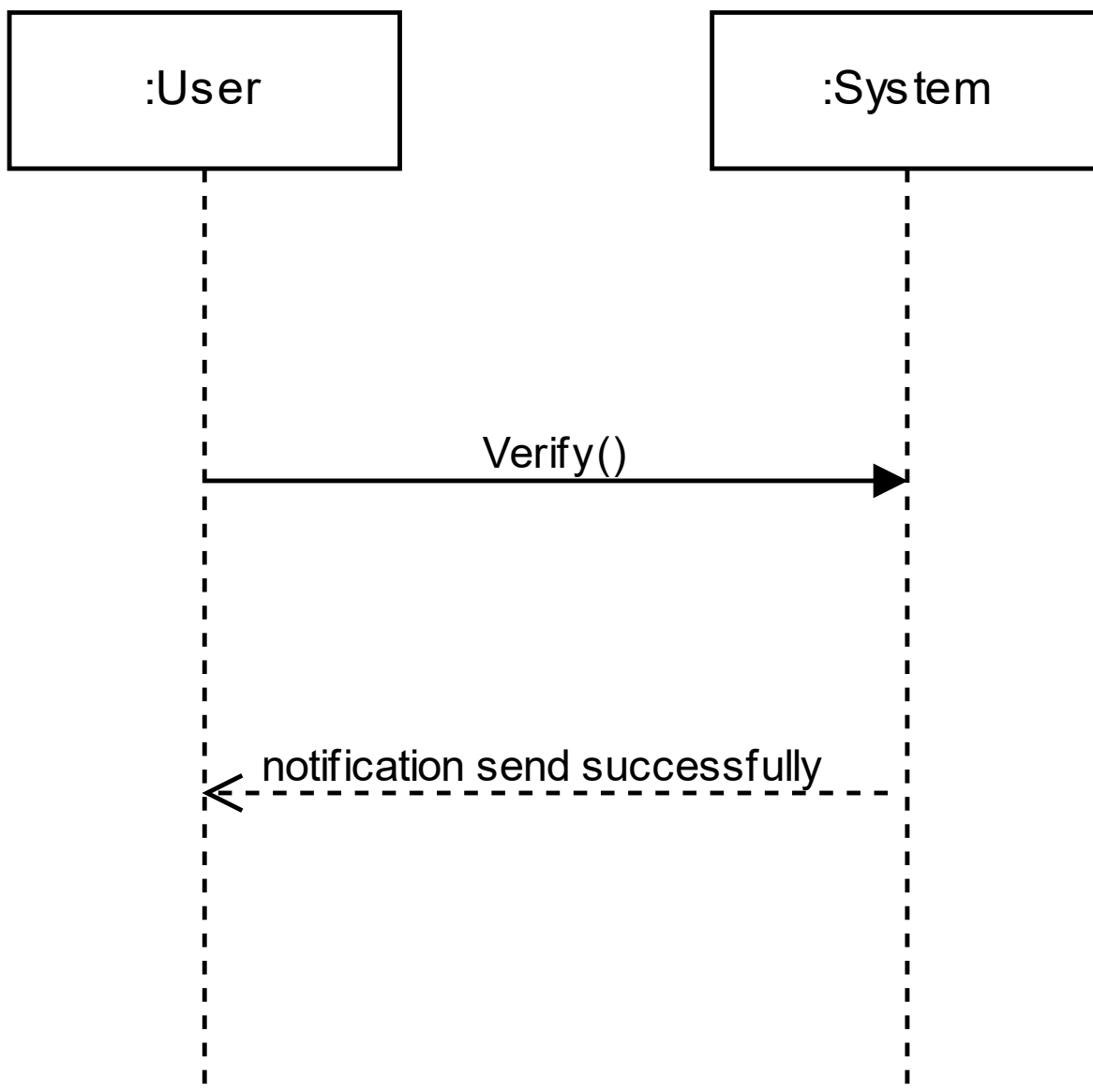
2.5-3. Text Extraction.



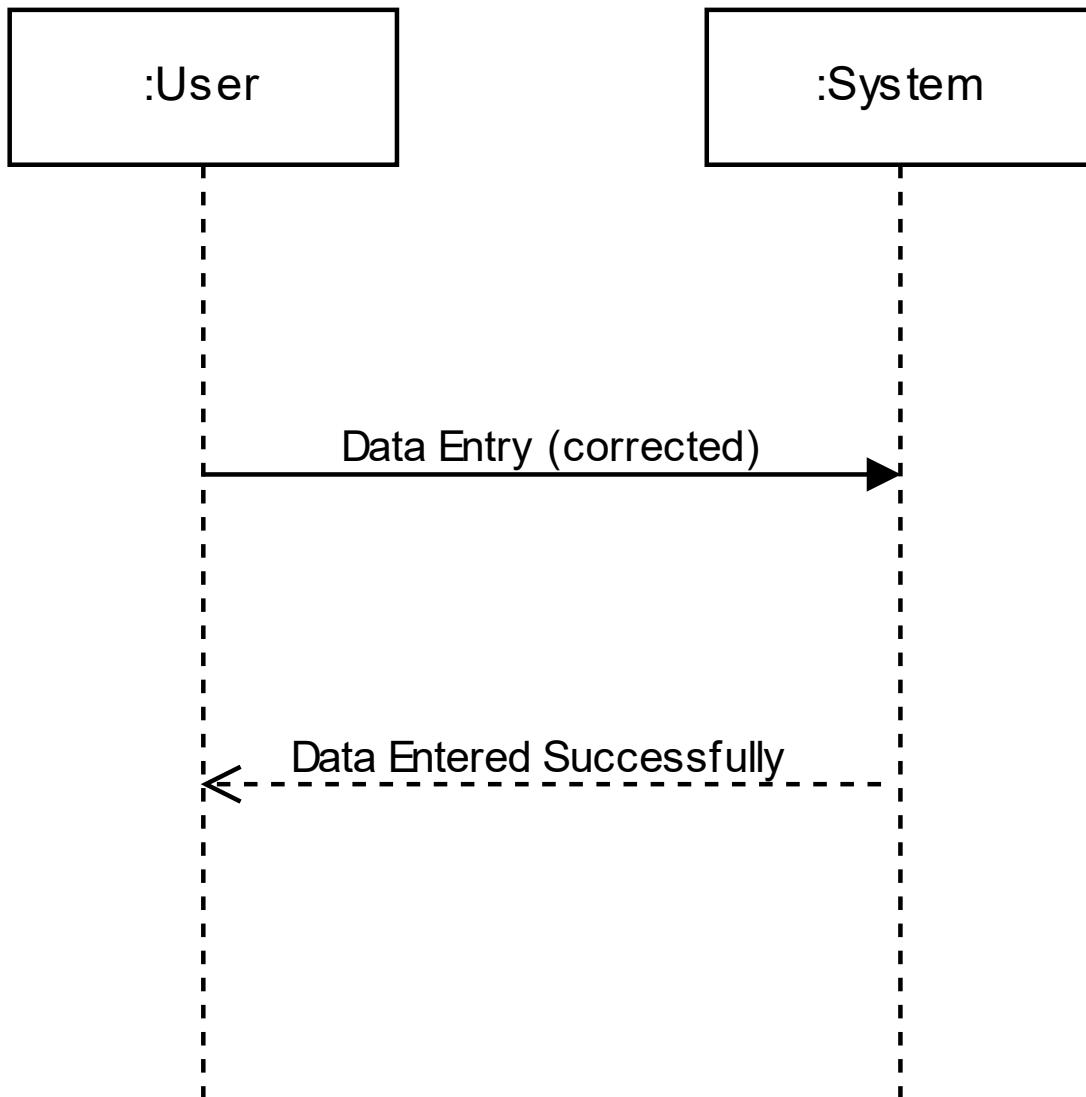
2.5-4. Data Validation.



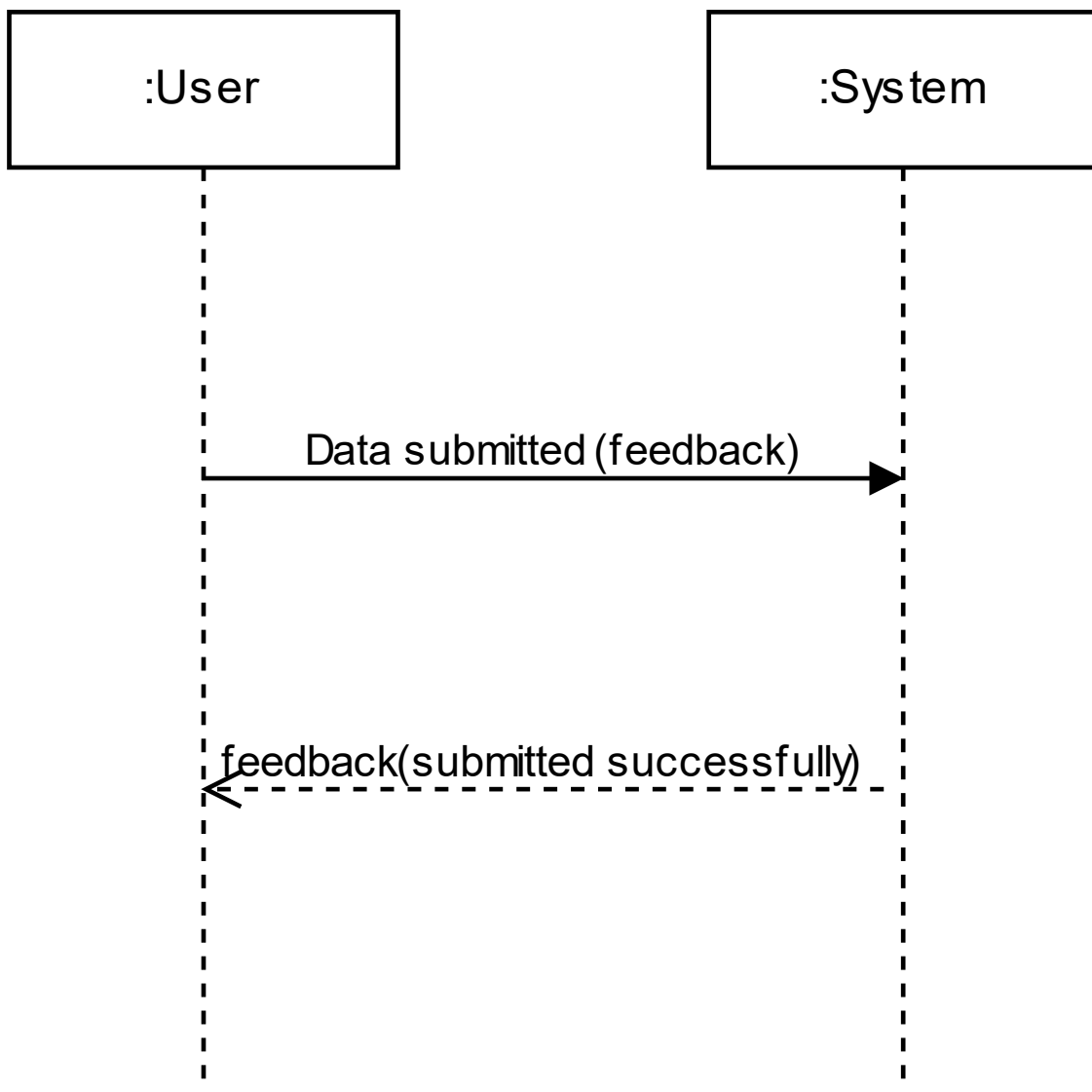
2.5-5. User Notification.



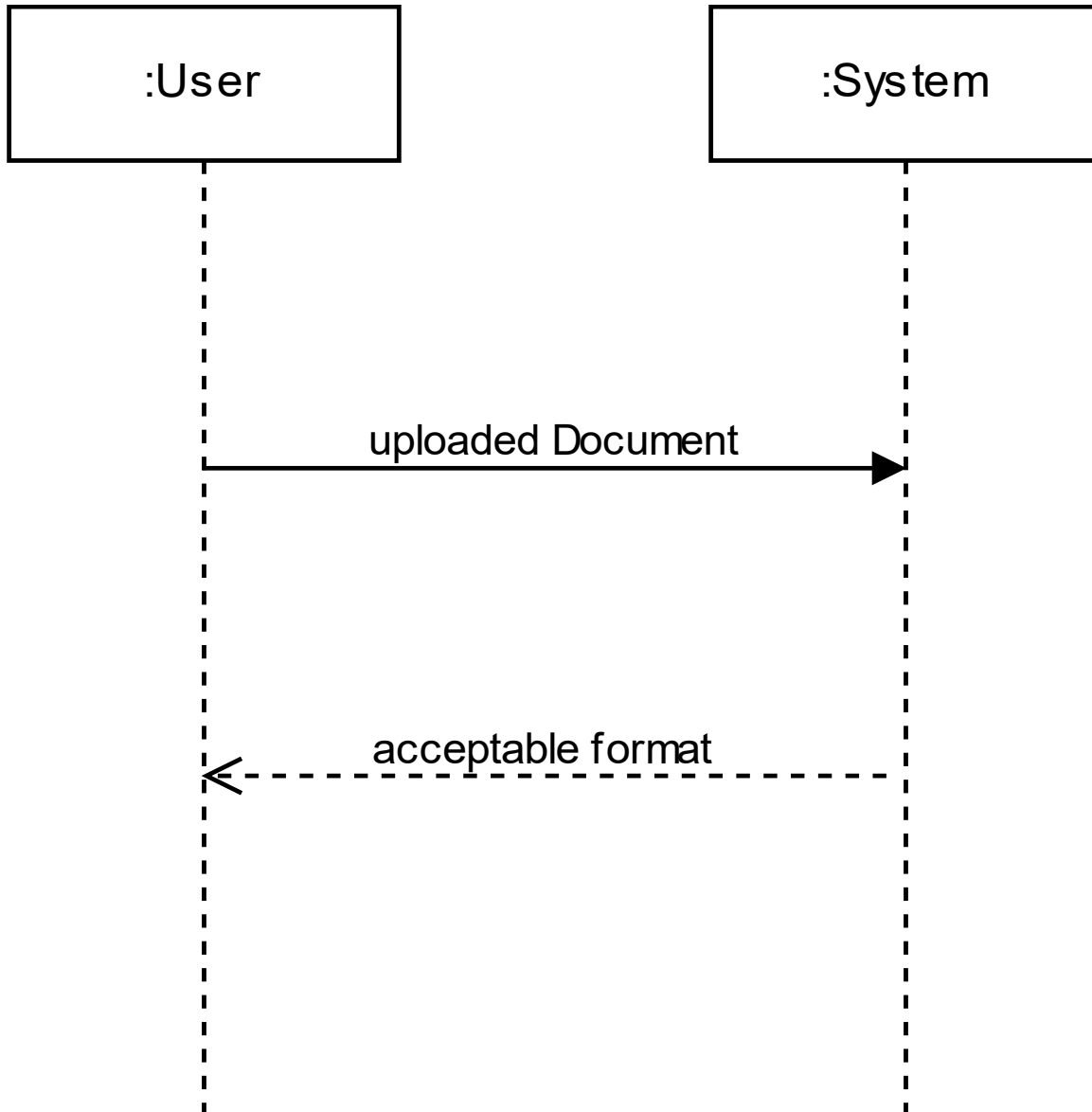
2.5-6. Corrective Action.



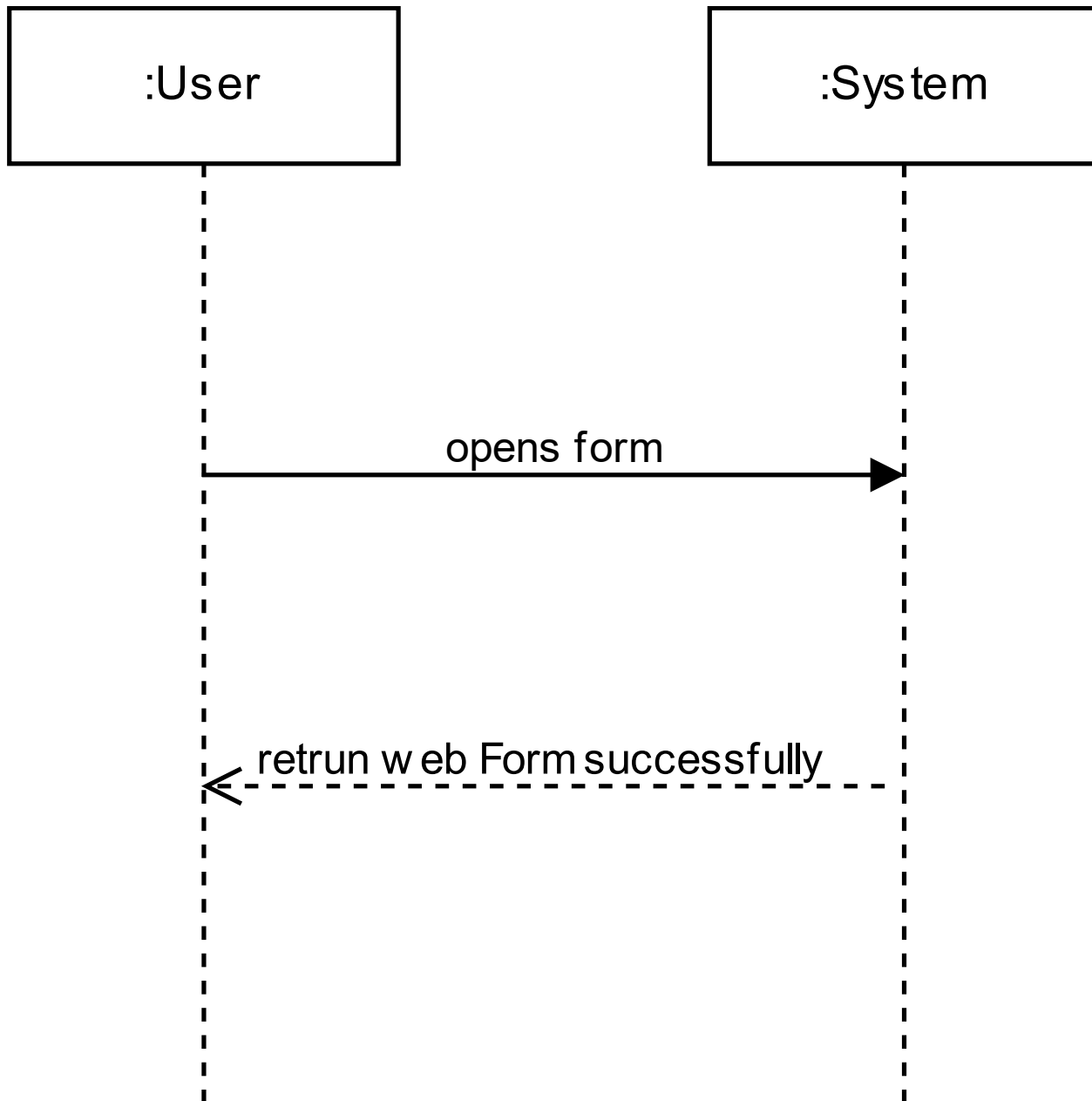
2.5-7. Feedback Mechanism.



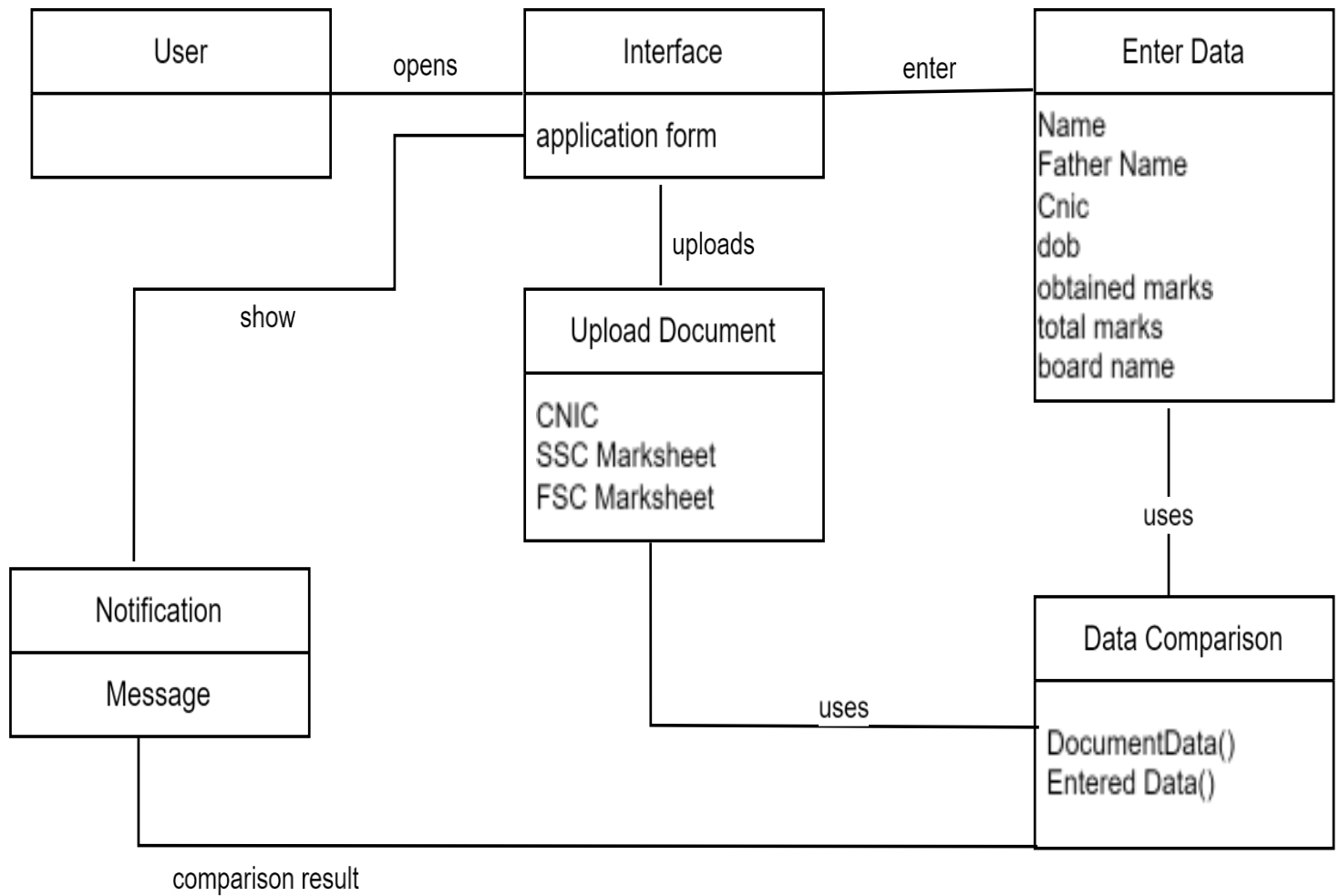
2.5-8. Document Type Agnosticism.



2.5-9. User Interface.



2.6- Domain Model.



CHAPTER 3:
**SOFTWARE DESIGN DESCRIPTION
(SDD)**

3.1- Introduction:

Software Design Description (SDD) is the representation of a software design which is used for communicating design information of a system to its stakeholders. It shows how the software system will be structured to satisfy the requirements. The SDD document contains the interface design and class diagram. It includes a description of how the software will meet the requirements.

3.1-1. Purpose:

The purpose of the Software Design Document is to provide a description of the design of Document verification application to allow for software development to proceed with an understanding of what is to be built and how it is expected to build. The Software Design Document provides information necessary to provide a description of the details for the software and system to be built.

3.1-2. Design Overview:

The software design document provides design details of Document verification application. The document contains a complete low-level description of the system, providing insight into the structure and design of each component.

3.1-3. Requirement traceability matrix.

It is a document that maps and traces user requirements with test cases. The main purpose of Requirement Traceability Matrix is to see that all test cases are covered so that no functionality should be missing while testing. The Matrix is created at the very beginning of a project as it forms the basis of the project's scope and the deliverables that will be produced. The Matrix is bidirectional, as it tracks the requirement forward by examining the output of the deliverables and backward by looking at the business requirement that was specified for a particular feature of the product or app.

Requirement ID	Requirement Name	Interface	Test Case
UC-01	User Data Entry	Yes	No
UC-02	Upload Document	Yes	Yes
UC-03	Text Extraction	No	Yes
UC-04	Data validation	No	Yes

UC-05	User Notification	Yes	Yes
UC-06	Corrective Action	Yes	Yes
UC-07	Feedback Mechanism	Yes	Yes
UC-08	Document Type Agnosticism	Yes	Yes
UC-09	Interface	Yes	Yes

3.2- User Interface Design.

The user will be directed to a web application form where he can enter the data and upload the images if there is no mismatching between document data and the user entered data, the form will be submitted and if there is any mismatching, user will be notified.

lvS

[Home](#)[About](#)[Account](#)

Name

Nasir hussain

Phone

03431293458

Obtained Marks (SSC)

848

Obtained Marks (FSC)

425

Marksheet Upload (SSC)

Choose File

sss_nasir_.jpg

Father's Name

Ghulam Ali

Date of Birth

02/19/2024

Total Marks (SSC)

1100

Total Marks (FSC)

550

Marksheet Upload (FSC)

Choose File

nasir class 11th.jpg

CNIC

71302-6240559-2

Gender:

M

Percentage (SSC)

90

Percentage (FSC)

89

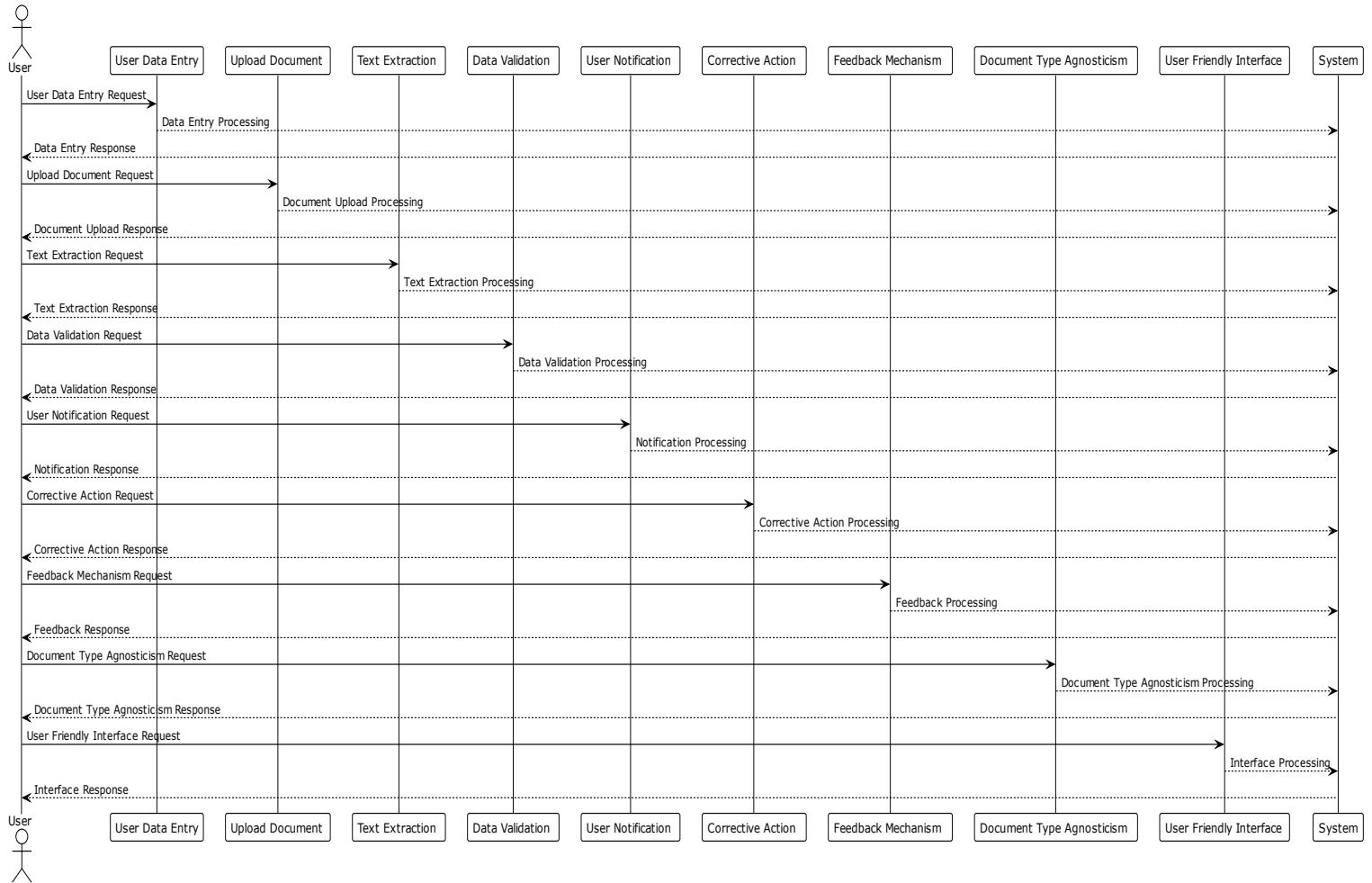
Verify

© 2024 Information Verification System All rights reserved by Muhammad Taha

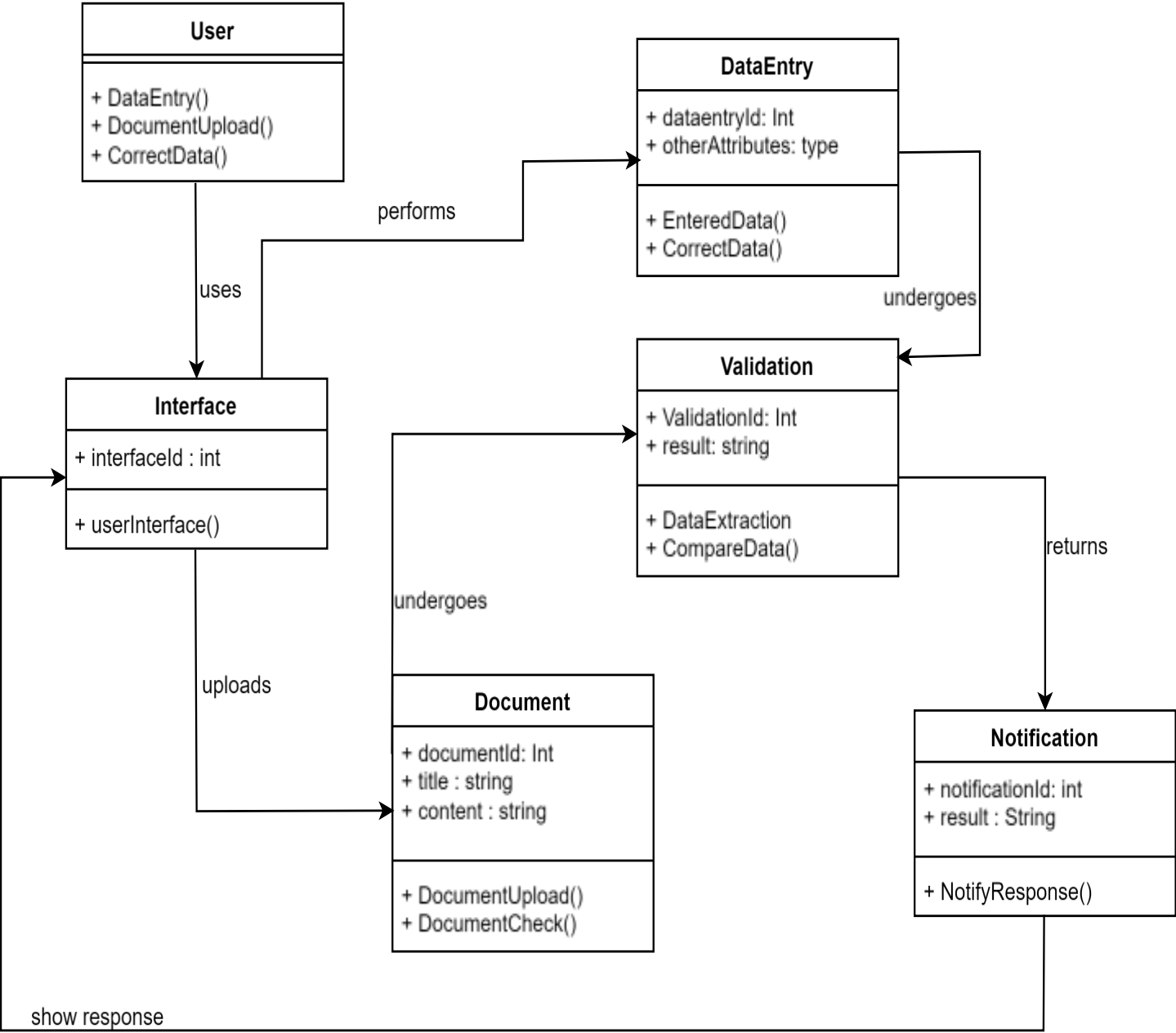
Activate Windows

Go to Settings to activate Windows.

3.3- Sequence Diagram.



3.4- Class Diagram.



Explanation:

The class diagram depicts the structure and relationships of key entities within an application. It includes classes such as "User," "Document," "Data Entry," "Validation," "Notification," and "Interface." Users upload documents, triggering validation processes for both data and documents. The system generates notifications based on validation results. The "Interface" class represents the user interface facilitating data entry and document uploads. Each class encapsulates relevant attributes, such as user details, document content, and validation outcomes. The relationships, represented by arrows and labels, illustrate how instances of these classes interact. The diagram aids in understanding the data flow and interactions within the application, providing a visual representation of its key components and their connections.

3.5- Summary.

This chapter outlines the Software Design Description (SDD) for the Information Verification System project. The SDD serves as a detailed plan for software development, offering insights into its architecture and features. It encompasses a Requirement Traceability Matrix (RTM) that links user requirements to test cases, ensuring thorough testing coverage. The document elaborates on the user interface design, featuring graphical representations for better understanding. Furthermore, it includes sequence and class diagrams to illustrate system interactions and structure. The class diagram highlights key classes and their relationships within the system, demonstrating how the Information Verification System aims to enhance data verification processes while maintaining security and accuracy.

CHAPTER 4: **IMPLEMENTATION AND TESTING**

Implementation and Testing

In the process of developing the "Information Verification System" web-based application, testing and implementation have been integral phases. This chapter provides insights into the quality assurance measures undertaken to ensure the application's functionality, security, and user experience. It describes the strategies and methodologies employed to validate the system's performance and the steps taken to fulfill this project.

4.1- Implementation:

The "Information verification system" is a web-based application developed by using Vs code and programmed in React for frontend and Django for backend. It uses SQLite as a database. The application allows user to verify his/her data from the documents uploaded at the time of form submission. If it found any data mismatching, user will be notified and if no error found data will be submitted to database. The application allows admin panel to track all the submitted data by the users.

4.1.1- Frontend:

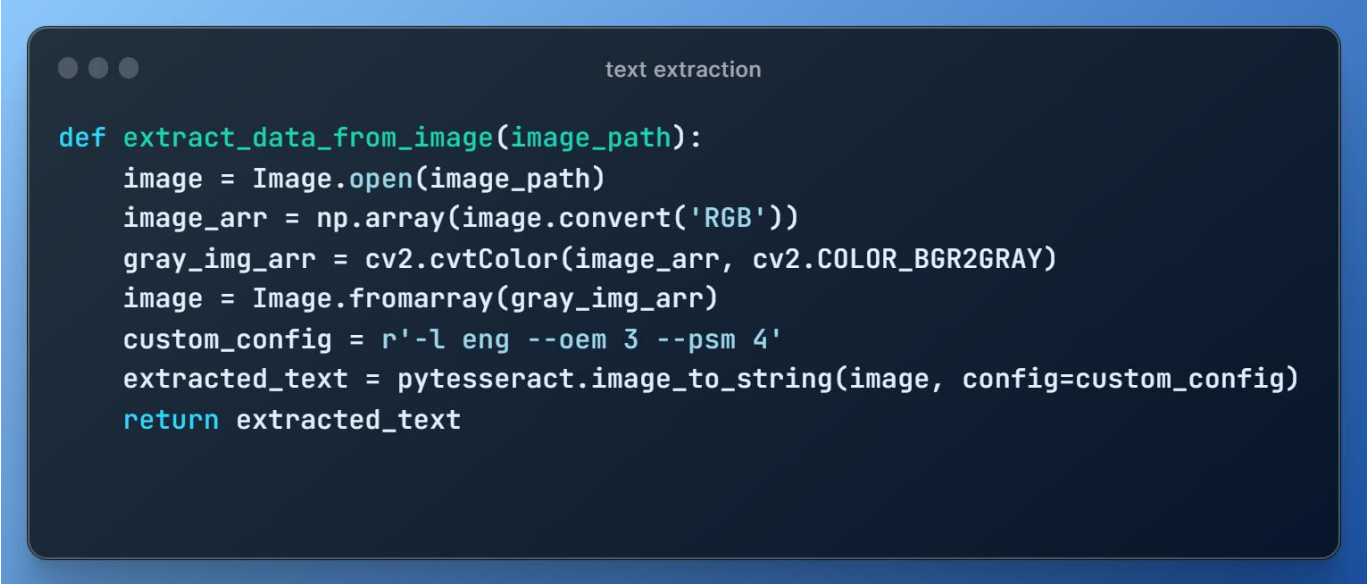
The frontend part has been programmed in React.js.

4.1.2- Backend:

Back-end implementation refers to the development of the server-side components and functionality that handle data processing and logic in a software system. For the Backend the prominent Python framework has been used along side it's default database SQLite to store data.

4.1.2.1- Text Extraction:

For text extraction from the image uploaded.

A screenshot of a code editor window with a dark blue background and light blue text. The window title is "text extraction". The code is a Python function named "extract_data_from_image" that takes "image_path" as an argument. It opens the image, converts it to RGB, then to grayscale using OpenCV, and finally uses pytesseract to extract text from the grayscale image. The extracted text is returned.

```
def extract_data_from_image(image_path):  
    image = Image.open(image_path)  
    image_arr = np.array(image.convert('RGB'))  
    gray_img_arr = cv2.cvtColor(image_arr, cv2.COLOR_BGR2GRAY)  
    image = Image.fromarray(gray_img_arr)  
    custom_config = r'-l eng --oem 3 --psm 4'  
    extracted_text = pytesseract.image_to_string(image, config=custom_config)  
    return extracted_text
```

Dependencies:

Pillow (PIL): Python Imaging Library for handling images.

NumPy: Library for numerical operations in Python.

OpenCV (cv2): Open-Source Computer Vision Library for image processing.

tesseract: Python wrapper for Google's Tesseract-OCR Engine

Steps:

1. Opens the image file specified by 'image path'.
2. Converts the image to a NumPy array with RGB color format.
3. Converts the RGB image array to a grayscale image array.
4. Converts the grayscale image array back to a Pillow Image.
5. Sets up Tesseract OCR configuration using custom settings.
 - Language: English ('-l eng')
 - OCR Engine Mode: Default ('--oem 3')
 - Page Segmentation Mode: Assume a single block of text ('--psm 4').
6. Applies Tesseract OCR to the grayscale image and extracts text.
7. Returns the extracted text.

4.1.2.2- Data Validation:

The extracted text undergoes tokenization using the 'nltk' library. Subsequently, each token is systematically compared with the corresponding data from the form. Following this, a JSON file is generated, which is then sent to the frontend. The JSON file serves the purpose of facilitating data correction in the form, particularly in cases where discrepancies or mismatches are identified.

4.1.2.3- Database integration:

Upon successful validation without errors in the JSON file, a submit button becomes visible to the user. When the user presses the submit button, the corrected and verified data is stored in the database. Subsequently, the data is retrieved and displayed on a dedicated admin page.

4.2- Testing:

Testing involves systematically evaluating the software to ensure it functions correctly and meets its intended requirements.

4.2.1- Introduction:

Software testing plays a vital role in the software development process. It involves a sequence of steps that a tester follows to confirm whether a software application is operating correctly and is devoid of errors, ensuring it aligns with the end-user requirements. This procedure holds significant importance throughout the software development lifecycle, as it allows for the evaluation of product quality and aids in the detection and resolution of potential issues that may arise.

4.2.2. Test Strategy:

The test strategy serves as a guide to test the system and manage potential risks throughout the software development process. Our testing approach consists of several key phases, beginning with unit tests for each project component, followed by integration testing, and concluding with acceptance testing.

4.2.3. Features to be Tested:

Following features of the system will be tested:

- Upload Document
- Text Extraction
- Data validation
- User Notification
- Corrective Action
- Feedback Mechanism
- Document Type Agnosticism
- Interface

4.2.4. Test Cases:

In software engineering a test case serves as a defined set of conditions. These conditions are employed by a tester to determine the functionality of an application, software system, or one of its specific features. The entire testing process hinges upon the execution and evaluation of these test cases

TC-01: Upload Document

Test ID	TC-01
Description	Check if the user can upload various document types for analysis.
Setup	User is authenticated and registered
Input	User Uploads a valid document
Instructions	1. Navigate to the document upload section. 2. Upload a valid document.
Expected Result	Document is securely stored.
Actual Result	Document is securely stored.
Verdict	pass

TC-02: Text Extraction

Test ID	TC-02
Description	Check if the system can extract text from uploaded documents for further analysis.
Setup	Document is already uploaded to the system.
Input	Valid document for text extraction.
Instructions	1. Ensure the document is uploaded. 2. Initiate text extraction process.
Expected Result	System successfully extracts text from the document.

Actual Result	System successfully extracts text from the document.
Verdict	pass

TC-03: Data Validation

Test ID	TC-03
Description	Check if the system can validate user-entered data against extracted document text.
Setup	1. Document is already uploaded by the user. 2. User has entered data.
Input	Valid user-entered data.
Instructions	1. Ensure the document and user data are available. 2. Initiate data validation.
Expected Result	System successfully validates user-entered data.
Actual Result	System successfully validates user-entered data.
Verdict	pass

TC-04: User Notification

Test ID	TC-04
Description	Check if the system notifies users in case of data discrepancies.
Setup	Data validation results are available.
Input	Valid data validation results.

Instructions	1. Ensure data validation results are available. 2. Initiate user notification process.
Expected Result	System successfully generates notifications for users with mismatched data.
Actual Result	System successfully generates notifications for users with mismatched data.
Verdict	pass

TC-05: Corrective Action

Test ID	TC-05
Description	Check if the system provides users with the ability to correct entered data.
Setup	User received a notification for data discrepancies.
Input	Valid user corrections.
Instructions	1. Ensure the user has received a notification. 2. Navigate to the correction interface. 3. Edit and resubmit data.
Expected Result	User successfully corrects entered data, and changes are reflected in subsequent validations.
Actual Result	User successfully corrects entered data, and changes are reflected in subsequent validations.
Verdict	pass

TC-06: Feedback Mechanism

Test ID	TC-06
----------------	--------------

Description	Check if the system provides feedback on the correctness of user-entered data.
Setup	Data validation results must be available.
Input	Valid data validation results.
Instructions	1. Ensure data validation results are available. 2. Initiate feedback mechanism.
Expected Result	System successfully generates feedback messages indicating the accuracy of user-submitted information. Guidance is provided to assist users in making accurate submissions.
Actual Result	System successfully generates feedback messages indicating the accuracy of user-submitted information. Guidance is provided to assist users in making accurate submissions.
Verdict	pass

TC-07: User Friendly Interface

Test ID	TC-07
Description	Check if the system displays an intuitive and user-friendly interface.
Setup	System must be accessible to the user.
Input	User accesses the system.
Instructions	1. Navigate to the system's interface. 2. Observe the design, navigation, and overall user-friendliness.
Expected Result	Users can access the system, and the interface is easy to navigate and visually appealing. Data entry

	forms are well-organized and user-friendly.
Actual Result	Users can access the system, and the interface is easy to navigate and visually appealing. Data entry forms are well-organized and user-friendly.
Verdict	Pass

4.3- Summary:

This chapter covers the implementation and testing phases of the project. The implementation section discussed the key functions of the system and provide details about how they have been implemented. It is important to note that testing is a crucial component of the software development life cycle, and in the testing section, there is a list of all the features and functions that are to be tested and describes the approach taken for testing.

CHAPTER 5: **CONCLUSION**

5.1- CONCLUSION

In conclusion, the Information Verification System presents a robust solution for ensuring data accuracy and reliability. The system's comprehensive functionalities cover user registration, data entry, document uploading, text extraction, data validation, user notification, corrective action, feedback mechanism, and an intuitive user interface.

By adhering to a structured design, the system optimally handles user interactions, document processing, and validation procedures. The implementation of various use cases, including user data entry, document upload, and data validation, is designed to meet the project's objectives efficiently.

The emphasis on user-friendly interfaces ensures accessibility, while the feedback mechanism and corrective actions address discrepancies promptly. The system's adaptability to various document types and the support for multiple functionalities contribute to its versatility.

In essence, the Information Verification System provides a reliable and user-centric platform for maintaining data accuracy and integrity, enhancing the overall user experience. Future enhancements could explore further integration with emerging technologies to continuously evolve and meet evolving user needs.

5.2- FUTURE WORK

The "Information Verification system" is a substantial achievement, there is always room for improvement and expansion. Here are some ways for future development:

Machine Learning Document Authentication:

Implement a machine learning model, particularly using Convolutional Neural Networks (CNN), to analyze document images and ascertain their authenticity. Training the model on a diverse dataset can enhance the system's ability to identify genuine documents and mitigate the risk of fraudulent submissions.

Extended Document Format Support:

Broaden the system's capabilities by extending its support to multiple document formats, such as PDFs, images, and text files. This enhancement will ensure users can upload and validate information from a wider array of document types, increasing the system's versatility.

Diversification into Other Information Verification Fields:

Expand the scope of the Information Verification System to cater to various information verification needs beyond the current focus. This could include validating data in healthcare records, financial documents, or any other field where accurate and secure information verification is paramount.

Multilingual Data Extraction:

Currently, the system is designed to extract data primarily in English. A future enhancement could involve incorporating features to enable the extraction of information from documents in multiple languages. This would make the Information Verification System more inclusive and adaptable to a diverse range of linguistic inputs.

Enhanced Image Processing:

Implement advanced image processing algorithms and optical character recognition (OCR) techniques to improve the system's capability to extract information accurately from complex images and documents with varying layouts.

Integration of Biometric Data:

Explore the integration of biometric data validation, such as fingerprint or facial recognition, to add an additional layer of identity verification, enhancing the system's security and reliability.

Cross-Platform Compatibility:

Develop the system to be compatible with various platforms, including web browsers, mobile applications, and desktop environments, to ensure users can seamlessly access and utilize the system from their preferred devices.