THE STATE UNIVERSITY OF ZANZIBAR SCHOOL OF BUSINESS (SOB)

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

FINAL PROJECT REPORT

PROJECT TITLE: UNIDOCS – DOCUMENTS 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 incorporates 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.

Acknowledgment

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

Table of Figures

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 whereby Students will be molding their Resumes for Career development. In other ways the System seamlessly 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 features 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. Absence 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 Resumes 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.

1.5 Objectives

This section outlines the main objectives of the UNIDOCS system. It highlights what the project aims to achieve, including the core goals related to improving efficiency, automating document generation, enhancing user experience, and ensuring secure access for both students and administrators.

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 Resume / 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.

1.6 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 does not violate legal laws and Regulations as it maintains privacy and Good conduct no immoral acts are portrayed on overall system processes.

Operational Feasibility: UNIDOCS addresses real administrative challenges faced by universities, making it highly practical and beneficial for users.

1.7. Organization of the Project

This section provides an overview of the structure of the UNIDOCS project report, offering readers a clear roadmap of what to expect in each chapter. It highlights the logical flow of the report, from the identification of the problem to the design, development, implementation, and evaluation of the UNIDOCS system.

Chapters Overview

Chapter One: Introduction

This chapter introduces the UNIDOCS project by outlining the background and motivation behind developing an integrated document management system for universities. It addresses the key challenges in existing manual processes, including inefficiencies, lack of centralization, and the absence of automation. The chapter defines the project's objectives, scope, and limitations, establishing the foundation for the solution proposed.

Chapter Two: Literature Review

This chapter explores existing literature and related systems such as DocuWare, M-Files, and EduSec. It analyzes their capabilities and limitations while reviewing academic studies on AI-powered document processing and digital transformation in higher education. The insights gathered help identify gaps in current solutions and justify the need for a system like UNIDOCS.

Chapter Three: Research Methodology

Chapter Three explains the methodologies and approaches used during the development of UNIDOCS. It describes the use of Object-Oriented Analysis (OOA) and a Bottom-Up development strategy within the Agile framework. Data collection techniques such as interviews and surveys are detailed, along with the tools, technologies, and modeling approaches used in system analysis.

Chapter Four: System Analysis

This chapter presents an in-depth analysis of the current manual document processes in universities. It outlines business rules, functional and non-functional requirements, and system behavior through modeling tools such as Use Case, Sequence, Class, and Entity-Relationship diagrams. This forms the analytical base for designing the system architecture.

Chapter Five: System Design

Focusing on system architecture, Chapter Five discusses the proposed three-tier design: User Tier, Application Tier, and Data Tier. It covers subsystem decomposition, database design, user interface principles, access control, and security strategies. The design ensures that the system is scalable, maintainable, and secure.

Chapter Six: System Implementation and Testing

This chapter documents how the system was implemented using technologies like Spring Boot, Angular, Flutter, and PostgreSQL. It covers database setup, backend and frontend development, and integration. It also explains the testing methodologies used — unit, integration, system, and user acceptance testing — and presents results, screenshots, and a strengths and limitations analysis of the system.

Chapter Seven: Conclusion, Recommendations, and Challenges

The UNIDOCS report is structured to reflect a clear and systematic development process, progressing logically from problem identification to system design, implementation, and evaluation. Each chapter builds on the previous one, ensuring that readers can easily follow the rationale behind the project, understand how the system was developed, and recognize areas for future growth and refinement.

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.
- **EduSec** – A student information system that includes document management features, though it lacks AI-driven automation and chatbot integration.

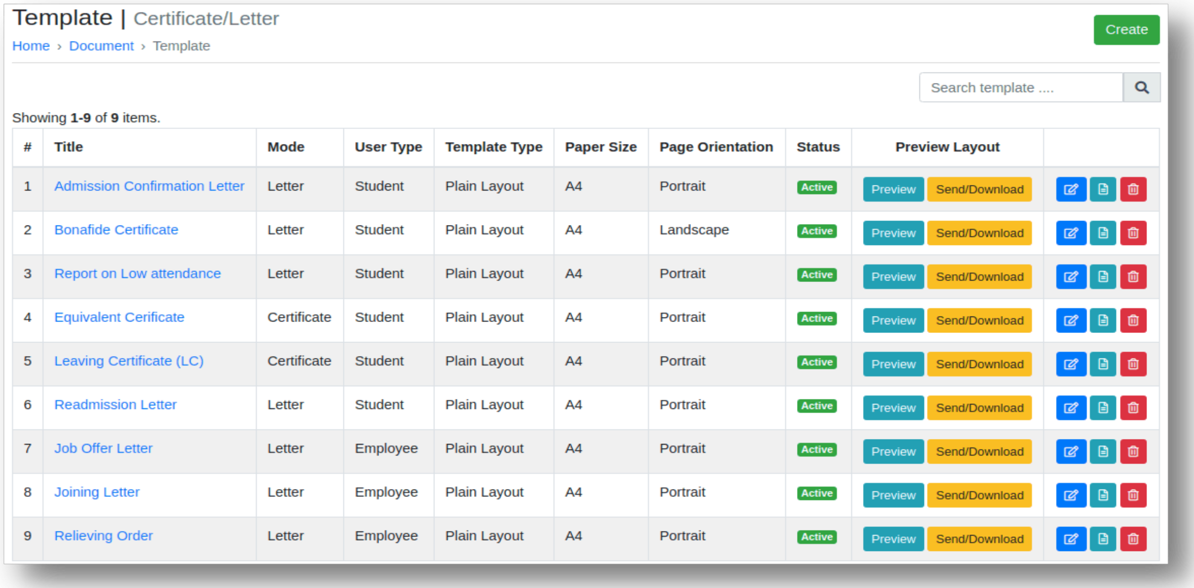
These platforms have improved document management but still leave gaps in accessibility, automation, and user interaction, which UNIDOCS aims to address.

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
- **Platform**: Web-based



Template | Certificate/Letter

Home › Document › Template

Showing **1-9** of **9** items.

#	Title	Mode	User Type	Template Type	Paper Size	Page Orientation	Status	Preview Layout	
1	Admission Confirmation Letter	Letter	Student	Plain Layout	A4	Portrait	Active	Preview Send/Download	
2	Bonafide Certificate	Letter	Student	Plain Layout	A4	Landscape	Active	Preview Send/Download	
3	Report on Low attendance	Letter	Student	Plain Layout	A4	Portrait	Active	Preview Send/Download	
4	Equivalent Cerificate	Certificate	Student	Plain Layout	A4	Portrait	Active	Preview Send/Download	
5	Leaving Certificate (LC)	Certificate	Student	Plain Layout	A4	Portrait	Active	Preview Send/Download	
6	Readmission Letter	Letter	Student	Plain Layout	A4	Portrait	Active	Preview Send/Download	
7	Job Offer Letter	Letter	Employee	Plain Layout	A4	Portrait	Active	Preview Send/Download	
8	Joining Letter	Letter	Employee	Plain Layout	A4	Portrait	Active	Preview Send/Download	
9	Relieving Order	Letter	Employee	Plain Layout	A4	Portrait	Active	Preview Send/Download	

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
- **Platform**: Web-based

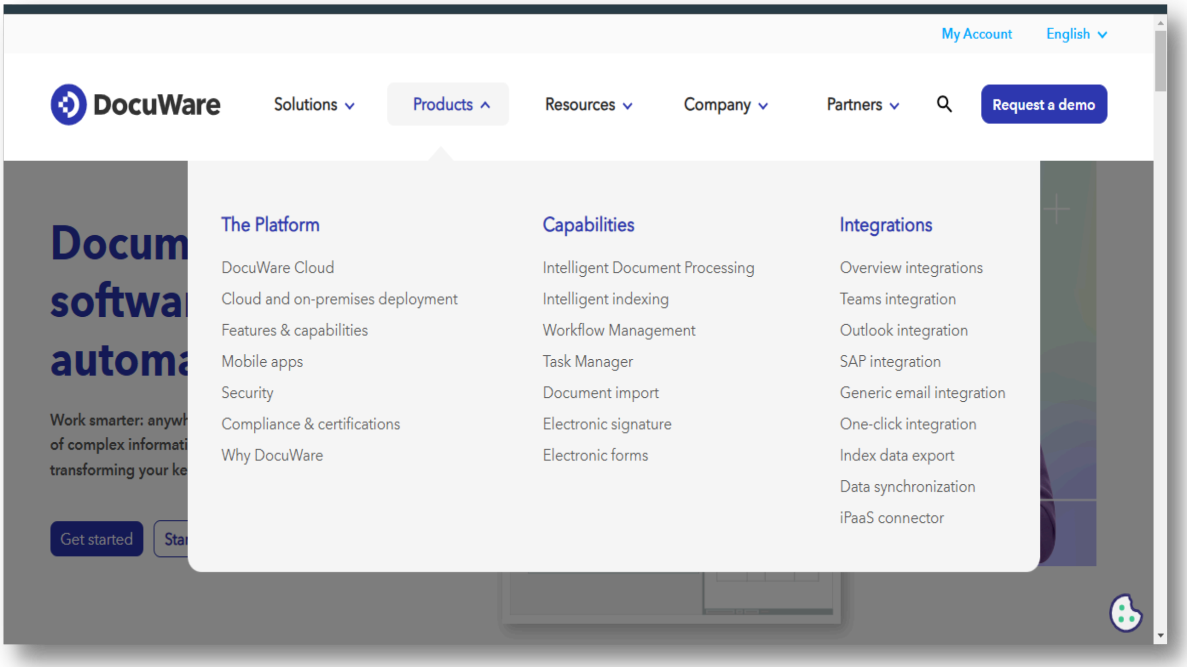
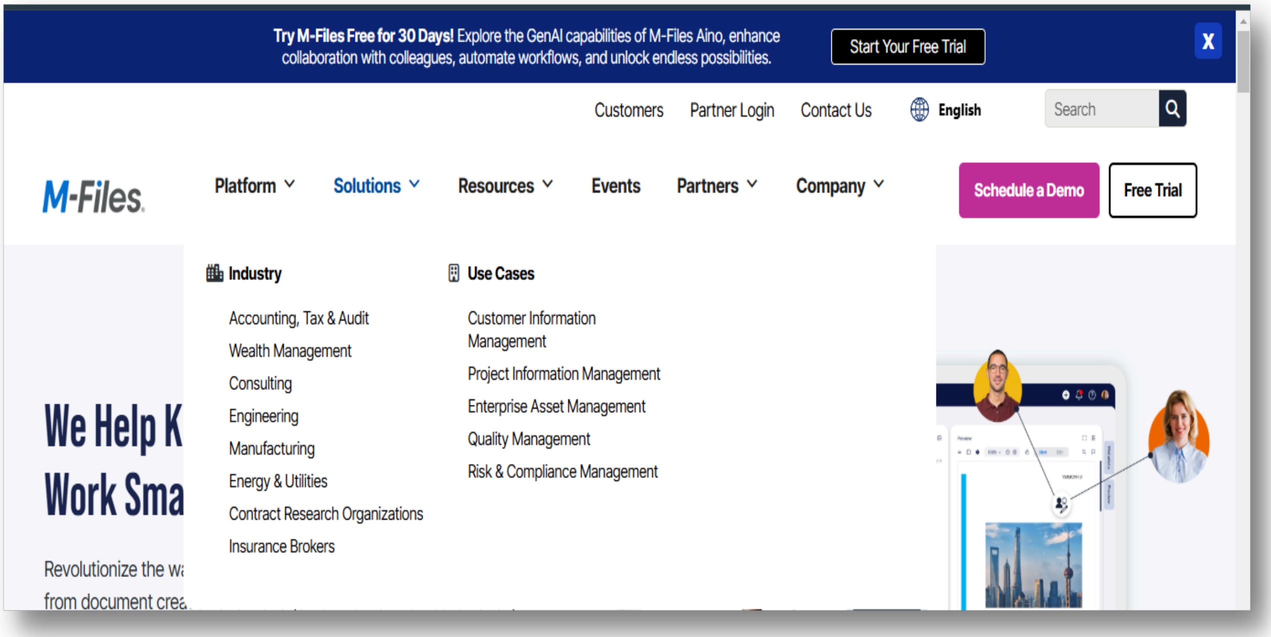


Figure 2: DocuWave

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
- **Platform**: Web and Mobile



Try M-Files Free for 30 Days! Explore the GenAI capabilities of M-Files Aino, enhance collaboration with colleagues, automate workflows, and unlock endless possibilities.

We Help K
Work Sma

Revolutionize the wa
from document crea

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.

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. **Scalability** – Many existing systems are rigid and do not easily adapt to growing user demands.

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.

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:

I. **Requirement Analysis**: Gathering functional and non-functional system requirements through surveys, interviews, and literature reviews.

II. **Design**: Developing system architecture diagrams and defining data flows to ensure a seamless structure.

III. **Implementation**: Developing the frontend using **Angular** and the backend with **Spring Boot**, ensuring system functionality as planned.

IV. **Testing**: Conducting **unit, integration, and system testing** to identify and resolve issues early in the development cycle.

V. **Deployment & Maintenance**: Deploying the system and providing ongoing updates to maintain performance, security, and scalability.

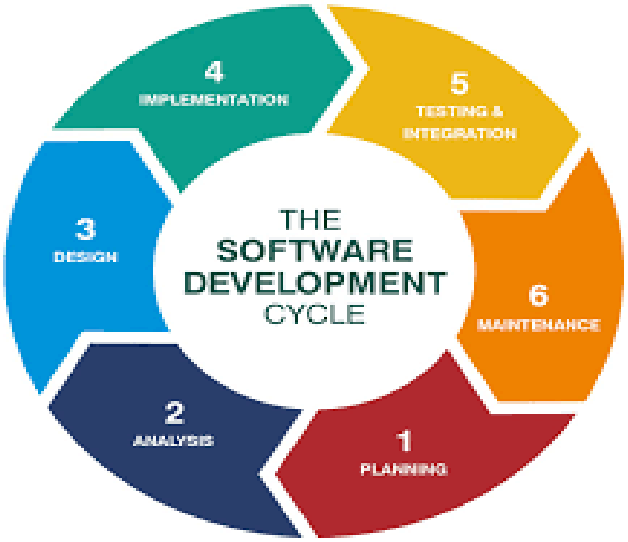


Figure 4: The Software Development Cycle

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 Interface before Actual Development.

Implementation:

I. Frontend Development: Angular and Flutter (For Mobile App)

II. Backend Development: Spring Boot, Java

III. Database: PostgreSQL

IV. 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:

I. **Frontend**: Angular.js and Flutter for creating dynamic and responsive user interfaces.

II. **Backend**: Spring Boot for developing scalable Restful APIs.

III. **Database**: Postgres for structured data management.

IV. **Version Control**: Git and GitHub for source code management.

V. **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:

I. **Interviews**: Conducted with university staff and students to understand document management challenges.

II. **Literature Review**: Examined best practices and previous research on document automation.

III. **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.
- **Re-usability**: 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

Key System Components

Core Entities:

The UNIDOCS comprises 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

Figure 5: Core Entities

Supporting Services:

The UNIDOCS System comprises Multiple supporting services to Ensure the streamlined Functionality of Pr-described Features. The Table Below Highlights 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

Figure 6: Supporting Services

Frontend Applications:

Angular Web Application: 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, often 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 Passwords for ensuring document authenticity and Confidentiality.
4. The System automatically sends Email to Users (Students) upon updates or Announcements
5. Admin will be capable of publishing 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.
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 minimum 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 Mobile 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 the Object-Oriented Analysis approach to ensure clear understanding of system requirements, functionality, and structure.

4.4. Dataflow Diagram

D FD 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.

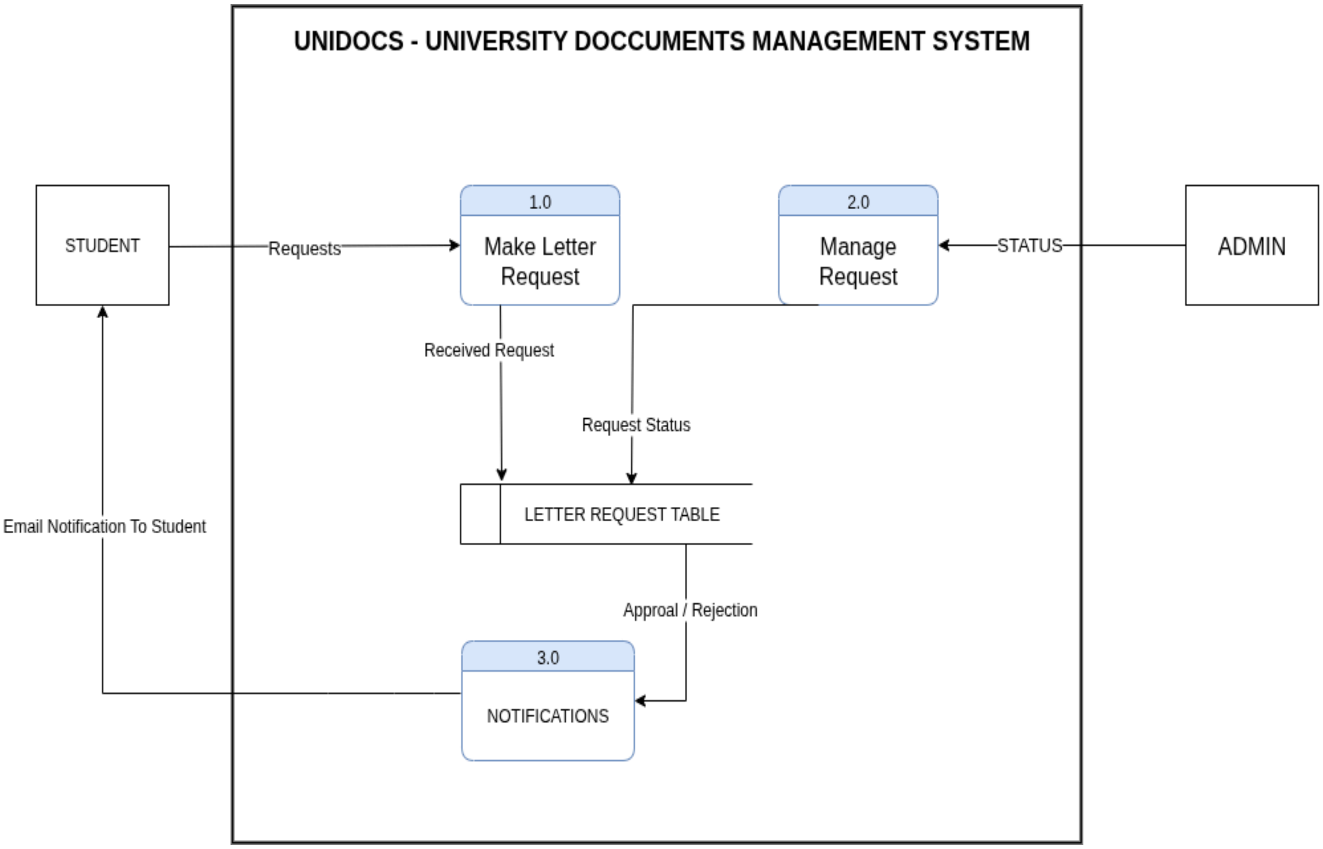


Figure 7: Data Flow Diagram

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

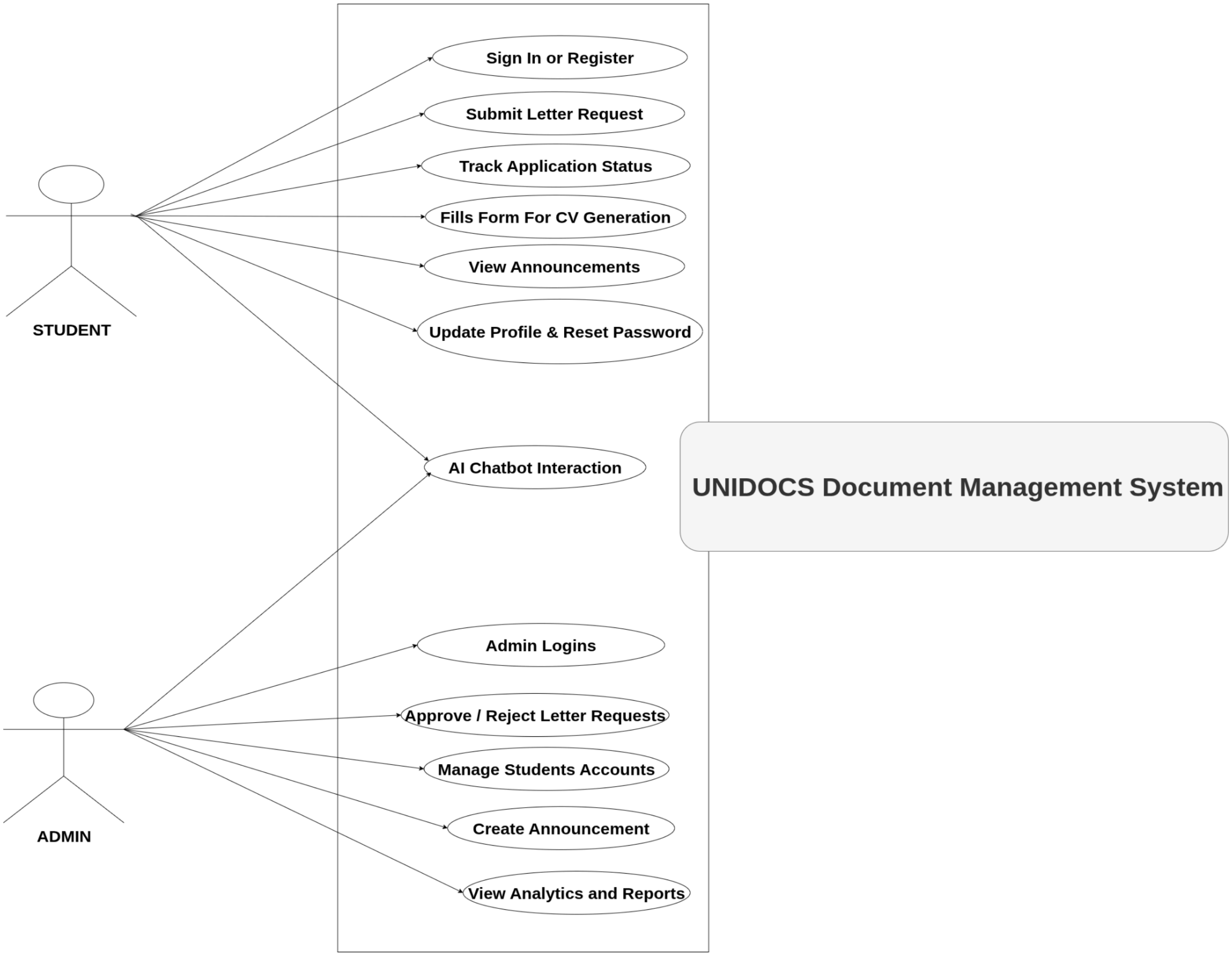


Figure 8: Use Case Diagram

4.5.2. Use Case Documentation

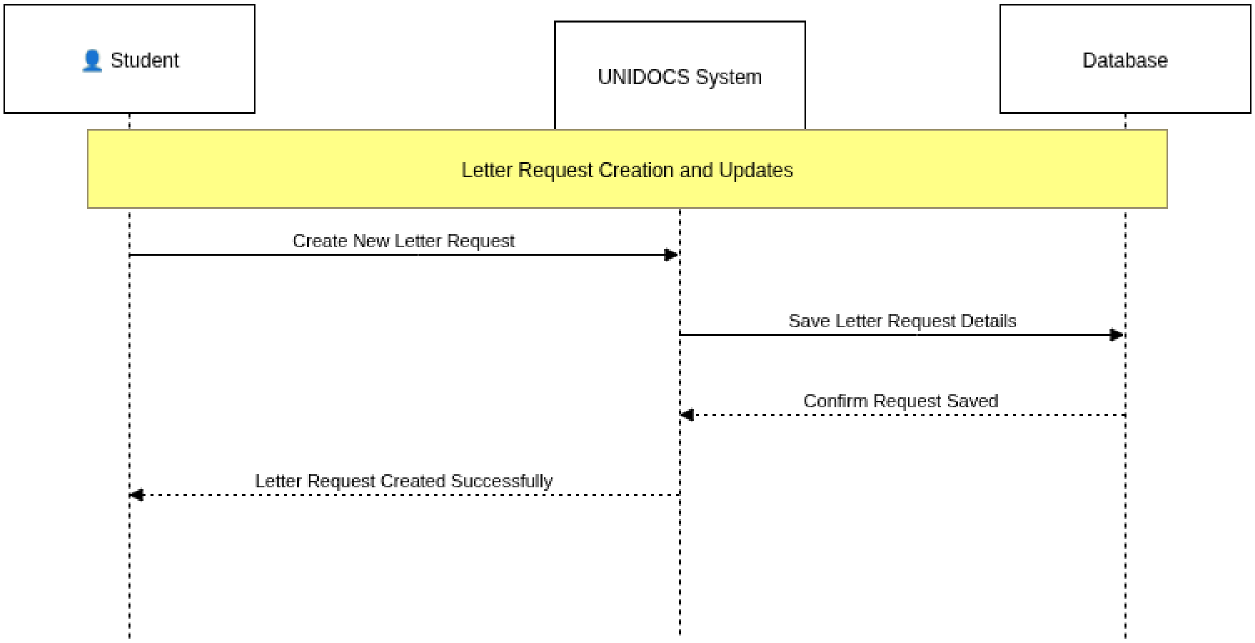
Students Use Cases

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


Student
UNIDOCS System
Database
Letter Request Creation and Updates
Create New Letter Request
Save Letter Request Details
Confirm Request Saved
Letter Request Created Successfully

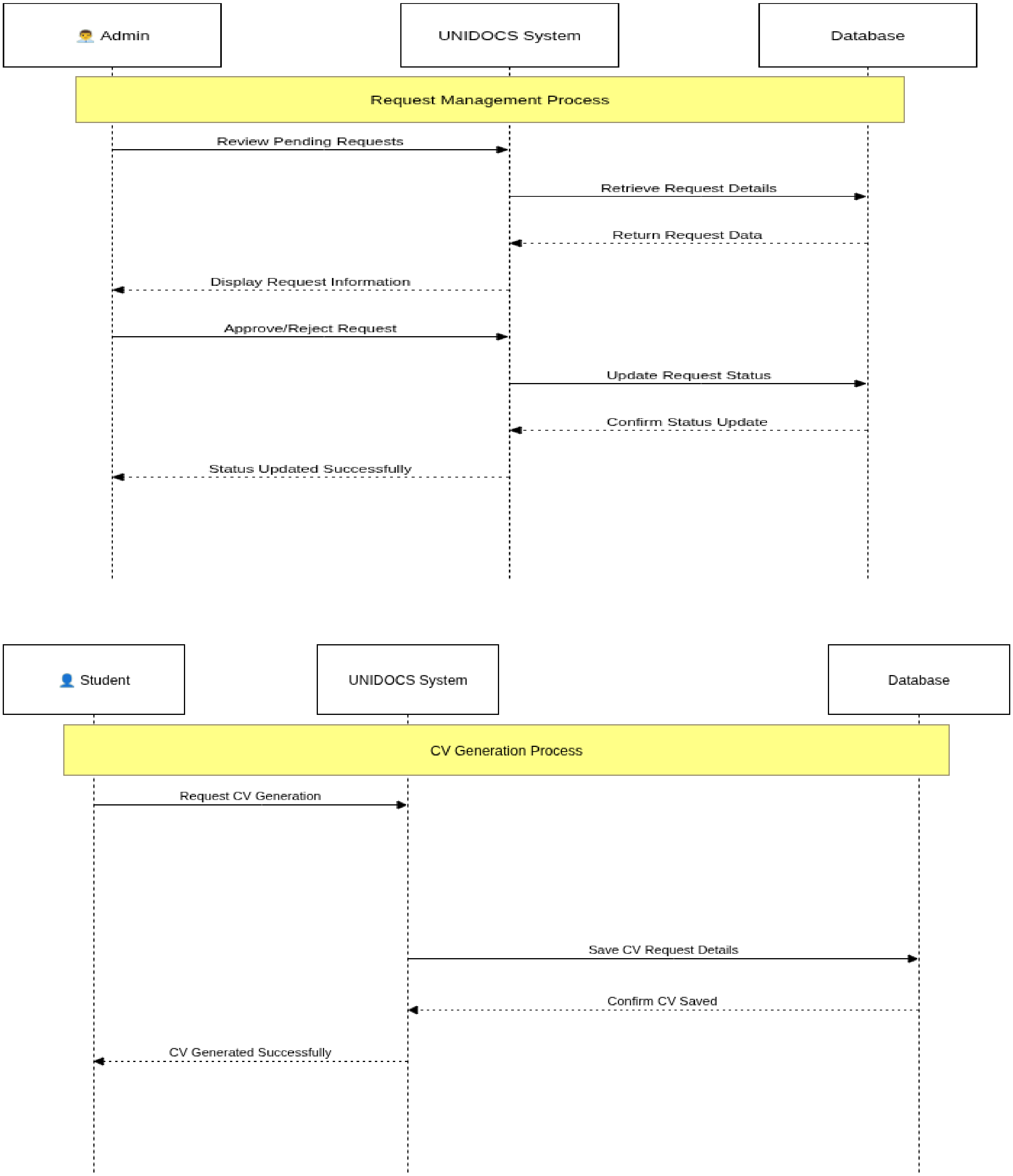


Figure 9: Sequence Diagram

4 .5.4. Conceptual modelling: Class diagram

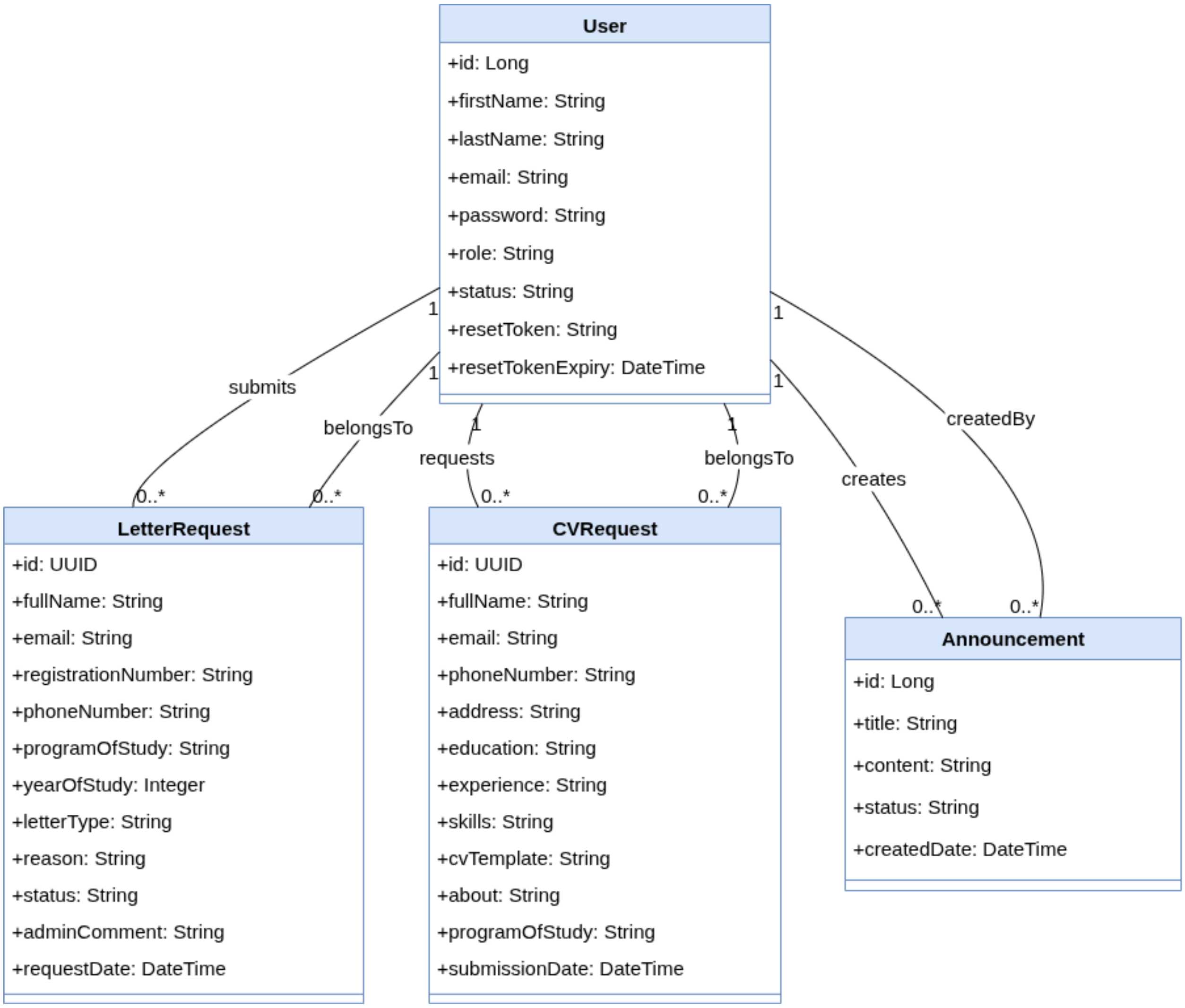


Figure 10: Class Diagram

4.5.5. Entity Relationship diagram

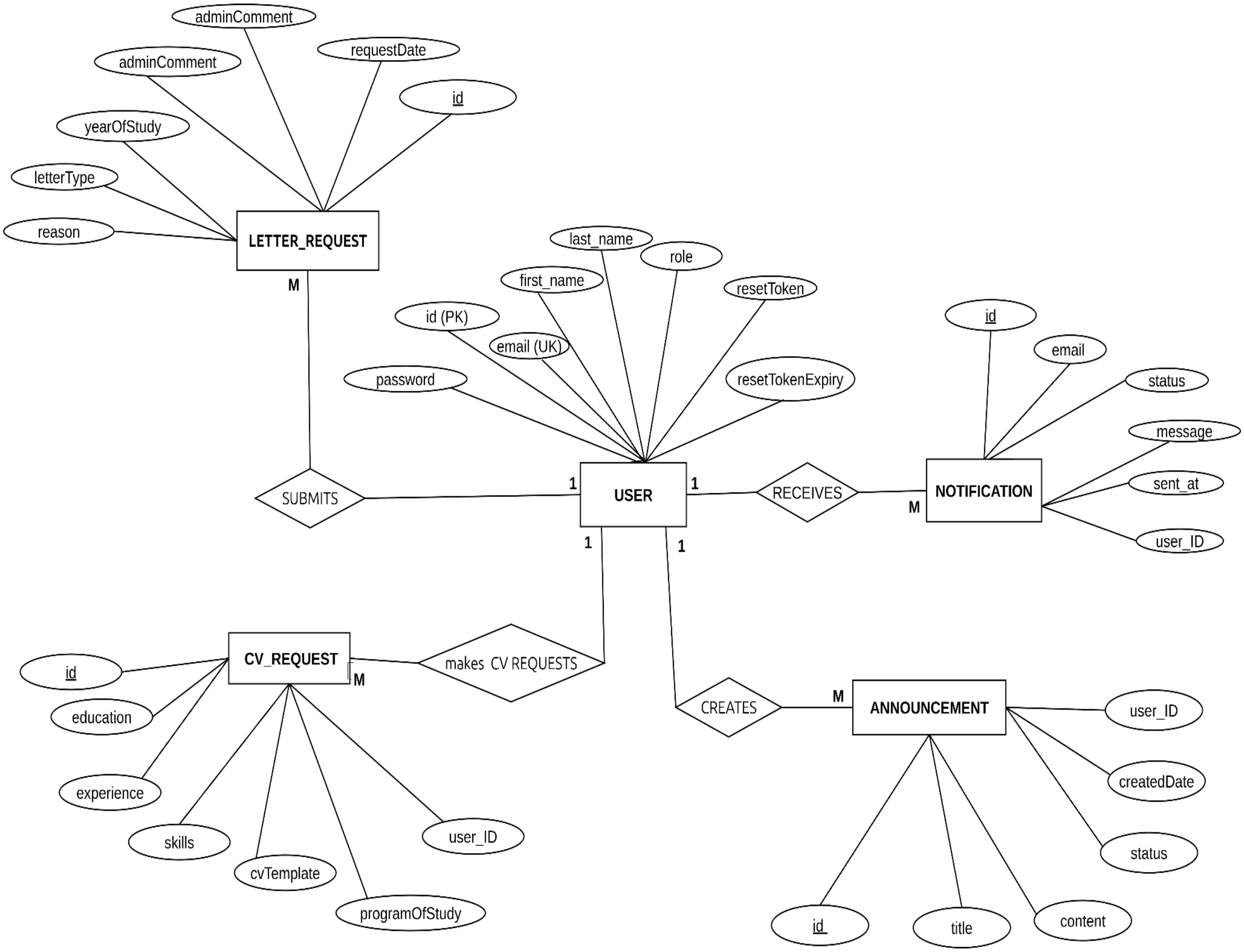


Figure 11: Entity Relationship (E-R) Diagram

CHAPTER 5: SYSTEM DESIGN

5.1. Architectural Design

Architectural Design is the process of defining the overall structure, organization, and high-level components of a software system. It involves making fundamental decisions about how the system will be structured, how components will interact with each other, and how the system will meet its functional and non-functional requirements.

5.1.1. Current Software Architecture

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

Multi-Tier Architecture:

- ➢ **Presentation Layer:** Angular (Web) & Flutter (Mobile)
- ➢ **API Layer:** REST Controllers handling HTTP requests
- ➢ **Business Logic Layer:** Service classes implementing business logic
- ➢ **Data Access Layer:** Repository classes for data operations
- ➢ **Data Layer:** PostgreSQL Database & External Services

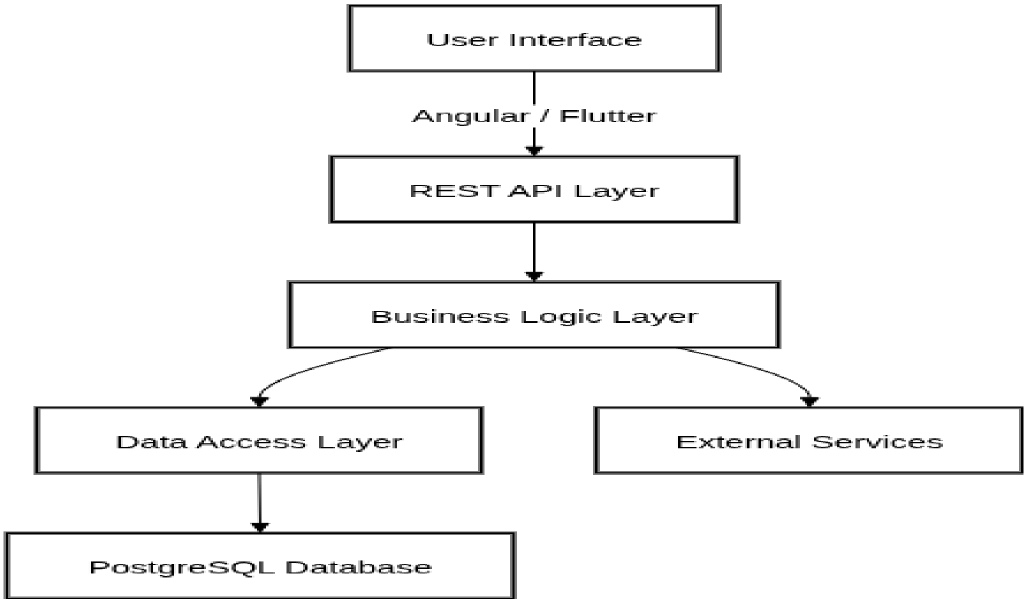


Figure 12: System Architecture Diagram

5.2. Proposed Software Architecture

The proposed architecture is based on microservices, enhancing scalability, maintainability, and security. Each subsystem is loosely coupled and independently deployable.

Subsystem Decomposition:

- ➢ **User Management Subsystem:** Handles authentication, authorization, and profile management
- ➢ **Document Management Subsystem:** Manages document workflows and CV generation
- ➢ **Communication Subsystem:** Sends notifications and real-time messages
- ➢ **Analytics Subsystem:** Monitors system usage and generates reports
- ➢ **AI Integration Subsystem:** Integrates PawaAI for CV generation and smart document processing

5.3 Database Design

This section outlines the design of the UNIDOCS database, detailing how data is structured, stored, and related. It includes the key entities, relationships, and design considerations used to ensure data integrity, consistency, and efficient access throughout the system.

5.3.1. Relational Model

The database design follows 3NF principles to ensure minimal redundancy and maximum data integrity.

Entities & Relationships:

- ➢ **USER** (1) → **LETTER_REQUEST** (M)
- ➢ **USER** (1) → **CV_REQUEST** (M)
- ➢ **USER** (1) → **ANNOUNCEMENT** (M)
- ➢ **USER** (1) → **NOTIFICATION** (M)

5.3.2. Data Description

- ➢ **USER:** Stores student/admin profile and login details
- ➢ **LETTER_REQUEST:** Captures all letter application data and approval tracking
- ➢ **CV_REQUEST:** Stores CV data including template and user details
- ➢ **ANNOUNCEMENT:** Manages system-wide communications
- ➢ **NOTIFICATION:** Logs all system-generated messages and alerts

5.3.3. Data Dictionaries

Table	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	First name
USER	last_name	VARCHAR(100)	NOT NULL	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 ID
LETTER_REQUEST	full_name	VARCHAR(255)	NOT NULL	Full name of student
LETTER_REQUEST	email	VARCHAR(255)	NOT NULL	Student email
LETTER_REQUEST	letter_type	VARCHAR(50)	NOT NULL	Type of letter
LETTER_REQUEST	status	VARCHAR(20)	NOT NULL	Request status
CV_REQUEST	id	UUID	PRIMARY KEY	Unique CV request ID

Table	Field Name	Data Type	Constraints	Description
CV_REQUEST	cv_template	VARCHAR(50)	NOT NULL	CV template name
CV_REQUEST	education	TEXT	NOT NULL	Education info
CV_REQUEST	experience	TEXT	NOT NULL	Work experience
ANNOUNCEMENT	id	BIGINT	PRIMARY KEY	Announcement ID
ANNOUNCEMENT	title	VARCHAR(255)	NOT NULL	Title
ANNOUNCEMENT	content	TEXT	NOT NULL	Content
NOTIFICATION	id	BIGINT	PRIMARY KEY	Notification ID
NOTIFICATION	type	VARCHAR(50)	NOT NULL	Notification type
NOTIFICATION	status	VARCHAR(20)	NOT NULL	Delivery status

Figure 13: Data Dictionary Table

5.4. User Interface Design

This section describes the design of the UNIDOCS user interface, focusing on how users interact with the system. It outlines the structure of various forms and reports, as well as the design principles used to ensure the interface is responsive, user-friendly, and consistent across different devices and user roles.

5.4.1. Forms and Reports

Student Forms:
The UNIDOCS system provides a set of intuitive and functional forms designed specifically for student interactions. The **Registration Form** allows users to sign up and log in to the UNIDOCS portal, forming the entry point to the system. The **Letter Request Form** is a structured interface for students to submit detailed requests for various official letters. To assist students in their career development, the **CV Generation Form** offers a highly customized layout for easy creation of professional curriculum vitae. Additionally, the **Profile Update Form** enables students to conveniently update their personal information using a responsive interface, especially useful in correcting any errors during registration.

Admin Forms:
The **Request Management Interface** displays all submitted letter requests, allowing efficient review and approval workflows. The **Announcement Editor** serves as a content management layout for publishing institutional announcements. Managing user accounts is streamlined through the **User Management Dashboard**, which allows for actions such as suspending or deleting student accounts. Finally, the **Analytics Dashboard** provides visual representations and reports of system-wide data, helping administrators monitor performance and usage patterns.

Reports:

The system includes essential reporting capabilities to enhance transparency and accountability. The **Letter Request Status** report provides real-time updates on the progress and status of each request submitted by students. In addition, the **Document Generation Report** presents statistics and logs of all documents generated within the system, supporting auditing and system evaluation processes.

5.4.2. Interface Design Principles

The design of the UNIDOCS user interface adheres to several key principles to ensure optimal usability. **Responsive Design** is implemented to maintain a consistent and accessible layout across all devices, including desktops, tablets, and mobile phones. **User-Friendly Navigation** simplifies the process of moving between different pages or views, reducing cognitive load and enhancing user satisfaction. **Consistent Styling** ensures visual uniformity throughout the platform, reinforcing brand identity and system cohesion. Furthermore, **Interactive Feedback**, such as hover effects, loading indicators, and confirmation prompts, is embedded throughout the interface to improve engagement and system transparency during user interactions.

5.5. Access Control and Security

This section explains how access to the UNIDOCS system is controlled and secured. It covers the role-based access permissions for different users, authentication mechanisms, data protection strategies, and security best practices implemented to safeguard user information and system resources.

5.5.1. Role-Based Access Control (RBAC)

Student Role:

Students are required to **register, log in, or reset their passwords** before accessing system services. Once authenticated, they can **submit and track letter and CV requests**, enabling them to engage with institutional processes efficiently. If errors occur during registration, students can **update their profiles** through the profile management interface. Additionally, students have access to **view announcements and personal analytics**, allowing them to stay informed and review their system activity.

Admin Role:
Administrators have elevated privileges to manage the overall system. They can **manage user accounts**, including suspending or deleting students as necessary. Admins are also responsible for **approving or rejecting requests**, ensuring that only valid and appropriate letters are issued. The ability to **create announcements** enables them to disseminate official communication effectively. Lastly, they can **view reports and analytics** to monitor system performance, user activities, and request volumes.

5.5.2. Security Measures

Authentication:
Security in UNIDOCS begins with **JWT-based token authentication**, which secures sessions and enables authorized access. Passwords are encrypted using **BCrypt**, providing strong protection for user credentials. Additionally, **secure session timeout management** is implemented to automatically log out inactive users, minimizing the risk of unauthorized access.

Data Security:
UNIDOCS applies **encryption both at rest and in transit** to ensure the confidentiality of sensitive data throughout the system. The use of **parameterized queries** effectively prevents SQL injection attacks, securing the backend database against unauthorized manipulations.

Access Control:
The platform reinforces its security framework by implementing **REST endpoint protection** and **method-level authorization**, ensuring that each function is accessible only to users with the appropriate roles. Furthermore, **brute force rate limiting** is applied to prevent automated attacks on the login system.

5.5.3. Security Implementation Best Practices

To safeguard the system against common vulnerabilities, UNIDOCS integrates several best practices in its implementation. **Output encoding** is used to prevent **Cross-Site Scripting (XSS)** attacks by neutralizing potentially harmful user input. Additionally, **secure error messaging** ensures that system error outputs do not expose sensitive internal information, thereby minimizing the risk of exploitation.

This comprehensive design ensures that UNIDOCS remains scalable, secure, and user-friendly. It supports both student and administrator roles effectively while aligning with modern software engineering standards and security best practices.

CHAPTER 6: SYSTEM IMPLEMENTATION AND TESTING

6.1. Technologies

The UNIDOCS system is built using a robust and modern technology stack, carefully selected to support scalability, maintainability, and high performance. It integrates various tools and frameworks across the backend, frontend, mobile development, and external services.

6.1.1. Backend Technologies

The backend of UNIDOCS is developed using **Java 17**, a reliable and high-performance programming language suited for enterprise applications. The application is structured using **Spring Boot 3.3.1**, which simplifies the development of production-ready services. For secure access control, **Spring Security** is used to manage authentication and authorization. Data persistence is handled via **Spring Data JPA**, which works seamlessly with the **Hibernate ORM** for database interactions. The primary database is **PostgreSQL**, a powerful open-source relational database system. For stateless user authentication, **JWT (JSON Web Tokens)** is employed to ensure secure and scalable user sessions. Additionally, **Maven** is used for build automation, project structure management, and dependency control.

6.1.2. Frontend Technologies

On the client side, UNIDOCS utilizes **Angular 16**, a component-based framework ideal for building dynamic and responsive web applications. The logic is written in **TypeScript**, a superset of JavaScript that offers static typing for enhanced developer productivity and error checking. **HTML5** is used for structuring the web pages, while **Bootstrap** is incorporated for styling and ensuring that the design is responsive across various screen sizes and devices.

6.1.3. Mobile Technologies

For mobile platform support, the system uses **Flutter 3.x**, a modern and efficient cross-platform framework developed by Google. It enables UNIDOCS to offer a consistent user experience on both Android and iOS devices. The mobile application's logic is written in **Dart**, the native language for Flutter. The user interface design follows **Material Design** guidelines, ensuring a clean, intuitive, and familiar mobile experience.

6.1.4. External Services

UNIDOCS also integrates external services to enhance functionality. The **PawaAI API** is used for AI-powered document generation, automating the creation of various student documents. An **SMTP server** is connected to the system to enable email notifications for users regarding account activities and document processing. **Docker** is employed to containerize the application, ensuring smooth deployment across different environments. Lastly, **Git** serves as the version control system, facilitating collaborative development, source code tracking, and rollback capabilities.

6.2. Database Implementation

This section describes how the UNIDOCS database was designed and implemented to support data storage, retrieval, and integrity. It outlines the structure, tools, and techniques used to manage relational data and ensure efficient interaction between the application and the database

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	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	First name
USER	last_name	VARCHAR(100)	NOT NULL	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 ID
LETTER_REQUEST	full_name	VARCHAR(255)	NOT NULL	Full name of student
LETTER_REQUEST	email	VARCHAR(255)	NOT NULL	Student email
LETTER_REQUEST	letter_type	VARCHAR(50)	NOT NULL	Type of letter
LETTER_REQUEST	status	VARCHAR(20)	NOT NULL	Request status
CV_REQUEST	id	UUID	PRIMARY KEY	Unique CV request ID
CV_REQUEST	cv_template	VARCHAR(50)	NOT NULL	CV template name
CV_REQUEST	education	TEXT	NOT NULL	Education info
CV_REQUEST	experience	TEXT	NOT NULL	Work experience
ANNOUNCEMENT	id	BIGINT	PRIMARY KEY	Announcement ID
ANNOUNCEMENT	title	VARCHAR(255)	NOT NULL	Title
ANNOUNCEMENT	content	TEXT	NOT NULL	Content
NOTIFICATION	id	BIGINT	PRIMARY KEY	Notification ID
NOTIFICATION	type	VARCHAR(50)	NOT NULL	Notification type
NOTIFICATION	status	VARCHAR(20)	NOT NULL	Delivery status

Figure 14: Data Dictionary Table

6.3. Testing

Testing is a crucial phase in software development aimed at evaluating a system to identify defects, verify functionality, and ensure that the software performs according to its specified requirements. It helps guarantee that the final product is reliable, secure, and ready for deployment by uncovering bugs, usability issues, and integration problems before the system goes live.

6.3.1. System Testing Methodology

The UNIDOCS system underwent comprehensive testing using a layered approach to verify its performance, functionality, and user experience. This process included **Unit Testing**, where individual components of the application were tested in isolation to confirm that each function or module performed correctly on its own. This helped detect and correct errors early in the development process.

Next, **Integration Testing** was conducted to ensure that different modules and components interacted smoothly. For instance, the connection between the frontend and backend, the database interactions, and the integration with external services like email and document generation were all verified for seamless communication.

After verifying integration points, **System Testing** was performed to evaluate the behavior of the entire system in a complete and production-like environment. This stage ensured that all functional requirements were met and that the application delivered the intended outcomes under normal conditions.

Finally, **User Acceptance Testing (UAT)** involved testing the system with real users, such as students and administrators, to validate its usability and effectiveness in real-world scenarios. This helped confirm that the system was not only functional but also user-friendly and aligned with user needs.

6.3.2. Testing Results and Answers

Does each functionality of the system produce an expected output?

YES - All core functionalities produce expected outputs:

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

Does the system respond with invalid data entry?

YES - The system implements comprehensive input validation:

1. **Email Validation**: Rejects invalid email formats
2. **Required Field Validation**: Prevents submission with missing mandatory fields
3. **Data Type Validation**: Ensures correct data types for all inputs
4. **Length Validation**: Enforces character limits on text fields
5. **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:

1. **Invalid Email**: System displays "User not found" message
2. **Incorrect Password**: System shows "Invalid credentials" error
3. **Inactive Account**: System prevents login and shows account status
4. **Multiple Failed Attempts**: System implements rate limiting
5. **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:

1. Cannot access admin dashboard
2. Cannot approve/reject requests
3. Cannot create announcements
4. Cannot view other users' data
5. Cannot access system analytics

Admin Privileges:

1. Full access to request management
2. Ability to create and manage announcements
3. Access to user management features
4. System analytics and reporting access
5. 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 approach:

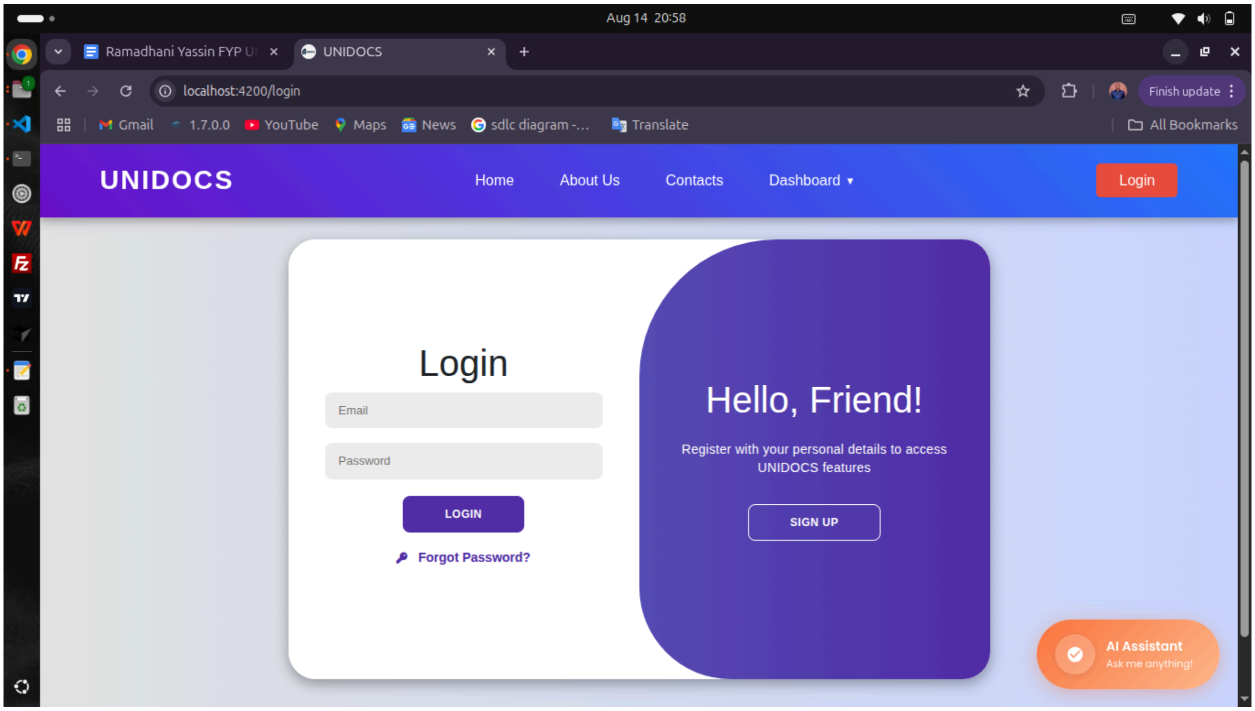
Object-Oriented Implementation:

1. Clear class hierarchies and relationships
2. Proper encapsulation of data and methods
3. Inheritance used for user roles
4. Polymorphism in service implementations

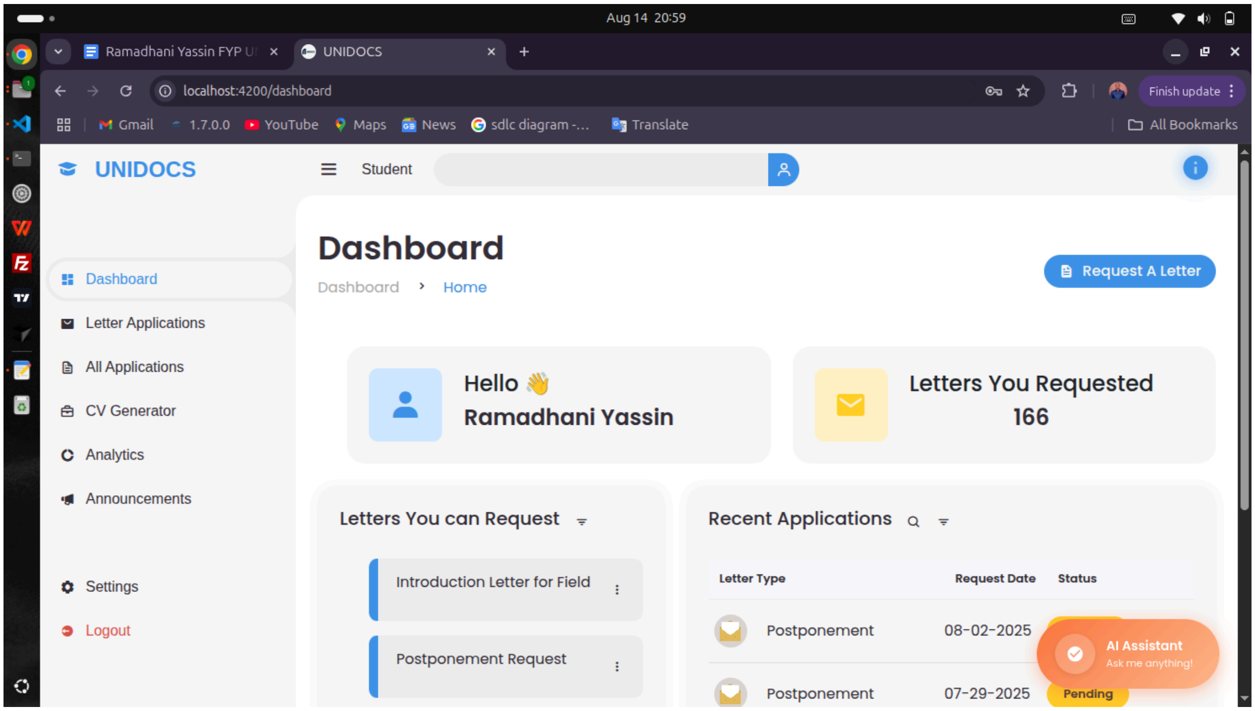
Technology Stack Compliance:

1. Spring Boot backend as proposed
2. Angular web frontend as planned
3. Flutter mobile app as designed
4. PostgreSQL database as specified
5. AI integration with PawaAI as intended

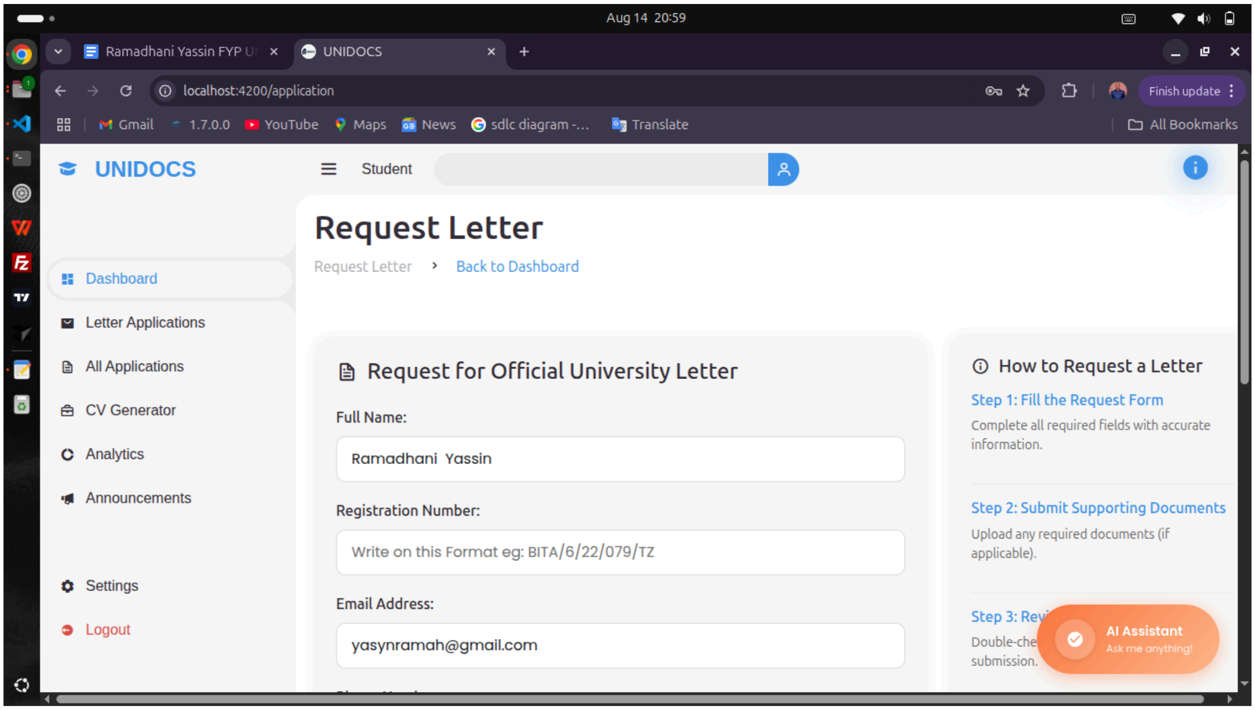
6.4. User Interfaces



Register with your personal details to access UNIDOCS features



Letter Type	Request Date	Status
Postponement	08-02-2025	
Postponement	07-29-2025	Pending



Request Letter

Request Letter › Back to Dashboard

Request for Official University Letter

Full Name: Ramadhani Yassin

Registration Number: Write on this Format eg: BITA/6/22/079/TZ

Email Address: yasynramah@gmail.com

How to Request a Letter

Step 1: Fill the Request Form

Complete all required fields with accurate information.

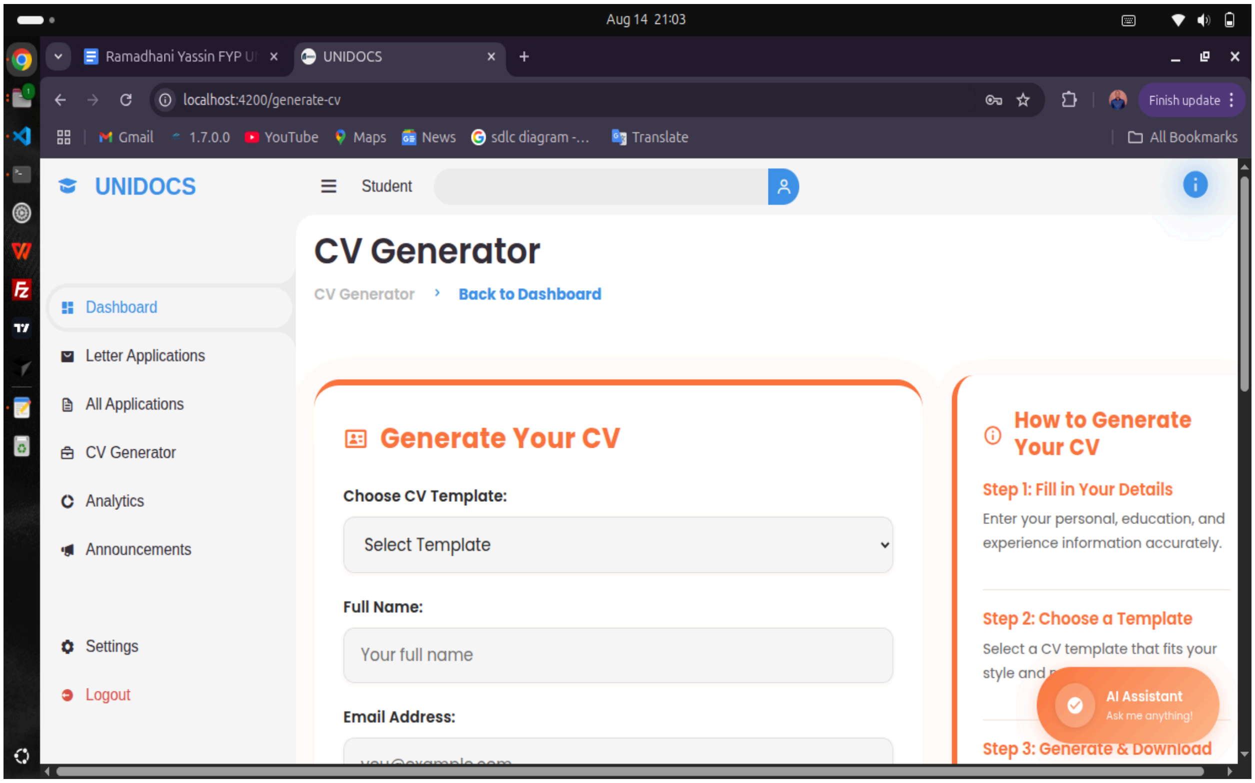
Step 2: Submit Supporting Documents

Upload any required documents (if applicable).

Step 3: Revi

Double-che

submission.



CV Generator

CV Generator › Back to Dashboard

Generate Your CV

Choose CV Template: Select Template

Full Name: Your full name

Email Address: you@example.com

How to Generate Your CV

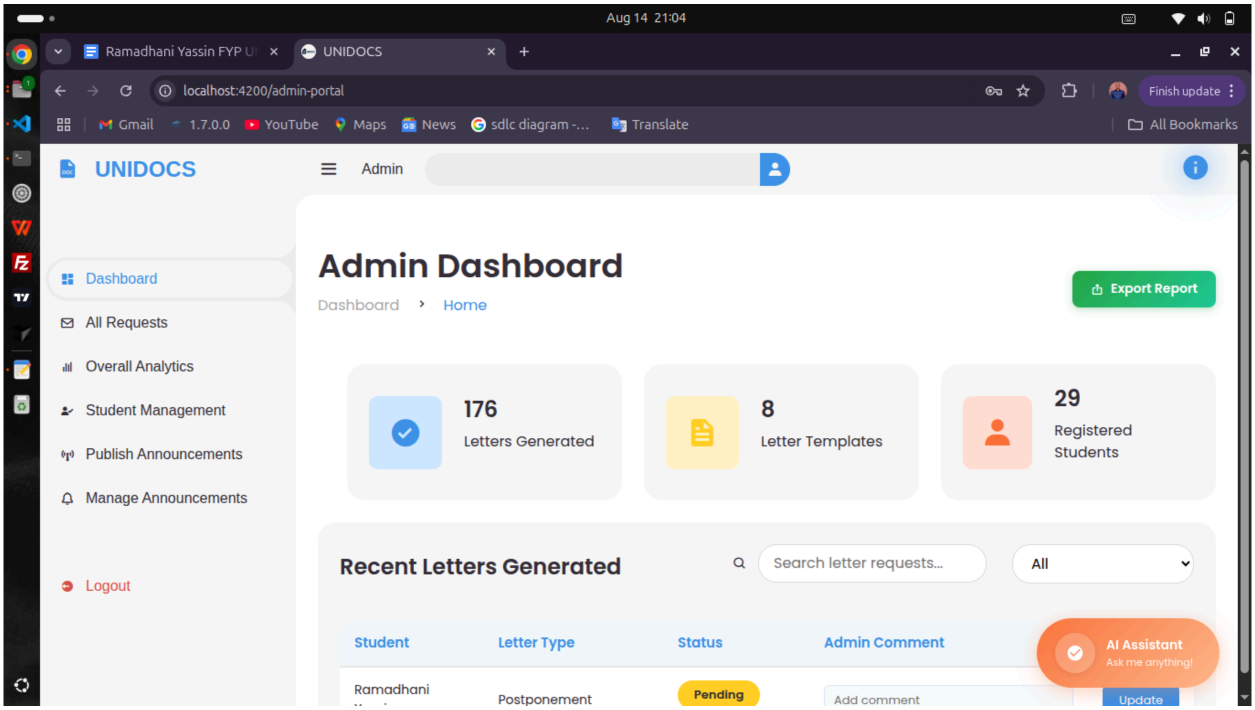
Step 1: Fill in Your Details

Enter your personal, education, and experience information accurately.

Step 2: Choose a Template

Select a CV template that fits your style and

Step 3: Generate & Download



Admin Dashboard

Dashboard › Home

176 Letters Generated

8 Letter Templates

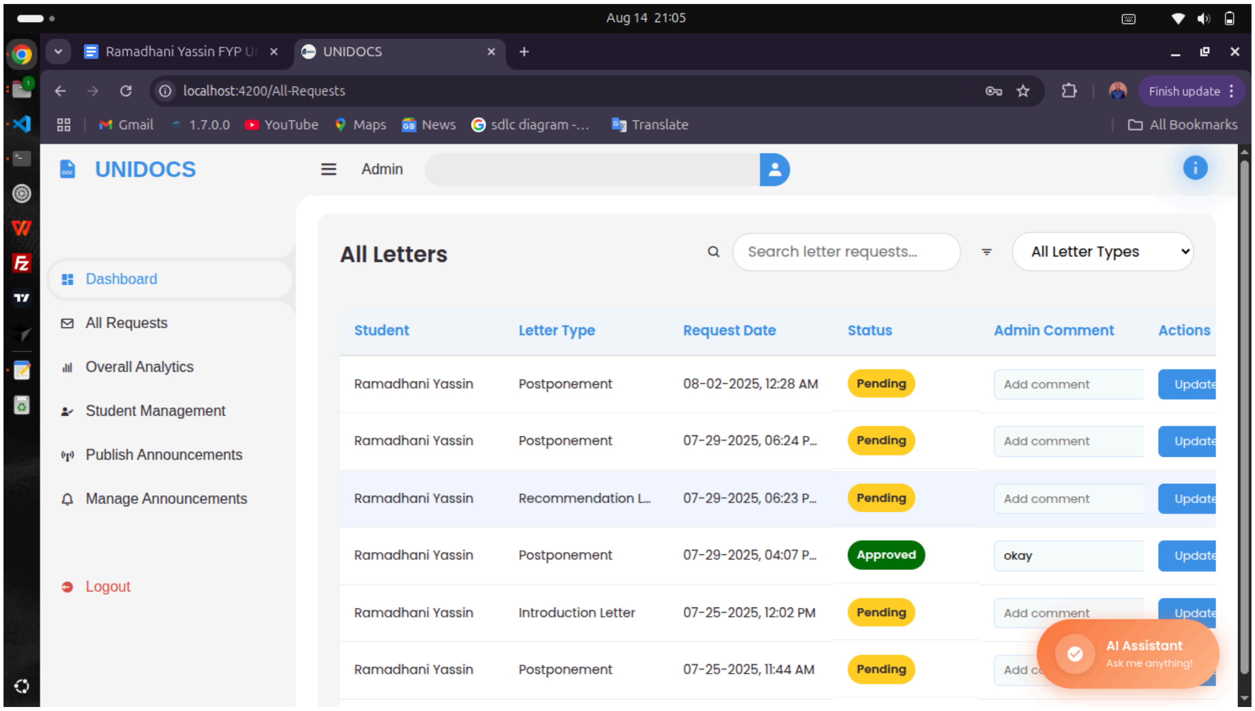
29 Registered Students

Recent Letters Generated

Search letter requests...

All

Student	Letter Type	Status	Admin Comment
Ramadhani ...	Postponement	Pending	Add comment

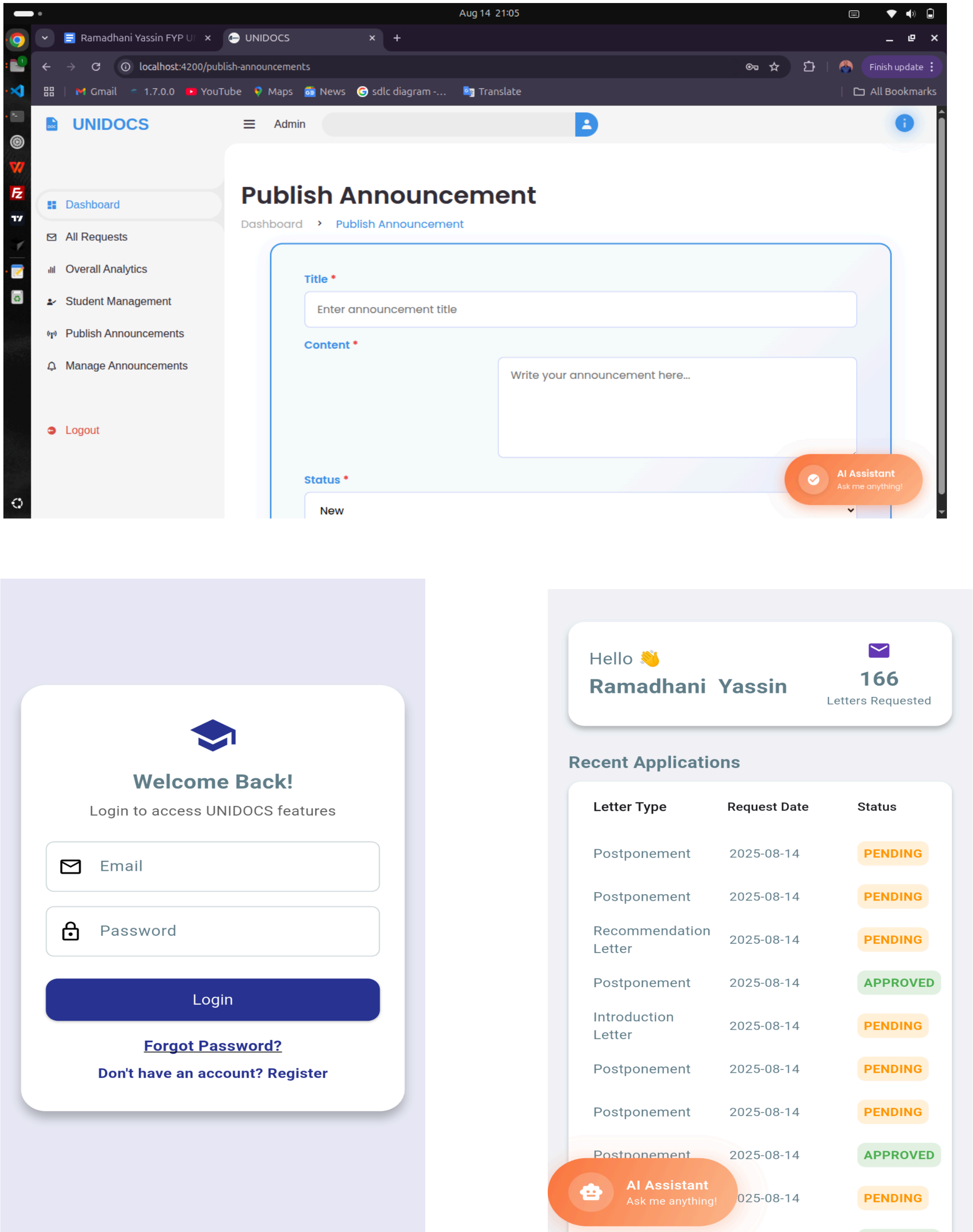


All Letters

Search letter requests...

All Letter Types

Student	Letter Type	Request Date	Status	Admin Comment	Actions
Ramadhani Yassin	Postponement	08-02-2025, 12:28 AM	Pending	Add comment	Update
Ramadhani Yassin	Postponement	07-29-2025, 06:24 P...	Pending	Add comment	Update
Ramadhani Yassin	Recommendation L...	07-29-2025, 06:23 P...	Pending	Add comment	Update
Ramadhani Yassin	Postponement	07-29-2025, 04:07 P...	Approved	okay	Update
Ramadhani Yassin	Introduction Letter	07-25-2025, 12:02 PM	Pending	Add comment	Update
Ramadhani Yassin	Postponement	07-25-2025, 11:44 AM	Pending	Add c	



Publish Announcement

Dashboard › Publish Announcement

Title *

Enter announcement title

Content *

Write your announcement here...

Status *

New

Welcome Back!

Login to access UNIDOCS features

Email

Password

Login

Forgot Password?

Don't have an account? Register

Hello 👋

Ramadhani Yassin

166

Letters Requested

Recent Applications

Letter Type	Request Date	Status
Postponement	2025-08-14	PENDING
Postponement	2025-08-14	PENDING
Recommendation Letter	2025-08-14	PENDING
Postponement	2025-08-14	APPROVED
Introduction Letter	2025-08-14	PENDING
Postponement	2025-08-14	PENDING
Postponement	2025-08-14	PENDING
Postponement	2025-08-14	APPROVED
	025-08-14	PENDING

Figure 15: UseInterfaces

6.5. Strengths and Limitations of the System

This section highlights the key strengths of the UNIDOCS system, including features that were successfully implemented based on the project requirements. It also outlines the current limitations of the system and areas identified for future improvement and enhancement.

6.5.1. What is Covered from Requirements

The UNIDOCS system successfully implements the core requirements as outlined in the project scope, with several fully functional modules that enhance both administrative and student interactions. The **User Management System** allows for complete user registration and authentication, supports role-based access control for students and administrators, includes password reset functionality, and provides profile management capabilities for users to update their personal data.

In terms of **Letter Request Management**, the system offers a comprehensive submission process for various types of official letters. It supports multiple letter formats, includes tracking and status update features, and incorporates an approval or rejection workflow for administrators. The **CV Generation System** is integrated with PawaAI to deliver AI-powered resume generation, offering multiple templates and customizable content fields, along with downloadable document outputs.

The **Announcement System** allows administrators to create announcements with rich text content, manage their status, and enable students to view updates from within the platform. A **Communication System** is also in place, which sends automated email notifications for major events, tracks notification history, and provides users with real-time status updates.

The system also includes several **Security Features**, such as JWT-based authentication for secure user sessions, thorough input validation and sanitization to prevent common vulnerabilities, role-based access control to limit system access appropriately, and audit logging to monitor user actions. Moreover, UNIDOCS supports **Multi-Platform Access**, offering both a responsive web application developed with Angular and a mobile version built using Flutter to ensure accessibility across devices.

6.5.2. What is Not Covered

Despite its robust foundation, the current version of UNIDOCS has certain limitations and areas marked for future enhancement. In terms of **Advanced Features**, the system does not yet support real-time chat for user support, bulk operations for administrators, or a document versioning system to track revisions over time.

There are also **Integration Limitations**. The system currently supports limited third-party integrations and lacks support for payment gateways or SMS-based notifications, relying solely on email for communication. These integrations could enhance user experience and administrative efficiency if implemented in the future.

Regarding **Scalability Considerations**, the system has not yet implemented load balancing for high traffic management, caching mechanisms for optimized data retrieval, microservices architecture for modular scalability, or advanced monitoring tools to track system performance in real-time.

The **User Experience** could also benefit from enhancements such as a dark mode option, more customization settings for individual preferences, and advanced search capabilities to help users find content more efficiently. On the **Administrative Features** side, there is a lack of advanced user management tools, system configuration interfaces, and comprehensive backup and recovery options. The audit trail features are basic and could be expanded for better traceability.

Looking ahead, the **Future Development Roadmap** includes plans for implementing real-time notifications, developing an advanced analytics dashboard, enhancing mobile app functionality, integrating additional AI-powered services, strengthening security mechanisms, optimizing overall system performance, and improving user experience through various interface upgrades.

Overall, while the system meets all of its core functional requirements and delivers a stable and secure foundation, there remains significant potential for expansion and improvement in future development cycles.

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
- 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
- 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
- GitHub. (2023). "Collaborative Software Development." Retrieved from https://github.com