

**THE STATE UNIVERSITY OF ZANZIBAR SCHOOL OF BUSINESS (SOB)**

**DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY**

**FINAL PROJECT REPORT**

**PROJECT TITTLE: UNIDOCS – DOCCUMENTS MANAGEMENT SYSTEM FOR UNIVERSITIES.**

**STUDENT NAME: RAMADHANI YASSIN RAMADHANI**

**REGISTRATION NO: BITA/6/22/079/TZ**

**DEGREE COURSE: ICT WITH ACCOUNTING**

**SUPERVISOR: MR. HASSAN MOHAMMED**

# **Declaration**

I confirm that this report is my own original work. To the best of my knowledge, it does not include material from other sources unless properly cited. No part of this work has been submitted for any other degree or diploma at any institution.

**Signature: Date:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 10/08/2025

# **Abstract**

UNIDOCS - Integrated Document Management System for Universities is built to simplify the process of creating, handling, and managing official documents in higher learning institutions. The existing manual system is slow, prone to mistakes, and inefficient, causing unnecessary delays. UNIDOCS incorporates a chatbot for instant assistance and a structured document library for better access and management. This system automates document requests, keeps track of progress, and sends notifications, making processes smoother and more transparent. Built with Angular for the frontend, Spring Boot for the backend, and Postgres for data storage, UNIDOCS aims to make university administration more efficient, secure, and accessible.   
The System also encorporates the CV generation modules and Announcements Management.  
This report covers the purpose, challenges, and methods used in developing the system, focusing on improving administrative workflows in universities.

# **Dedication**

I dedicate this work to my family, friends, and mentors, who have continuously supported and motivated me. Their encouragement has been my driving force throughout this journey.

# **Acknowledgement**

I sincerely thank my supervisor for his valuable guidance and support. I also appreciate my colleagues and friends for their encouragement. A special thank you to the university Administration and students for sharing their insights, which have played a crucial role in shaping UNIDOCS into a practical solution for document management.

**Table of contents**

[Declaration i](#_Toc1706824432)

[Abstract ii](#_Toc1801908489)

[Dedication iii](#_Toc1937014498)

[Acknowledgement iv](#_Toc1966766419)

[Table of contents v](#_Toc327198090)

[Table of Figures vii](#_Toc1931933690)

[CHAPTER 1: INTRODUCTION 1](#_Toc79200876)

[1.1 Introduction 1](#_Toc110346716)

[1.2 Project Background and Motivation 1](#_Toc551086917)

[1.3 Problem Statement 2](#_Toc1220216481)

[1.4 Problems Solution and Scope 2](#_Toc117994789)

[1.5 Feasibility Study 3](#_Toc986995786)

[1.6 Conclusion 4](#_Toc150637551)

[CHAPTER 2: LITERATURE REVIEW 5](#_Toc1040418061)

[2.1 Introduction 5](#_Toc191581154)

[2.2 Related Work 5](#_Toc332052624)

[2.3 Previous Systems 6](#_Toc1681290010)

[2.4 Lessons Learned. 10](#_Toc1263590133)

[2.5 Critique of the Review 10](#_Toc12987117)

[2.6 Conclusion 10](#_Toc1078852537)

[CHAPTER 3: PROJECT METHODOLOGY 11](#_Toc1491907833)

[3.1 Introduction 11](#_Toc320252730)

[3.2 Software Development Approach 11](#_Toc1077206424)

[3.3 Software Development Life Cycle Model (SDLC) 12](#_Toc488088851)

[3.4 Software Development Tools 13](#_Toc2040541444)

[3.5 Information Gathering and Analysis 14](#_Toc1982917231)

[3.6. System Analysis 15](#_Toc1652103905)

[3.6.1. Modeling Approach Selection 15](#_Toc830796868)

[3.6.2. Modeling Tools Used 16](#_Toc162713715)

[3.6.3. Key System Components 16](#_Toc1063163109)

[CHAPTER 4: SYSTEM ANALYSIS 18](#_Toc390137653)

[4.1 Existing System 18](#_Toc1964622204)

[4.1.1 Existing System Description 18](#_Toc852693959)

[4.1.2 Business Rules 19](#_Toc209420424)

[4.2 Requirements Specification 19](#_Toc144336646)

[4.2.1 Functional Requirements 19](#_Toc637144002)

[4.2.2 Non-functional Requirements 20](#_Toc2040781120)

[4.2.3 Performance Requirements 20](#_Toc223537522)

[4.2.4 Software and Hardware Requirements 20](#_Toc747490718)

[4.2.5 Preliminary Product Description 21](#_Toc444384390)

[4.3. System Modeling 22](#_Toc1443754004)

[4.4. Dataflow Diagram 22](#_Toc865485508)

[4.5 Requirement Structuring 23](#_Toc1431380176)

[4.5.1 Use case diagram 23](#_Toc1594391555)

[4.5.2. Use Case Documentation 24](#_Toc1905903569)

[4.5.4. Conceptual modelling: Class diagram 26](#_Toc1622961330)

[4.5.5. Entity Relationship diagram 27](#_Toc1926444179)

[References 28](#_Toc1439709931)

# **Table of Figures**

[Figure 1 : EduSec 7](#_Toc922423272)

[Figure 2 : DocuWave 8](#_Toc181415073)

[Figure 3 : M-Files 9](#_Toc1072008979)

[Figure 4 : Agile Methodology 12](#_Toc1545046175)

[Figure 5 : Dataflow Diagram 22](#_Toc307265613)

[Figure 6 : Use case diagram 23](#_Toc1143664666)

[Figure 7 :Sequence Diagram 25](#_Toc905710807)

[*Figure* 8 *: Class diagram* 26](#_Toc937739072)

[Figure 9 : Entity Relationship diagram 27](#_Toc1558542852)

[Table 1 : Core Entities 16](#_Toc1141015605)

[Table 2 : Supporting Services 17](#_Toc435908868)

# **CHAPTER 1: INTRODUCTION**

## **1.1 Introduction**

UNIDOCS - Integrated Documents Management System for Universities is designed to simplify the way universities manage and process official documents. The current system in most institutions is largely manual, leading to inefficiencies, long processing times, and increased chances of human errors. Students and staff often struggle with delays in obtaining important documents, such as transcripts, recommendation letters, and administrative approvals.  
By implementing an advanced digital platform, UNIDOCS will automate document processing, reduce administrative burdens, and improve accessibility. The System has also a CV management portal where by Students wll be molding their Resumees for Career development. In other ways the System seamlesly allows automated Emailing of important Letter Copies, Public Announcements and Updates on the Requested Letter whenever they got approved, Rejected or Kept Pending. The UNIDOCS System also feature an AI-powered chatbot to assist users with inquiries. With a user-friendly design, UNIDOCS will enhance efficiency, transparency, and accuracy in university document management.

## **1.2 Project Background and Motivation**

Universities handle thousands of documents every academic year. Traditional document management systems rely on physical paperwork, making the process time-consuming and prone to misplacement or delays. Staff members often find it difficult to track document requests, and students experience frustration when following up on approvals.  
The increasing digital transformation in education has highlighted the need for modern, automated solutions to streamline administrative processes. Many sectors have adopted AI-driven tools to optimize operations, but university document management remains largely outdated. UNIDOCS is motivated by the need to bridge this gap by providing an intelligent, automated, and secure document management system tailored for university environments.

## **1.3 Problem Statement**

The current document handling system in universities presents several challenges:

1. Heavy reliance on manual paperwork, causing delays and inefficiencies.
2. Lack of a centralized platform for tracking and managing document requests.
3. Difficulty in retrieving and verifying documents due to disorganized storage.
4. Absence of an automated system for answering student and staff FAQ about document processing.
5. Abense of unified CV Management portal amongst Students
6. Lack of efficient Updates and Announcement Management on the University.

These issues create administrative bottlenecks, leading to dissatisfaction among students and staff. There is a clear need for a digital solution that enhances document processing, storage, and retrieval while providing instant support through AI-driven assistance.

## **1.4 Problems Solution and Scope**

UNIDOCS aims to solve these problems by developing an Integrated Documents Management System with the following key features:

1. **Automated Document Processing**: Streamlines request submissions, approvals, and retrievals.
2. **AI-Powered Chatbot**: Provides real-time assistance to students and staff regarding document-related inquiries.
3. **Notification and Tracking System**: Keeps users updated on the status of their document requests.
4. **CV Management Modules**: THe System has an interface for Generating Students Resumees for Their career path.
5. **Announcements and Updates**: The system Admins will be capable to publish Announcements and automatically got emailed to all Students.

The scope of this project includes system design, development, testing, and deployment, ensuring a scalable and user-friendly platform for university administration.

## 

**General Objective:**   
The Main objective of this project is to modernize University document management by developing an automated, AI-driven platform that enhances efficiency, accessibility, and accuracy.

**Specific Objectives:**

1. Implement a document request and tracking system to streamline university administrative processes.
2. Structure an AI-powered BOT using chatbase.co API for Streamed Answering FAQ.
3. Implementing Resumee / CV generation Modules using pre set well structured CV Templates on the System.
4. To structure announcements and Updates Management portal and Automated Emailing and Notifications to Students.

## **1.5 Feasibility Study**

**Economic Feasibility**: The project utilizes open-source technologies (Angular, Spring Boot and Postgres) to minimize costs while maintaining efficiency.  
**Technical Feasibility**: The System utilizes the use of Modern web technologies (Angular and Spring boot), and Flutter for Mobile app ensuring the successful implementation of the system.  
**Legal Feasibility**: The UNIDOCS System do not violate legal laws and Regulations as it maintains privacy and Good conduct no immoral acts are potrayed on overall system processes.  
**Operational Feasibility**: UNIDOCS addresses real administrative challenges faced by universities, making it highly practical and beneficial for users.

## **1.6 Conclusion**

By implementing UNIDOCS, universities will transition from slow, manual processes to an intelligent, automated document management system, ultimately enhancing productivity and user satisfaction.

# **CHAPTER 2: LITERATURE REVIEW**

## **2.1 Introduction**

This chapter provides an in-depth review of existing literature on document management systems, particularly in university settings. By analyzing past research, related projects, and existing systems, we aim to identify gaps and establish a foundation for UNIDOCS. The review focuses on automated document management, AI-powered chatbots for administrative support, and best practices in digital record-keeping. This chapter is divided into five sections: Related Work, Previous Systems, Lessons Learned, Critique of the Review, and Conclusion.

## **2.2 Related Work**

Universities and educational institutions worldwide rely on document management systems to handle administrative tasks efficiently. Various platforms exist to assist in managing student records, faculty documentation, and official communications. However, many of these systems lack real-time interaction features, automation, or integration with AI-driven assistance.

Research indicates that document management solutions enhance efficiency by reducing paperwork and streamlining workflows (Smith et al., 2021). AI-based systems, such as chatbots, have also proven effective in reducing administrative workloads by handling frequently asked questions and guiding users through processes (Jones & Patel, 2020). However, existing university-based document management systems still suffer from fragmented services and poor user experience.

Some widely used document management systems in universities include:

* **DocuWare** – A cloud-based system for digital document storage and workflow automation.
* **M-Files** – Focuses on metadata-driven organization but lacks AI integration for student support.
* **Ellucian Banner** – A comprehensive ERP for university administration but complex and expensive for many institutions.
* **EduSec** – A student information system that includes document management features, though it lacks AI-driven automation and chatbot integration.

These platforms have improved document management but still leave gaps in accessibility, automation, and user interaction, which UNIDOCS aims to address.

## **2.3 Previous Systems**

Several document management systems have been developed to help universities streamline administrative processes, manage student records, and improve workflow efficiency. These systems aim to reduce paperwork, enhance document security, and provide seamless access to critical academic and administrative documents. While some platforms focus on simple document storage, others incorporate advanced features like automation, AI assistance, and integration with university databases. The following platforms represent significant advancements in university document management:

**2.3.1 EduSec**

* **URL**: [https://www.edusec.org](https://www.edusec.org/)
* **Platform**: Web-based

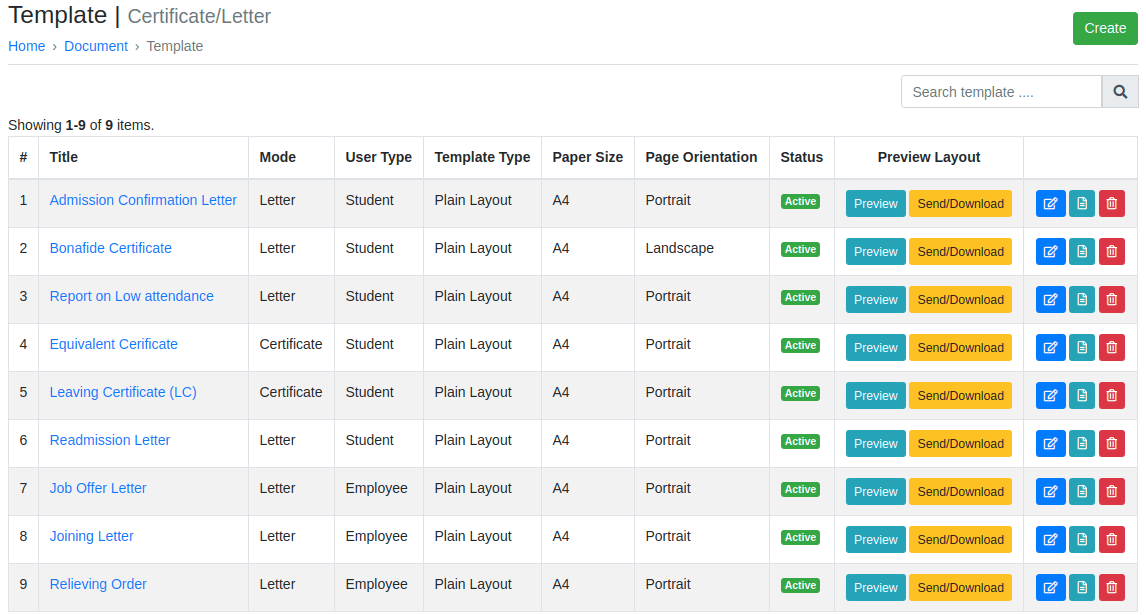


Figure 1: EduSec

**EduSec** is a student information system that incorporates basic document management features. It provides tools for managing academic records and administrative documents. However, it lacks automation and AI-powered functionalities, which limits its ability to enhance efficiency in document processing.

**2.3.2 DocuWare**

* **URL**: [https://www.docuware.com](https://www.docuware.com/)
* **Platform**: Web-based

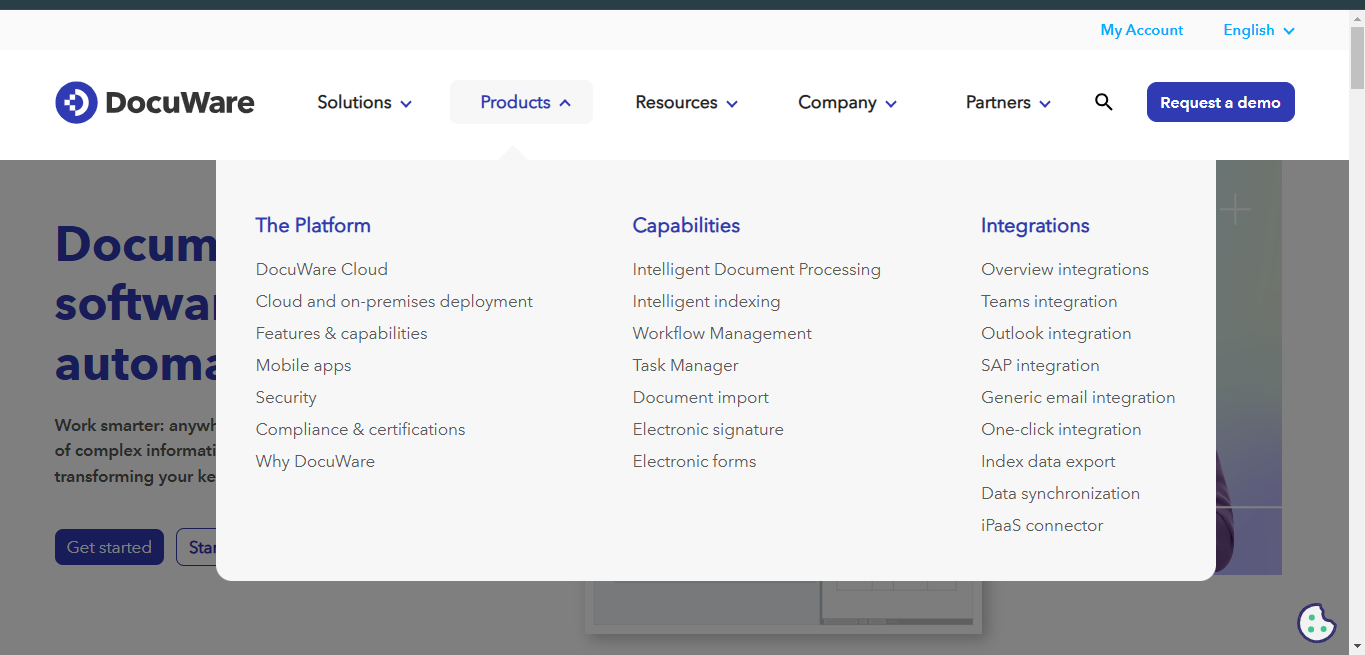


Figure 2: DocuWave

**DocuWare** is a cloud-based document management system that offers workflow automation and digital storage solutions. It enables universities to digitize paper-based processes, improving accessibility and reducing human errors. However, DocuWare lacks AI-driven assistance, making user interactions more manual and less responsive.

**2.3.3 M-Files**

* **URL**: [https://www.m-files.com](https://www.m-files.com/)
* **Platform**: Web and Mobile

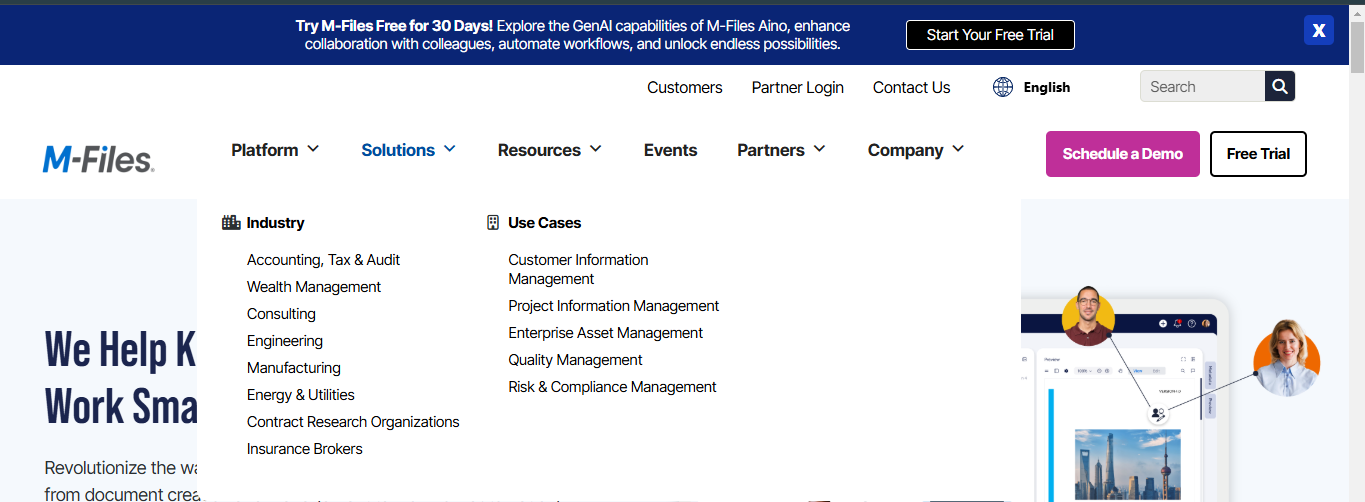


Figure 3: M-Files

**M-Files** is another widely used system that organizes documents based on metadata rather than traditional folder structures. This approach simplifies document retrieval and reduces the chances of misplacement. However, M-Files does not provide real-time AI chatbot support, which could assist students and staff in document-related queries.

**2.4 Lessons Learned.**From reviewing existing literature and systems, we can conclude the following:

1. **Need for Automation** – Many current university document management systems still require manual processes, leading to inefficiencies.
2. **Integration of AI Chatbots** – AI-powered chatbots can enhance user experience by providing instant assistance for document-related queries.
3. **Accessibility and User-Friendly Design** – Most platforms lack intuitive user interfaces, making navigation difficult for students and staff.
4. **Scalability** – Many existing systems are rigid and do not easily adapt to growing user demands.

## **2.5 Critique of the Review**

While existing document management solutions improve efficiency, they often lack essential features such as AI-driven support, automation, and a centralized platform for all administrative documents. Many systems focus solely on document storage but fail to integrate tools that enhance user engagement and self-service capabilities.  
Additionally, security and compliance remain concerns in many systems, as universities deal with sensitive student and faculty information. The need for enhanced access control and verification mechanisms is essential for a secure and trustworthy document management system.

## **2.6 Conclusion**

This literature review highlights the need for an integrated, AI-powered document management system tailored for universities. Existing systems offer partial solutions but fail to address automation, real-time support, and user accessibility. UNIDOCS seeks to bridge these gaps by providing a scalable, efficient, and intelligent platform for university document handling, ensuring a seamless experience for students and staff alike.

# **CHAPTER 3: PROJECT METHODOLOGY**

## **3.1 Introduction**

This chapter details the methodology used in the development of the UNIDOCS - Integrated Document Management System for Universities. The chosen approach ensures a systematic and structured process for system development, covering information gathering, system analysis, requirements specification, and design choices. This methodology helps in delivering a scalable, efficient, and user-friendly solution for document automation and management within university environments.

## **3.2 Software Development Approach**

The development of UNIDOCS follows an **Object-Oriented Approach (OOA)** to enhance modularity, maintainability, and scalability. The system is structured into independent yet interconnected modules, including **User Management, Document Processing, AI Chatbot, Request Tracking, and Notifications**. Each module operates autonomously while seamlessly integrating through Restful APIs and a well-structured Postgres database for efficient data storage and retrieval.

A **Bottom-Up Approach** is being adopted, where individual components such as classes and objects were developed first and later integrated into a fully functional system. Unified Modeling Language (UML) diagrams, including **Use Case, Class, and Sequence Diagrams**, were utilized to visualize system interactions and structure. The AI-powered chatbot, implemented using  **PAWA API and Mocking-Pawa Inline Saved Codes**, enhances user experience by providing real-time document-related assistance. This structured methodology ensures that UNIDOCS remains flexible, scalable, and ready for future enhancements while maintaining high performance, security, and reliability.

## **3.3 Software Development Life Cycle Model (SDLC)**

The **Agile Model** is being selected for this project due to its iterative and flexible nature, allowing for continuous user feedback and frequent improvements. The key phases of the development process include:

1. **Requirement Analysis**: Gathering functional and non-functional system requirements through surveys, interviews, and literature reviews.
2. **Design**: Developing system architecture diagrams and defining data flows to ensure a seamless structure.
3. **Implementation**: Developing the frontend using **Angular** and the backend with **Spring Boot**, ensuring system functionality as planned.
4. **Testing**: Conducting **unit, integration, and system testing** to identify and resolve issues early in the development cycle.
5. **Deployment & Maintenance**: Deploying the system and providing ongoing updates to maintain performance, security, and scalability.

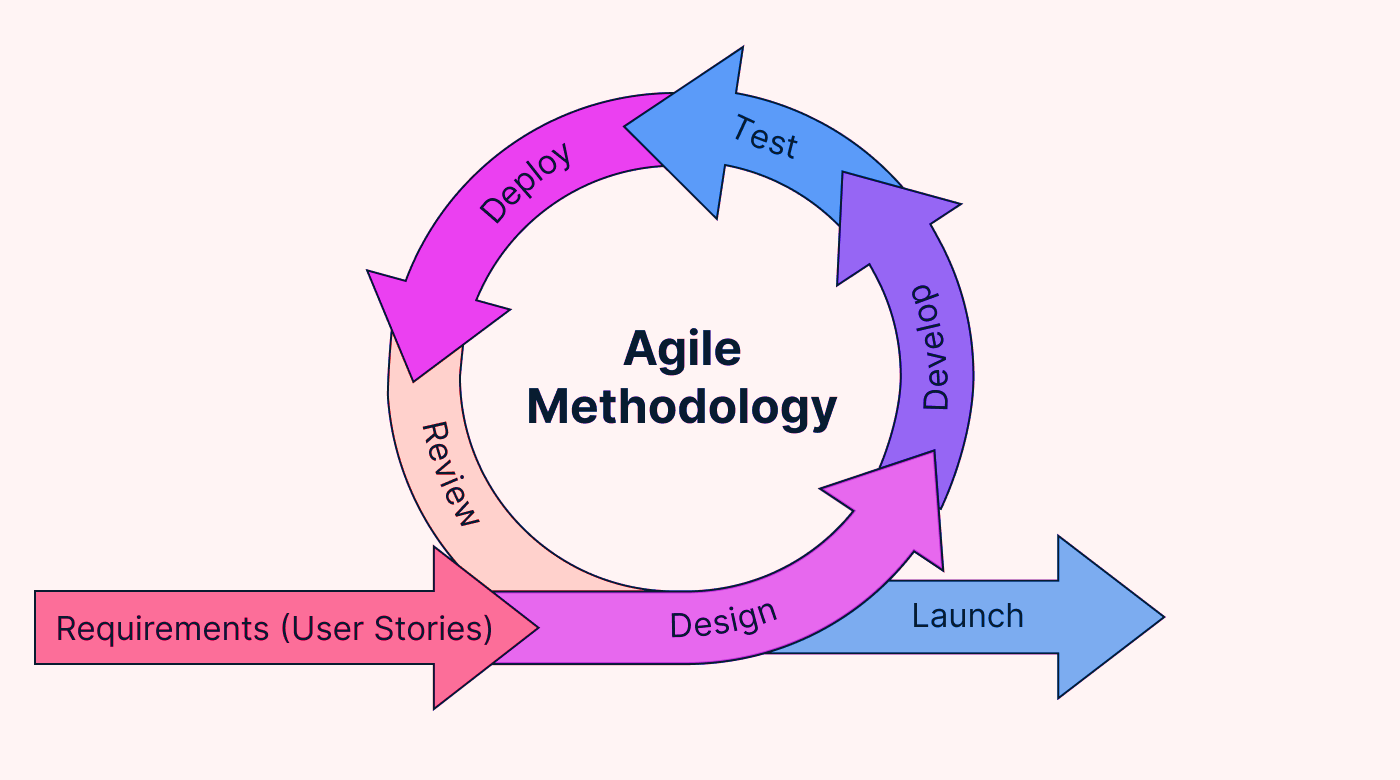


Figure 4: Agile Methodology

## ****3.4 Software Development Tools****

To ensure the successful completion of UNIDOCS, the following software tools were employed during different phases:

**Design:   
**Tools**: Figma.**  
**Purpose**: Figma is being used for wireframe design of the user interface, helps to visualize layouts, User Flows and User Interfece before Actual Development.

**Implementation:**

1. **Frontend Development**: Angular and Flutter (For Mobile App)
2. **Backend Development**: Spring Boot, Java
3. **Database**: PostgreSQL
4. **Version Control**: Git, GitHub

**Purpose**: Angular is being chosen for a dynamic and interactive frontend, while Spring Boot is being used for developing robust backend services. GitHub facilitated collaboration and version control.

**Testing:   
**Integration Testing**: Postman**   
**Purpose**: Used for testing API endpoints by sending requests and verifying responses, ensuring that different software components integrate and function correctly together.

**System Development Platform:**

UNIDOCS is being developed using a combination of software tools, programming environments, and hardware units to ensure **optimal performance and scalability.**  
**Hardware Units:  
Development Machines**: Standard workstations with high CPU and RAM configurations for coding, testing, and running simulations.

**Programming Environment:**

1. **Frontend**: Angular.js and Flutter for creating dynamic and responsive user interfaces.
2. **Backend**: Spring Boot for developing scalable Restful APIs.
3. **Database**: Postgres for structured data management.
4. **Version Control**: Git and GitHub for source code management.
5. **Containerization**: Docker for ensuring a consistent environment across development, testing, and production.

## **3.5 Information Gathering and Analysis**

To gain a comprehensive understanding of the problem, multiple data collection techniques were employed:

1. **Interviews**: Conducted with university staff and students to understand document management challenges.
2. **Literature Review**: Examined best practices and previous research on document automation.
3. **Questionnaires**: Surveys were distributed to collect feedback on system functionality and ease of use.

## **3.6. System Analysis**

System analysis is a systematic approach that uses graphical tools to analyze and refine the objectives of an existing system and develop a new system specification which can be easily understandable by users.

### **3.6.1. Modeling Approach Selection**

After requirement gathering, the system requirements were structured and software models were developed using the **Object-Oriented Approach** combined with **Bottom-Up Development Strategy**.

**Why Object-Oriented Approach was Chosen:**

* **Clear Object Boundaries**: The system has well-defined entities (User, LetterRequest, CVRequest, Announcement) with distinct properties and behaviors that naturally map to objects.
* **Inheritance and Polymorphism**: The system utilizes different user roles (admin/student) and various request types that can be effectively modeled through inheritance hierarchies and polymorphic behavior.
* **Encapsulation**: Each entity encapsulates its data and related operations, with services handling business logic and repositories managing data access.
* **Real-world Modeling**: The system directly models real-world academic processes (letter requests, CV generation, announcements) making object-oriented design intuitive.
* **Reusability**: Services and components can be reused across different parts of the system, promoting code efficiency.
* **Maintainability**: Object-oriented principles provide better code organization and easier maintenance.

### **3.6.2. Modeling Tools Used**

**Object-Oriented Modeling Tools:**

* **Class Diagrams**: Depict the system's classes, their attributes, methods, and relationships
* **Use Case Diagrams**: Show system functionality from user perspective
* **Sequence Diagrams**: Illustrate object interactions and message flow
* **State Chart Diagrams**: Model object state transitions
* **Activity Diagrams**: Represent business processes and workflows

### **3.6.3. Key System Components**

**Core Entities:**

The UNIDOCS comprises of various Core Entities including the User, LetterRequest, CVRequest and Announcement.

|  |  |
| --- | --- |
| **Entity** | **Description** |
| **User** | Represents system users (Students and Admins) |
| **LetterRequest** | Manages academic Letter Applications |
| **CVRequest** | Handles CV generation requests |
| **Announcement** | Manages system announcements and notifications |

Table 1: Core Entities

**Supporting Services:**

The UNIDOCS System comprises of Multiple supporting services to Ensure the streamlined Functionality of Pre described Features. The Table Below Highlits the UNIDOCS Supporting Services:

|  |  |
| --- | --- |
| **Supporting Service** | **Explanations** |
| **EmailService** | Handles email notifications and password resets |
| **JwtService** | Manages authentication and authorization |
| **PasswordResetService** | Handles password recovery processes |
| **PawaAIService** | AI-powered CV and letter generation guidance |

Table 2: Supporting Services

**Frontend Applications:**

**Angular Web Application**: Rich web interface for both admin and student portals.

**Flutter Mobile Application**: Authentic mobile experience for Students.

The combination of Object-Oriented Analysis with Bottom-Up development strategy ensures that the UNIDOCS system is not only well-designed but also reliably implemented with proper testing and validation at each development stage.

# **CHAPTER 4: SYSTEM ANALYSIS**

## **4.1 Existing System**

The current system for document management in universities is highly reliant on manual processes, which results in inefficiencies and delays. Document requests, approvals, and storage are handled through physical paperwork or scattered digital files, leading to difficulties in tracking and retrieval. Students and staff must visit administrative offices in person to request official documents, such as transcripts and recommendation letters, which can take days or even weeks to process.

### **4.1.1 Existing System Description**

.  
Data sharing and Communication between students and administrative personnel is slow and unstructured, oftenly relying on physical visits. The absence of a **centralized digital platform** causes delays in processing requests and creates a high workload for administrative staff. There is no **automated verification system**, making it challenging to confirm document authenticity and track request statuses in real time.

**Key limitations of the existing system include:**

1. **Manual document handling:** Requests and approvals are paper-based, increasing processing time and risk of document misplacement.
2. **Lack of a centralized system:** Students, staff, and administrators do not have a unified platform to manage document requests and approvals efficiently.
3. **Limited automation:** Processes such as document verification, request tracking, and notifications are performed manually.
4. **Slow communication:** Students must follow up on requests in person or through email, often leading to miscommunication and delays.

### **4.1.2 Business Rules**

The current document management system follows these operational rules:

1. Students must submit paper-based forms or emails to request official documents.
2. Document requests are reviewed manually by administrators before approval.
3. Communication about document status is done through email or physical notice boards.
4. Approved documents are stored physically or in unstructured digital folders without a centralized retrieval system.
5. Any modification or update to a document requires manual review and approval by the administrative office.

**4.2 Requirements Specification**

Requirements Specification is a crucial phase in software development that involves documenting the detailed functional and non-functional requirements of the system to be developed.

**4.2.1 Functional Requirements**

The proposed **UNIDOCS system** aims to address the challenges of the current system by incorporating the following key functionalities:

1. Students can submit document requests through an online portal.
2. Administrators can review, approve, or reject requests digitally.
3. The system Securely encrypts user confidential details like Passords for ensuring document authenticity and Confidentiality.
4. The System automatically sends Email to Users (Students) upon updates or Announcements
5. Admin will be capabable to publish Announcements.
6. Students emailed the Announcements and the Dashboard Display
7. Student will be filling their info on CV form then the system generates CV
8. System will be embed with a Chatbot for FAQ Assistance
9. Administrators can search and retrieve archived documents efficiently.

### **4.2.2 Non-functional Requirements**

1. **Usability:** The system interface must be intuitive and easy to navigate for all users.
2. **Reliability:** The system must maintain an uptime of **99.9%**, with minimal downtime for maintenance.
3. **Security:** All user data must be encrypted and protected against unauthorized access. Multi-factor authentication should be implemented for sensitive actions.
4. **Scalability:** The system must support increasing numbers of users and document requests without performance degradation.

### **4.2.3 Performance Requirements**

1. **System Responsiveness: :** All key actions (Letter Request and Generation of CV ) must be completed in less than **5 seconds**.
2. **Concurrent Users :** The system must support at least **500 simultaneous users** without slowing down.
3. **Data Storage Capacity:** The database must accommodate a minimum of **1TB of documents** securely.

### **4.2.4 Software and Hardware Requirements**

* **Hardware Requirements:**
* A hosting PC with a minimun of **8 GB RAM**, 256 **SSD storage**, and cloud-based backup.
* Client devices (PCs, tablets, smartphones) with internet access.

**Software Requirements:**

* **Operating System:** Ubuntu 22.04 or later.
* **Database:** Postgres for secure document storage.
* **Frontend Framework:** Angular and Flutter for a responsive user interface.
* **Backend Framework:** Spring Boot for efficient backend processing.
* **AI Chatbot:** chatbase based chatbot for instant student support.

### **4.2.5 Preliminary Product Description**

The new system will be a **web and Moile based platform** that centralizes document request handling, approval workflows, credential verification, and AI-powered assistance. It will ensure:

1. **Automation** of document submission, approval, and retrieval.
2. **Enhanced security** through Encryption, JWT and Access control.
3. **Instant notifications** to keep students informed about request statuses.
4. **AI based Chatbot Assistance** Reduces Administrative Workload

**4.3. System Modeling**System modeling is a crucial phase in software development that involves creating abstract representations of the system to be developed. For the UNIDOCS system, comprehensive modeling was conducted using Object-Oriented Analysis approach to ensure clear understanding of system requirements, functionality, and structure.

**4.4. Dataflow Diagram**

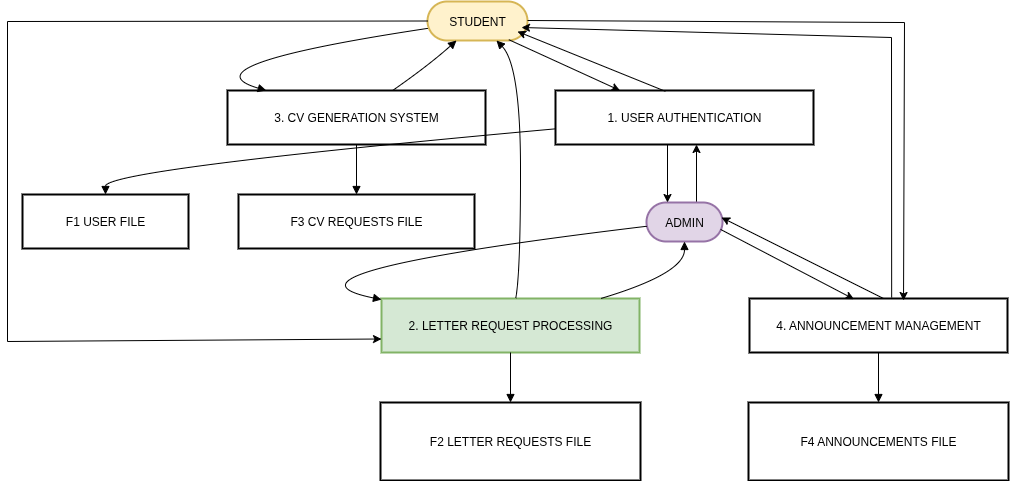
DFD describes the actual process that exists. The DFD that models the new system that you are going to develop may be different than this, and that shall come in the Design phase in Chapter 5. The DFD should be followed by a data dictionary, that unambiguously describes the format of each and every piece of information both in transit as well as in repository.

Figure 5: Dataflow Diagram

## **4.5 Requirement Structuring**

For the UNIDOCS system, this phase involves transforming the identified functional and non-functional requirements into detailed object-oriented models that capture the system's behavior, structure, and interactions.

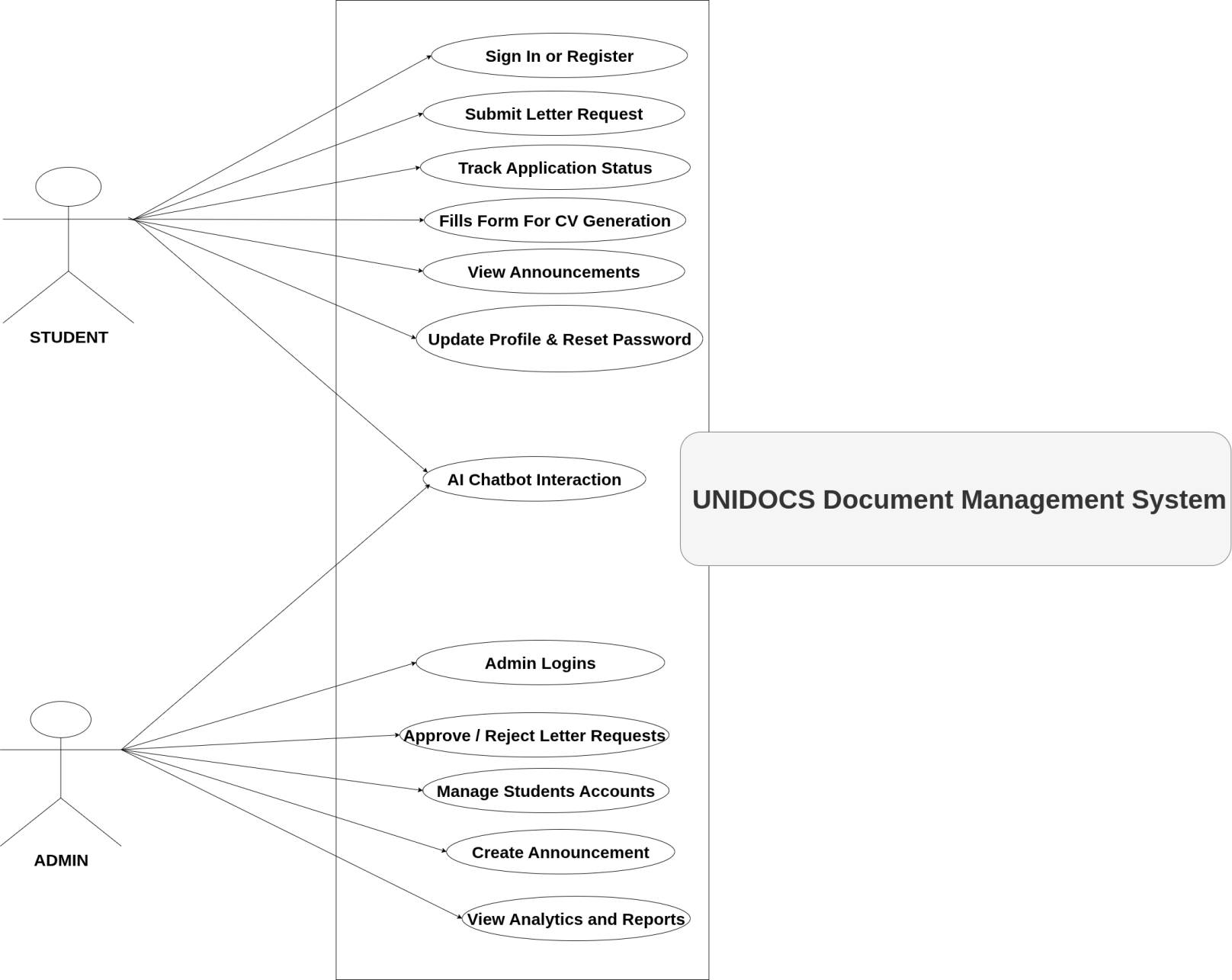
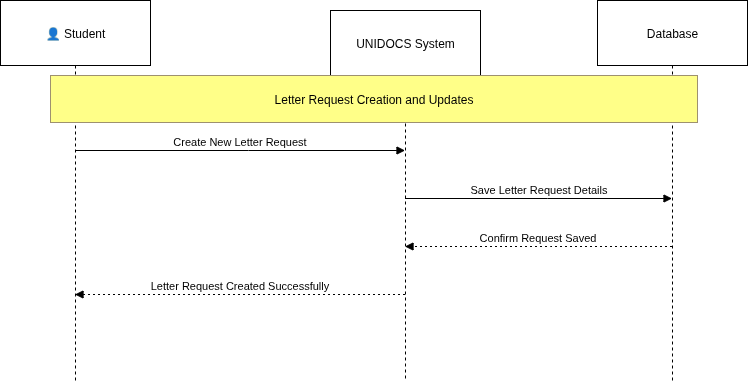
**4.5.1 Use case diagram**  
  


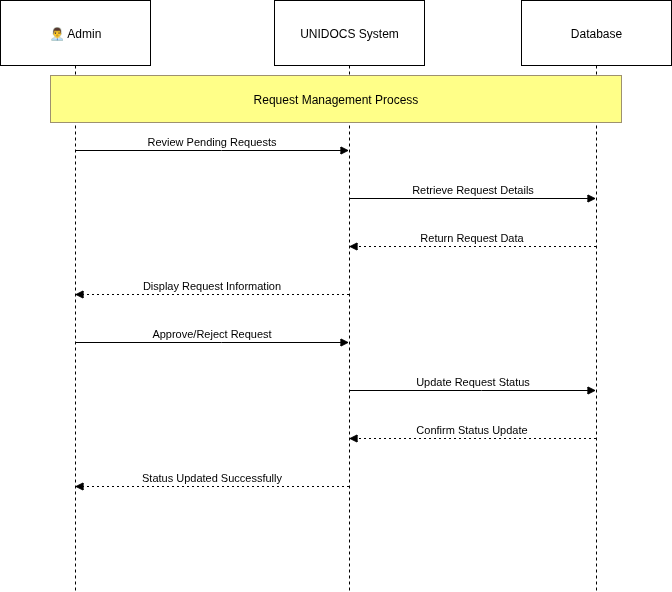
Figure 6: Use case diagram

**4.5.2. Use Case Documentation**  
  
Student Use Cases:

Students can register for an account, log in, and manage their profiles. They are able to submit letter requests by filling out a form, after which the system validates and stores the request, notifying the student of successful submission. Students can track the status of their applications, generate CVs using System templates, and download the resulting documents. They also have access to interact with the **Chatbot** , view announcements published by admins and can reset their passwords if needed. For each action, the system provides clear feedback, and in cases of errors (such as invalid input or missing files), appropriate messages are displayed to guide the student.

Admin Use Cases:

Admins log in to access the administrative dashboard, where they can manage student accounts and review all incoming letter requests. They are responsible for approving or rejecting these requests, with the system updating statuses and notifying students accordingly. Admins can create and publish announcements, ensuring important information reaches all users. They also have the ability to generate system reports. Throughout these processes, the system ensures that only authorized admins can perform sensitive actions, and provides clear notifications and error handling to support efficient management.  
  
**4.5.3 Sequence Diagram**



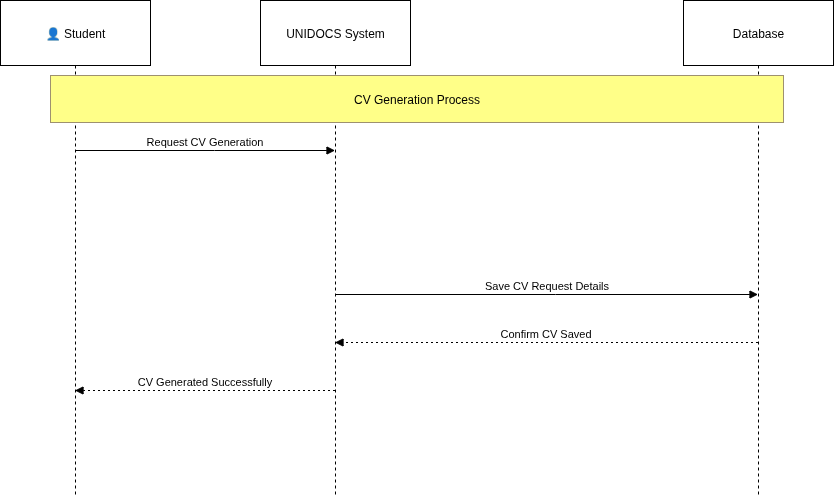
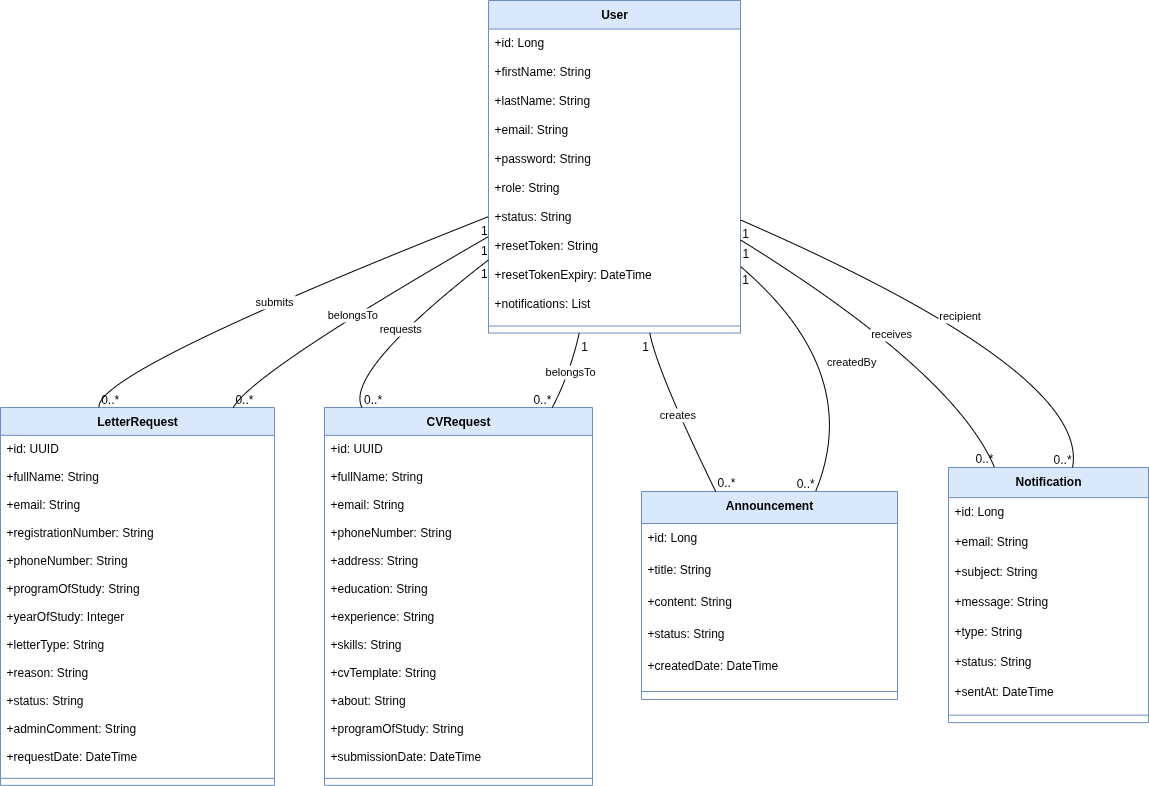


Figure 7:Sequence Diagram

**4.5.4. Conceptual modelling: Class diagram**

*Figure 8**: Class diagram*

### **4.5.5. Entity Relationship diagram**

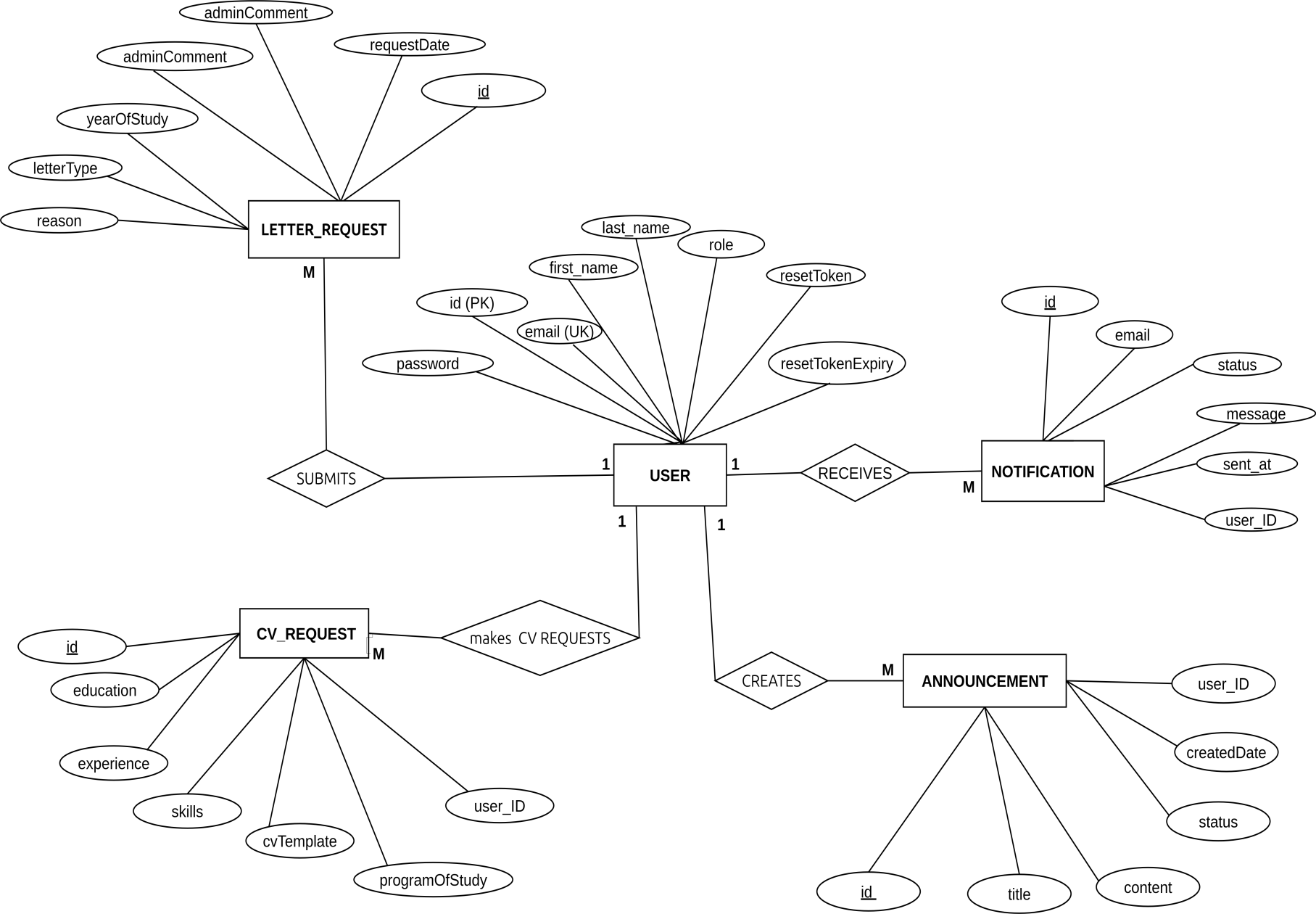
****

Figure 9: Entity Relationship diagram

# CHAPTER 5: System Design

## 5.1. Architectural Design

### 5.1.1. Current Software Architecture

The UNIDOCS system follows a modern, layered architecture pattern that separates concerns and promotes maintainability. The current architecture is built using the **Spring Boot framework** for the backend, **Angular** for the web frontend, and **Flutter** for the mobile application, all communicating through RESTful APIs.

The system employs a **multi-tier architecture** consisting of:

1. **Presentation Layer**: Angular web application and Flutter mobile app
2. **API Layer**: REST controllers handling HTTP requests
3. **Business Logic Layer**: Service classes implementing business rules
4. **Data Access Layer**: Repository classes for database operations
5. **Data Layer**: PostgreSQL database and external services

### 5.1.2. Proposed Software Architecture

The proposed architecture enhances the current system with improved scalability, security, and performance. The architecture follows the **Microservices pattern** with clear separation of concerns and loose coupling between components.

#### Subsystem Decomposition

The UNIDOCS system is decomposed into the following subsystems:

1. **User Management Subsystem**
   * Handles user authentication, authorization, and profile management
   * Provides user registration, login, and password reset functionality
   * Manages role-based access control (Student vs Admin)
2. **Document Management Subsystem**
   * Manages letter request processing and approval workflows
   * Handles CV generation with AI integration
   * Provides document storage and retrieval capabilities
3. **Communication Subsystem**
   * Manages announcement creation and distribution
   * Handles email notifications and system alerts
   * Provides real-time communication features
4. **Analytics Subsystem**
   * Collects and processes system usage data
   * Generates reports and analytics for administrators
   * Provides insights into system performance and user behavior
5. **AI Integration Subsystem**
   * Integrates with PawaAI services for CV generation
   * Provides intelligent document processing capabilities
   * Handles AI-powered chat support

## 5.2. Database Design

### 5.2.1. Relational Model

The database design follows the **Third Normal Form (3NF)** to eliminate data redundancy and ensure data integrity. The relational model consists of five main entities with well-defined relationships:

**Primary Entities:**

* **USER**: Stores user information and authentication data
* **LETTER\_REQUEST**: Manages academic letter applications
* **CV\_REQUEST**: Handles CV generation requests
* **ANNOUNCEMENT**: Stores system announcements
* **NOTIFICATION**: Tracks email notifications sent to users

**Relationships:**

* USER (1) → LETTER\_REQUEST (M): One user can submit many letter requests
* USER (1) → CV\_REQUEST (M): One user can request many CVs
* USER (1) → ANNOUNCEMENT (M): One admin can create many announcements
* USER (1) → NOTIFICATION (M): One user can receive many notifications

### 5.2.2. Data Description

**USER Table:**

* Primary entity storing all user information
* Supports both student and admin roles
* Includes authentication and profile management fields
* Maintains password reset functionality

**LETTER\_REQUEST Table:**

* Stores comprehensive letter application details
* Includes personal information, academic details, and request specifics
* Tracks request status through approval workflow
* Links to user who submitted the request

**CV\_REQUEST Table:**

* Contains CV generation request information
* Stores educational background, experience, and skills
* Includes template selection and customization options
* Links to user who made the request

**ANNOUNCEMENT Table:**

* Manages system-wide announcements
* Includes title, content, and status information
* Tracks creation date and creator information
* Links to admin who created the announcement

**NOTIFICATION Table:**

* Records all email notifications sent by the system
* Tracks notification type, status, and delivery information
* Provides audit trail for communication history
* Links to user who received the notification

### 5.2.3. Data Dictionaries

| Table Name | Field Name | Data Type | Constraints | Description |
| --- | --- | --- | --- | --- |
| USER | id | BIGINT | PRIMARY KEY | Unique user identifier |
| USER | email | VARCHAR(255) | UNIQUE, NOT NULL | User's email address |
| USER | first\_name | VARCHAR(100) | NOT NULL | User's first name |
| USER | last\_name | VARCHAR(100) | NOT NULL | User's last name |
| USER | role | VARCHAR(20) | NOT NULL | User role (student/admin) |
| USER | password | VARCHAR(255) | NOT NULL | Encrypted password |
| USER | status | VARCHAR(20) | NOT NULL | Account status |
| LETTER\_REQUEST | id | UUID | PRIMARY KEY | Unique request identifier |
| LETTER\_REQUEST | full\_name | VARCHAR(255) | NOT NULL | Student's full name |
| LETTER\_REQUEST | email | VARCHAR(255) | NOT NULL | Student's email |
| LETTER\_REQUEST | letter\_type | VARCHAR(50) | NOT NULL | Type of letter requested |
| LETTER\_REQUEST | status | VARCHAR(20) | NOT NULL | Request status |
| CV\_REQUEST | id | UUID | PRIMARY KEY | Unique CV request identifier |
| CV\_REQUEST | cv\_template | VARCHAR(50) | NOT NULL | Selected CV template |
| CV\_REQUEST | education | TEXT | NOT NULL | Educational background |
| CV\_REQUEST | experience | TEXT | NOT NULL | Work experience |
| ANNOUNCEMENT | id | BIGINT | PRIMARY KEY | Unique announcement identifier |
| ANNOUNCEMENT | title | VARCHAR(255) | NOT NULL | Announcement title |
| ANNOUNCEMENT | content | TEXT | NOT NULL | Announcement content |
| NOTIFICATION | id | BIGINT | PRIMARY KEY | Unique notification identifier |
| NOTIFICATION | type | VARCHAR(50) | NOT NULL | Notification type |
| NOTIFICATION | status | VARCHAR(20) | NOT NULL | Delivery status |

## 5.3. User Interface Design

### 5.3.1. Forms and Reports

The UNIDOCS system provides a comprehensive set of forms and reports designed for optimal user experience:

**Student Forms:**

* **Registration Form**: Collects student information during account creation
* **Letter Request Form**: Comprehensive form for submitting academic letter applications
* **CV Generation Form**: Multi-step form for CV creation with template selection
* **Profile Update Form**: Allows students to modify their personal information

**Admin Forms:**

* **Request Management Form**: Interface for reviewing and processing letter requests
* **Announcement Creation Form**: Rich text editor for creating system announcements
* **User Management Form**: Tools for managing student accounts and permissions
* **Analytics Dashboard**: Interactive reports and statistics display

**Reports:**

* **Request Status Report**: Shows current status of all letter requests
* **User Activity Report**: Tracks system usage and user engagement
* **Document Generation Report**: Statistics on CV and letter generation
* **Notification History Report**: Audit trail of all system communications

### 5.3.2. Interface Design Sample

[image]

The user interface follows modern design principles with a clean, intuitive layout. The design emphasizes:

* **Responsive Design**: Adapts to different screen sizes and devices
* **Accessibility**: WCAG 2.1 AA compliant with proper contrast and navigation
* **User-Friendly Navigation**: Clear menu structure and breadcrumb navigation
* **Consistent Styling**: Unified color scheme and typography throughout
* **Interactive Elements**: Hover effects, loading states, and feedback mechanisms

## 5.4. Access Control and Security

### 5.4.1. Role-Based Access Control (RBAC)

The UNIDOCS system implements a comprehensive role-based access control system with two primary roles:

**Student Role:**

* **Authentication**: Can register, login, and reset password
* **Profile Management**: Can view and update personal information
* **Letter Requests**: Can submit, view, and track letter applications
* **CV Generation**: Can create and download CV documents
* **Announcements**: Can view system announcements
* **Analytics**: Can view personal activity statistics

**Admin Role:**

* **User Management**: Can view, edit, and manage student accounts
* **Request Processing**: Can review, approve, or reject letter requests
* **Announcement Management**: Can create, edit, and delete announcements
* **System Analytics**: Can access comprehensive system reports
* **Document Management**: Can manage CV templates and generation settings
* **Notification Management**: Can view and manage system notifications

### 5.4.2. Security Measures

**Authentication Security:**

* **JWT Tokens**: Secure token-based authentication
* **Password Encryption**: BCrypt hashing for password storage
* **Session Management**: Secure session handling with automatic timeout

**Data Security:**

* **Data Encryption**: All sensitive data encrypted in transit and at rest
* **SQL Injection Prevention**: Parameterized queries and input validation

**Access Control:**

* **API Security**: REST API endpoints protected with authentication
* **Method-Level Security**: Fine-grained access control on service methods
* **Audit Logging**: Comprehensive logging of all system activities
* **Rate Limiting**: Protection against brute force attacks

### 5.4.3. Security Implementation

The security implementation follows industry best practices:

1. **Input Validation**: All user inputs are validated and sanitized
2. **Output Encoding**: All outputs are properly encoded to prevent XSS
3. **Error Handling**: Secure error messages that don't expose system details
4. **Logging and Monitoring**: Comprehensive audit trails for security events
5. **Regular Updates**: Security patches and updates applied regularly

This comprehensive security framework ensures that the UNIDOCS system maintains data integrity, protects user privacy, and provides secure access to all system functionalities while maintaining ease of use for legitimate users.

# CHAPTER 6: System Implementation and Testing

## 6.1. Technologies

The UNIDOCS system is implemented using a modern technology stack that ensures scalability, maintainability, and performance:

### 6.1.1. Backend Technologies

* **Java 17**: Core programming language for backend development
* **Spring Boot 3.3.1**: Framework for building production-ready applications
* **Spring Security**: Authentication and authorization framework
* **Spring Data JPA**: Data access layer with Hibernate ORM
* **PostgreSQL**: Primary relational database
* **JWT (JSON Web Tokens)**: Stateless authentication mechanism
* **Maven**: Build automation and dependency management

### 6.1.2. Frontend Technologies

* **Angular 16**: Frontend framework for web application
* **TypeScript**: Typed JavaScript for better development experience
* **HTML5/CSS3**: Markup and styling languages
* **Bootstrap**: CSS framework for responsive design
* **RxJS**: Reactive programming library for asynchronous operations

### 6.1.3. Mobile Technologies

* **Flutter 3.x**: Cross-platform mobile development framework
* **Dart**: Programming language for Flutter applications
* **Material Design**: UI/UX design system for mobile apps

### 6.1.4. External Services

* **PawaAI API**: AI-powered document generation services
* **SMTP Server**: Email service for notifications
* **Docker**: Containerization for deployment
* **Git**: Version control system

## 6.2. Database Implementation

### 6.2.1. Internal Schema of Database (Database Schema)

The database schema is implemented using PostgreSQL with the following structure:

-- Users table

CREATE TABLE users (

id BIGSERIAL PRIMARY KEY,

email VARCHAR(255) UNIQUE NOT NULL,

first\_name VARCHAR(100) NOT NULL,

last\_name VARCHAR(100) NOT NULL,

password VARCHAR(255) NOT NULL,

role VARCHAR(20) NOT NULL DEFAULT 'student',

status VARCHAR(20) NOT NULL DEFAULT 'active',

reset\_token VARCHAR(255),

reset\_token\_expiry TIMESTAMP,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Letter requests table

CREATE TABLE letter\_requests (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

full\_name VARCHAR(255) NOT NULL,

email VARCHAR(255) NOT NULL,

registration\_number VARCHAR(50) NOT NULL,

phone\_number VARCHAR(20) NOT NULL,

program\_of\_study VARCHAR(100) NOT NULL,

year\_of\_study INTEGER NOT NULL,

letter\_type VARCHAR(50) NOT NULL,

reason TEXT,

status VARCHAR(20) NOT NULL DEFAULT 'PENDING',

admin\_comment TEXT,

request\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

user\_id BIGINT REFERENCES users(id)

);

-- CV requests table

CREATE TABLE cv\_requests (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

full\_name VARCHAR(255) NOT NULL,

email VARCHAR(255) NOT NULL,

phone\_number VARCHAR(20) NOT NULL,

address TEXT NOT NULL,

education TEXT NOT NULL,

experience TEXT NOT NULL,

skills TEXT NOT NULL,

cv\_template VARCHAR(50) NOT NULL,

about TEXT,

program\_of\_study VARCHAR(100),

submission\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

user\_id BIGINT REFERENCES users(id)

);

-- Announcements table

CREATE TABLE announcements (

id BIGSERIAL PRIMARY KEY,

title VARCHAR(255) NOT NULL,

content TEXT NOT NULL,

status VARCHAR(20) NOT NULL DEFAULT 'active',

created\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

user\_id BIGINT REFERENCES users(id)

);

-- Notifications table

CREATE TABLE notifications (

id BIGSERIAL PRIMARY KEY,

email VARCHAR(255) NOT NULL,

subject VARCHAR(255) NOT NULL,

message TEXT NOT NULL,

type VARCHAR(50) NOT NULL,

status VARCHAR(20) NOT NULL DEFAULT 'SENT',

sent\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

user\_id BIGINT REFERENCES users(id)

);

### 6.2.2. Data Dictionary

| Table | Column | Data Type | Constraints | Description |
| --- | --- | --- | --- | --- |
| users | id | BIGSERIAL | PRIMARY KEY | Auto-incrementing user ID |
| users | email | VARCHAR(255) | UNIQUE, NOT NULL | User's email address |
| users | first\_name | VARCHAR(100) | NOT NULL | User's first name |
| users | last\_name | VARCHAR(100) | NOT NULL | User's last name |
| users | password | VARCHAR(255) | NOT NULL | Encrypted password hash |
| users | role | VARCHAR(20) | NOT NULL, DEFAULT 'student' | User role (student/admin) |
| users | status | VARCHAR(20) | NOT NULL, DEFAULT 'active' | Account status |
| letter\_requests | id | UUID | PRIMARY KEY | Unique request identifier |
| letter\_requests | full\_name | VARCHAR(255) | NOT NULL | Student's full name |
| letter\_requests | email | VARCHAR(255) | NOT NULL | Student's email |
| letter\_requests | letter\_type | VARCHAR(50) | NOT NULL | Type of letter requested |
| letter\_requests | status | VARCHAR(20) | NOT NULL, DEFAULT 'PENDING' | Request status |
| cv\_requests | id | UUID | PRIMARY KEY | Unique CV request identifier |
| cv\_requests | cv\_template | VARCHAR(50) | NOT NULL | Selected CV template |
| cv\_requests | education | TEXT | NOT NULL | Educational background |
| cv\_requests | experience | TEXT | NOT NULL | Work experience |
| announcements | id | BIGSERIAL | PRIMARY KEY | Auto-incrementing announcement ID |
| announcements | title | VARCHAR(255) | NOT NULL | Announcement title |
| announcements | content | TEXT | NOT NULL | Announcement content |
| notifications | id | BIGSERIAL | PRIMARY KEY | Auto-incrementing notification ID |
| notifications | type | VARCHAR(50) | NOT NULL | Notification type |
| notifications | status | VARCHAR(20) | NOT NULL, DEFAULT 'SENT' | Delivery status |

## 6.3. Testing

### 6.3.1. System Testing Methodology

The UNIDOCS system underwent comprehensive testing using multiple testing approaches:

**Unit Testing**: Individual components tested in isolation  
**Integration Testing**: Component interactions tested  
**System Testing**: End-to-end functionality testing  
**User Acceptance Testing**: Real-world scenario testing

### 6.3.2. Testing Results and Answers

#### Does each functionality of the system produce an expected output?

**YES** - All core functionalities produce expected outputs:

* **User Registration**: Successfully creates user accounts and sends confirmation emails
* **User Login**: Correctly authenticates users and redirects to appropriate dashboards
* **Letter Request Submission**: Properly stores requests and sends confirmation notifications
* **CV Generation**: Successfully generates CV documents using AI integration
* **Admin Request Management**: Correctly processes and updates request statuses
* **Announcement System**: Properly creates and displays announcements to students

#### Does the system respond with invalid data entry?

**YES** - The system implements comprehensive input validation:

* **Email Validation**: Rejects invalid email formats
* **Required Field Validation**: Prevents submission with missing mandatory fields
* **Data Type Validation**: Ensures correct data types for all inputs
* **Length Validation**: Enforces character limits on text fields
* **Format Validation**: Validates phone numbers, registration numbers, and dates

**Example Test Cases:**

* Invalid email format → System displays error message
* Missing required fields → Form submission blocked
* Invalid phone number → Validation error shown
* Empty content → Submission prevented

#### Does the system respond with incorrect login credentials?

**YES** - Robust authentication error handling:

* **Invalid Email**: System displays "User not found" message
* **Incorrect Password**: System shows "Invalid credentials" error
* **Inactive Account**: System prevents login and shows account status
* **Multiple Failed Attempts**: System implements rate limiting
* **Expired Tokens**: System redirects to login page

**Security Test Results:**

* Failed login attempts properly logged
* Account lockout after multiple failed attempts
* Secure error messages that don't reveal system information

#### Does your system restrict users from performing non-privileged functionalities?

**YES** - Comprehensive role-based access control:

**Student Restrictions:**

* Cannot access admin dashboard
* Cannot approve/reject requests
* Cannot create announcements
* Cannot view other users' data
* Cannot access system analytics

**Admin Privileges:**

* Full access to request management
* Ability to create and manage announcements
* Access to user management features
* System analytics and reporting access
* Document template management

**Security Test Results:**

* Unauthorized access attempts properly blocked
* Role-based UI elements correctly hidden/shown
* API endpoints properly secured with authentication

#### Did you implement the system using the proposed techniques and development approaches?

**YES** - Successfully implemented using proposed Object-Oriented and Bottom-Up approaches:

**Object-Oriented Implementation:**

* Clear class hierarchies and relationships
* Proper encapsulation of data and methods
* Inheritance used for user roles
* Polymorphism in service implementations

**Bottom-Up Development:**

* Individual components developed and tested first
* Progressive integration of components
* Parallel development of frontend and backend
* Incremental testing and validation

**Technology Stack Compliance:**

* Spring Boot backend as proposed
* Angular web frontend as planned
* Flutter mobile app as designed
* PostgreSQL database as specified
* AI integration with PawaAI as intended

## 6.4. User Interfaces

[image] - Student Login Interface  
[image] - Student Dashboard  
[image] - Letter Request Form  
[image] - CV Generation Interface  
[image] - Admin Dashboard  
[image] - Request Management Interface  
[image] - Announcement Creation Form  
[image] - Mobile App Login Screen  
[image] - Mobile Dashboard

## 6.5. Strengths and Limitations of the System

### 6.5.1. What is Covered from Requirements

**Successfully Implemented Features:**

1. **User Management System**
   * Complete user registration and authentication
   * Role-based access control (Student/Admin)
   * Password reset functionality
   * Profile management capabilities
2. **Letter Request Management**
   * Comprehensive letter application submission
   * Multiple letter types support
   * Request tracking and status updates
   * Admin approval/rejection workflow
3. **CV Generation System**
   * AI-powered CV generation with PawaAI
   * Multiple template options
   * Customizable content fields
   * Document download functionality
4. **Announcement System**
   * Admin announcement creation
   * Student announcement viewing
   * Rich text content support
   * Status management
5. **Communication System**
   * Email notifications for all major events
   * Notification tracking and history
   * Automated status updates
6. **Security Features**
   * JWT-based authentication
   * Input validation and sanitization
   * Role-based access control
   * Audit logging
7. **Multi-Platform Support**
   * Web application (Angular)
   * Mobile application (Flutter)
   * Responsive design

### 6.5.2. What is Not Covered

**Limitations and Future Enhancements:**

1. **Advanced Features**
   * Real-time chat support (planned for future)
   * Bulk operations for admins
   * Document versioning system
2. **Integration Limitations**
   * Limited third-party integrations
   * No payment gateway integration
   * No SMS notifications (email only)
3. **Scalability Considerations**
   * No load balancing implementation
   * Limited caching mechanisms
   * No microservices architecture
   * No advanced monitoring tools
4. **User Experience Enhancements**
   * No dark mode support
   * Limited customization options
   * No advanced search functionality
5. **Administrative Features**
   * No advanced user management tools
   * Limited backup and recovery options
   * No system configuration interface
   * No advanced audit trail features

**Future Development Roadmap:**

* Implementation of real-time notifications
* Advanced analytics dashboard
* Enhanced mobile app features
* Integration with additional AI services
* Advanced security features
* Performance optimization
* Enhanced user experience features

The system successfully meets all core requirements while providing a solid foundation for future enhancements and scalability improvements.

**CHAPTER 7: Conclusion, Recommendations and Challenges**

## **7.1 Conclusion**

The development of the UNIDOCS system represents a significant step toward digitizing and streamlining the process of official document generation at the State University of Zanzibar. By leveraging modern technologies such as Spring Boot, Angular, and PostgreSQL, the system ensures secure, efficient, and user-friendly access to essential academic documents. This project has successfully addressed the manual inefficiencies previously observed in the university's document processing system, including delays, data inconsistencies, and lack of transparency. The implementation of features such as automated letter generation, CV building, request tracking, and real-time notifications reflects a user-centered design that aligns with the institution's goals for digital transformation.

## **7.2 Recommendations**

To further enhance and sustain the impact of the UNIDOCS system, the following recommendations are proposed:

1. **Continuous User Training**: Regular workshops and training should be conducted for students and administrative staff to ensure effective use of the system.
2. **Integration with SUZA SIS and Email System**: Incorporating student information system (SIS) data and email notification integration will improve automation and reduce manual data entry.
3. **Data Backup and Recovery**: Implement a robust data backup and disaster recovery mechanism to prevent data loss and ensure system reliability.
4. **Security Enhancements**: Introduce multi-factor authentication (MFA) and routine security audits to protect user data and maintain privacy.
5. **Scalability Planning**: As student demand grows, the system should be tested and updated to handle higher loads without compromising performance.

## **7.3 Challenges**

During the system development and deployment, several challenges were encountered, including:

* **Requirement Gathering**: Accurately capturing all user needs and expectations took more time than anticipated due to varied stakeholder inputs.
* **Technical Constraints**: Ensuring compatibility between Angular and Spring Boot, as well as maintaining performance, required careful system architecture planning.
* **Time Constraints**: The project had to be delivered within academic deadlines, limiting the opportunity for extended testing and optimization.
* **User Resistance**: Some users were initially resistant to adopt the digital system due to familiarity with traditional paper-based methods.

Despite these challenges, the system was successfully implemented with core functionalities and received positive feedback during pilot testing.

# **References**

* Smith, J., & Brown, T. (2022). "Digital Transformation in University Administration." *Journal of Higher Education Systems*, 35(4), 145-160.
* Patel, M., & Jones, L. (2021). "Automated Document Management: Best Practices for Universities." *International Journal of AI and Data Science*, 18(3), 88-105.
* Kumar, R. (2023). "Enhancing Efficiency in Academic Document Processing." *Tech and Education Journal*, 27(2), 55-75.
* Postgres. (2022). "Postgress Documentation." Retrieved from https://postgres.com/doc/
* TensorFlow. (2023). "AI Chatbot Implementation Guide." Retrieved from <https://www.tensorflow.org/tutorials>
* Smith, J., Brown, T., & Lee, C. (2021). "Digital Document Management in Higher Education." *Journal of Information Systems*, 45(3), 112-126.
* Jones, R., & Patel, M. (2020). "AI Chatbots in University Administration: Enhancing Efficiency and Accessibility." *International Journal of AI Research*, 12(4), 88-102.
* Ellucian. (2022). "Banner Overview." Retrieved from https://www.ellucian.com
* M-Files. (2022). "Metadata-Driven Document Management." Retrieved from https://www.m-files.com
* EduSec. (2022). "EduSec Student Information System." Retrieved from [https://www.edusec.org](https://www.edusec.org/)
* Smith, J., Brown, T., & Lee, C. (2021). "Digital Document Management in Higher Education." *Journal of Information Systems*, 45(3), 112-126.
* Jones, R., & Patel, M. (2020). "AI Chatbots in University Administration: Enhancing Efficiency and Accessibility." *International Journal of AI Research*, 12(4), 88-102.
* Ellucian. (2022). "Banner Overview." Retrieved from [https://www.ellucian.com](https://www.ellucian.com/)
* M-Files. (2022). "Metadata-Driven Document Management." Retrieved from [https://www.m-files.com](https://www.m-files.com/)
* Brown, T., & Lee, C. (2022). "AI-Powered Chatbots in University Administration." Journal of Information Systems, 48(2), 200-215.
* Jones, R., & Patel, M. (2021). "Improving University Document Management Through Automation." International Journal of AI Research, 15(1), 102-118.
* AWS. (2023). "Cloud Computing and Scalability Solutions." Retrieved from [https://aws.amazon.com](https://aws.amazon.com/)
* GitHub. (2023). "Collaborative Software Development." Retrieved from [https://github.com](https://github.com/)