Draft Version: 0.0

MAKERERE UNIVERSITY

ACADEMIC ISSSUE TRACKING SYSTEM (AITS):

Case Study: MAKERERE UNIVERSITY

By:

GROUP_S

COLLEGE OF COMPUTING AND INFORMATION SCIENCES

SCHOOL OF COMPUTING AND INFORMATICS TECHNOLY

DEPARTMENT OF COMPUTER SCIENCE

BACHELOR OF SCIENCE IN COMPUTER SCIENCE (Year One)

A project proposal in partial fulfilment of the requirements for the award of competence in **software development project** (CSC:1202) for **semester II** (Year One) - Academic Year

2024:

January 2025

PROJECT TEAM

Name	REG.NO	Role/Responsibility
1. Seanice Nabasirye (0770862549)	24/U/22871	Front-End Developer
2. Isaac Asiimwe (0770563394)	24/U/03765/PS	Front-End Developer/Project leader
3. Solomon Jessy Kiyingi (0767594340)	24/U/25920/EVE	Back-End Development
4. Jonan Akandwanaho (0756348528)	24/U/03053/PS	Front-End Development
5. Keith Paul Kato (0759021371)	24/U/26593/EVE	Documentation/Backend
6.		

PROJECT SCHEDULE

Week One: (Agenda):	
Project Overview:	❖ Scope ❖ Objectives ❖ Key Deliverables ❖ Programming Tasks: (Git/GitHub) Setup & Repositories ❖ Project Structure (Using Django and React
	❖

Document Change Control.

Version	Date	Author	Description of changes
0.0	17TH January 2025	GROUP_S	Initial draft created
1.0	22ND January 2025	GROUP_S	Revised requirements draft Updated
1.1	25TH January 2025	GROUP_S	Revised structure and formatting improvements
1.2	6TH February 2025	GROUP_S	Final version submitted for approval

DECLARATION

We declare that, we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission.

Signature…… Date: ……/……/……..

Seanice Nabasirye

Signature…… Date: ……/……/……..

Keith Paul Kato

Signature…… Date: ……/……/……..

Isaac Asiimwe

Signature…….. Date: ……/……/……..

Jessy Solomon Kiyingi

Signature…… Date: ……/……/……..

Jonah Akandwanaho

APPROVAL

The undersigned certifies that he has read and hereby recommends for the acceptance of the Makerere University System Project entitled "Academic Issues Tracking System" **(AITS)**

Signature…… Date: ……/……/……..

Dr. Peter Khisa Wakholi (Ph.D.)

Department of Computer Science

Makerere University

ACKNOWLEDGEMENT

We, in a special way, would like to take this opportunity to thank the almighty God that has given us life, wisdom to enabled us pursue our academic goals. Among which some have been geared towards us developing a working system (AITS) that meets the needs of our stakeholders. We still, in a special way, would like to thank our supervisor Dr. Peter Khisa Wakholi who, with continued guidance has enabled us reach satisfactory & successful completion of our project. And everybody else particularly in academia, that made any sort of contribution towards our project.

List of Tables

List of Figures

List of Abbreviations and Acronyms

Abstract

The Academic Issue Tracking System (AITS) in this case will be a comprehensive web-based solution designed to address the challenges faced by Makerere University in managing academic record-related discrepancies, missing marks, and registration errors, which often leads to frustration among students and inefficiencies in administrative workflows. AITS aims towards streamlining and digitizing these processes, fostering transparency, accountability, and efficiency among students, lecturers, and university administrators. The system employs a full-stack web development approach using Django for back-end and react for front-end, ensuring a scalable and robust application.

AITS provides role-based access control for three primary user roles namely, students, lecturers and administrators. Students will be able to log issues through a simple interface, monitor the status of their submission, and receive real-time notifications about updates or resolutions. Lecturers are empowered to manage assigned issues, providing updates and resolutions through an intuitive dashboard. Administrators will oversee the entire process, assigning issues, tracking their resolution progress, and analyzing trends through comprehensive reports.

To ensure reliability and ease of use, our AITS will integrate features such as RESTful APIs for seamless communication between components, a PostgreSQL database for secure and efficient data management, and a cloud-based deployment for accessibility and scalability. The project adheres to software development best practices including version control, automated testing, debugging, and user-centered design principles.

Table of Contents

1) CHAPTER ONE

1.1. Introduction

Makerere University faces challenges in efficiently managing academic record – related issues such as grade discrepancies, missing marks, and registration errors. These issues often involve lengthy processes that are prone to miscommunication and delays.

This Academic Issues Tracking System (AITS) aims to streamline the process of reporting, tracking and resolving academic record issues, ensuring transparency and accountability among students, lecturers, and administrators.

1.2. Problem Statement

At Makerere University, students frequently encounter challenges related to academic records, including missing marks, incorrect grades, and registration issues. The current process for reporting and resolving these issues is largely manual, inefficient, and somewhat lacks transparency. Students often have to physically visit offices, submit written complaints, or rely on email communication, leading to delays, miscommunication, and frustration

Lecturers and administrators, on the other hand, face difficulties in tracking and managing reported cases due to the absence of a centralized system. Without a structured workflow, some issues go unresolved, while others are duplicated or lost in the process. This lack of efficiency affects students' academic progress and delays administrative decision-making.

To address these challenges, the Academic Issue Tracking System (AITS) will provide a centralized digital platform where students can report academic issues, lecturers can review and resolve them whereas administrators can oversee and track the progress in real time. The system will enhance transparency, accountability, and efficiency by implementing role-based access, automated notifications, and real-time status tracking, ultimately improving the academic support services at Makerere University.

1.3. Purpose of the System

Through a role-based control, AITS will enable:

1. **Students** to be able to submit and track their academic issues in real time.
2. **Lecturers** to be able to review, manage, and update the status of assigned cases.

3. **Administrators** to be able to oversee the resolution process, assign tasks, and generate reports for decision-making

By leveraging modern web technologies like Django (for backend development) and React (for frontend development), AITS will ensure a user-friendly interface, seamless data management and real-time notifications to keep all stake holders informed. The system will not only improve efficiency but also enhance accountability, transparency, and overall student satisfaction in academic issue resolution.

1.4. General Objectives

1) To develop a web-based system in form of an Academic Issue Tracking System (AITS) that will effectively & seamlessly allow students to report academic issues.
2) To ensure timely resolution of academic issues through automated notifications and tracking.
3) To implement role-based access system for students, lecturers and administrators
4) To enable **real-time** notifications and status tracking for reported issues.
5) To ensure system scalability, security, and efficiency of our system.

1.5. Specific Objectives

1) To address Student issues satisfactorily.
2) To bridge the gap between Lecturers and Students as far as academic transparency is concerned.
3) To allow better collaboration between lecturers and Administrators on student's issues through the filtering Issue process our system will provide.
4) To foster informed decision-making by administrators about student's issues through reports and trends our system will provide.
5) To automate the student Issues reporting process and keep track of issue status to avoid miscommunication between lecturers and students about Issues.
6) To provide a cohesive approach to solving Student's Issues.
7) To help Assessors (Lecturers & Registrars) better understand the student's exact Issue to avoid misunderstandings thus taking appropriate measures towards solving students Issues.
8) To help Administrators effectively better plan for student's academic challenges before they even happen.
9) Documenting and presenting technical projects effectively.

1.6. Scope of the Project

The proposed project is to develop an Academic Issue Tracking System (AITS), a web-based application designed to save time by keeping track of Student's Issues such as missing marks, registration errors, remarks, medical emergencies along with other unique circumstances that would inadvertently negatively affect their academic progress. It is also designed to enhance collaboration and transparency between all involved parties to enhance the quality of service rendered by the institution (Makerere University).

1.6.1. Significance and beneficiary of the project

The beneficiaries of this system shall be Students, Lecturers, Academic Registrars because the system shall be able allow all of them to Submit Issues (Students), Review, Update and resolve Issues (Lecturers), handle escalated unique issues (Academic registrars) appropriately & timely. And finally, Administrators will be able to better plan for Makerere University students through making informed decisions.

1.7. Key Deliverables:

1) A fully functioning web application with an interactive dashboard.
2) A secure **REST API** for seamless communication between **front-end** and **back-end**.
3) Comprehensive **technical documentation** and a user manual.
4) Deployment of a system on a **cloud platform** for accessibility.
5) A final project **presentation** and **demonstration**.

1.7.1. Other Deliverable

1) A functional Academic Issue Tracking System (AITS)
2) User Authentication and role-based access control
3) An intuitive Student interface for Issue submission
4) A lecturer and Administrator dashboard for Issue resolution
5) Automated email/SMS notifications for updates
6) A reporting module for analytics and insights

1.8. Systems Requirements

1.8.1. Functional Requirements

- Students can submit academic Issues
- Lecturers can view and manage assigned Issue
- Administrators can assign Issues to appropriate personnel.
- System sends notifications for status updates.

1.8.2. Non-Functional Requirements

- ❖ The system should be accessible via modern web browsers such Microsoft edge.
- ❖ Data Security will be ensured through encryption.
- ❖ The system should be able to handle concurrent users efficiently.

1.9. Hardware and Software Requirements

1.9.1. Hardware Requirements

These will include the physical components needed for hosting, development, and user access

i) Server Requirements (Hosting Environment)

- **Processor Specifications sample:** Intel Xeon or AMD Ryzen 9 (minimum 8 cores, 3.5 GHz)
- **RAM Specs:** At least 32GB DDR4 (to handle concurrent users efficiently)
- **Storage Specs Sample:** 1TB SSD (Fast storage for database transactions)
- **Operating System:** Linux for better security and performance
- **Network Bandwidth:** Minimum **1Gbps network speed** for smooth data transfer
- **Database Server:** PostgreSQL or MySQL optimized for high-performance queries.

ii) Client Hardware Requirements

- **Desktop/Laptop:**
- **Mobile Devices:** Android (v10 or later)/ IOS (v13 or later) with at least 3GB RAM
- **Internet Connection:** Minimum 5 Mbps for smooth interaction with the system.
- **Display:** 1280 * 720 resolution or higher for proper UI rendering.

1.9.2. Software Requirements

These will include the necessary tools for development, deployment, and maintenance.

a) Development Tools

Since AITS is a full-stack web application, the following technologies will be used:

- **Backend Framework:** Django (Python) – for handling business logic and database interactions.
- **Frontend Framework:** React.js – for an interactive and modern user interface
- **Database System:** PostgreSQL or MySQL – for secure and structured data management.
- **Version Control:** Git + GitHub/GitLab – for tracking changes and collaborations.
- **IDE (Integrated Development Environment):**
  - ❖ **PyCharm** or **VS Code** for backend development.
  - ❖ **Visual Studio code** for frontend development.
- **Testing Frameworks:**
  - ❖ **Pytest** (for backend testing)

- ❖ **Jest** (for frontend testing)

b) Deployment Tools

To host and run the AITS system

- **Web Server:** Apache/Nginx (for handling web traffic)
- **Containerization:** Docker (for easy deployment and scalability)
- **Cloud Hosting (Optional):** AWS, Digital Ocean, or Google Cloud for scalability
- **CI/CD Tools:** GitHub Actions/Jenkins (for automated deployments)

c) Security and Maintenance Software

- **SSL Certificate:** For secure HTTPS connections
- **Firewall & Antivirus:** UFW (on Linux), Cloudflare (for DDoS protection)
- **Monitoring Tools:** Prometheus + Grafana (for system health monitoring)
- **Backup Solutions:** Automated daily backups via AWS S3 or local backup scripts

1.9.3. Programming Language

Component	Primary Language	Framework/ Technology
Backend	Python	Django/Django Rest Framework
Frontend	JavaScript	React.js
Database	SQL	PostgreSQL/MySQL
API Development	Python	Django Rest Framework
DevOps	Bash/YAML	Docker, CI/CD
Security	Python/JavaScript	JWT, OAuth

2) CHAPTER TWO

2.1. User Cases (Documentation)

- ❖ **Use case 1:**
- ❖ **Actors (Students, System)**
- ❖ **Description:** A student submits an Academic Issue (e.g., missing marks, course registration problems, lecturer disputes, timetable clashes,) for resolution.
- ❖ **Preconditions:**
  - The student must be registered in the system.

Main Flow

1. The student logs into AITS.
2. The student selects "Report Issue."
3. The student enters issue details (category, description, attachments, course details
4. The student submits the issue
5. The system generates an issue reference number and notifies the assigned university staff.

Alternative Flow

1. The student provides incomplete information, the system prompts for missing details.
2. If the student is not registered, access is denied, and registration is required.

Post Conditions:

- The issue is tracked and recorded in the system
- The student can monitor the progress of the issue

❖ **Use case 2: (Administrator assigns an issue to a staff member)**

❖ **Actors (administrators, Academic Staff, System)**

❖ **Description:** The administrator assigns issue to the appropriate department or lecturer for resolution.

❖ **Preconditions:**

- The issue must already be submitted by the student.

Main Flow

1. The administrator logs into AITS
2. The administrator views unassigned academic issues.
3. The administrator selects an issue and assigns it to a relevant staff member
4. The system notifies the staff member about the assignment.

Alternative Flow

5. If the administrator assigns an issue to a wrong staff member, it can be reassigned.
6. If no staff is available, the issue remains in the unassigned queue

Post Conditions:

- The issue is assigned to a responsible staff member.
- The student is notified of the progress.

❖ **Use case 3: (Staff reviews and resolves an issue)**
❖ **Actors (Academic staff, System)**
❖ **Description:** The assigned staff member reviews, updates, and resolves a reported issue.
❖ **Preconditions:**

- The issue must be assigned to a staff member

Main Flow

1. The staff member logs into AITS.
2. The staff member views assigned issue.
3. The staff member updates the issue status (e.g., In progress, Resolved).
4. The staff member provides a resolution (e.g., corrected marks, clarification).
5. The system notifies the student about the resolution.

Alternative Flow

6. If the issue requires further action, the staff can escalate it.
7. If the staff member is unavailable, the issue can be reassigned.

Post Conditions:

- The issue is either resolved or escalated.
- The student is informed of the outcome.

❖ **Use case 4: (Student reviews resolution and closes issue)**
❖ **Actors (Student, System)**
❖ **Description:** After an issue is marked as resolved, the student reviews the response and either accepts or reopens the issue.
❖ **Preconditions:**

- The issue must have a resolution provided.

Main Flow

1. The student logs into AITS.

2. The student reviews the resolution details.
3. If satisfied, the student confirms resolution and closes the issue.
4. If unsatisfied, the student reopens the issue with additional comments.

Alternative Flow

5. If the student does not respond within a set time, the system may auto close the issue.

Post Conditions:

- The issue is either open or closed for further action.

❖ **Use case 5: (Admin Generates reports on Issue Trends)**
❖ **Actors (Administrator, System)**
❖ **Description:** The administrator generates reports on issue types, resolution times, and department performance
❖ **Preconditions:**
- There must be resolved and pending issues in the system.

Main Flow

1. The administrator logs into AITS.
2. The administrator selects the "Generate Reports" option.
3. The administrator specifies report parameters (e.g., data range, issue type).
4. The system generates and displays the report.

Alternative Flow

1. If no data matches the filters, the system notifies the administrator.

Post Conditions:

- The report is available for analysis

- **Use case 6: (Staff escalates an Issue to a Higher Authority)**
- **Actors (Academic staff, Head of Department, System)**
- **Description:** If the lecturer or staff member cannot resolve an Issue, they escalate it to a higher authority.
- **Preconditions:**
    - The issue must be assigned to a staff member

Main Flow

1. The staff member logs into AITS.
2. The staff member selects “escalated Issue.”.
3. The staff member chooses the appropriate higher authority (e.g., HOD, Dean).
4. The system notifies the higher authority.

Alternative Flow

5. If no higher authority is available, the system queues the escalation.

Post Conditions:

- The issue is either is assigned to a higher authority for resolution.

- **Use case 7: (System Sends Automated Notifications)**
- **Actors (System)**
- **Description:** The system sends notifications to users regarding issue updates, resolutions, or escalations.
- **Preconditions:**
    - Users must have registered emails or phone numbers.

Main Flow

1. A student submits an Issue
2. The system sends a confirmation notification.
3. An Issue is updated by staff.
4. The system notifies the student of progress.
5. The issue is resolved.
6. The system notifies the student to review the resolution.

Alternative Flow

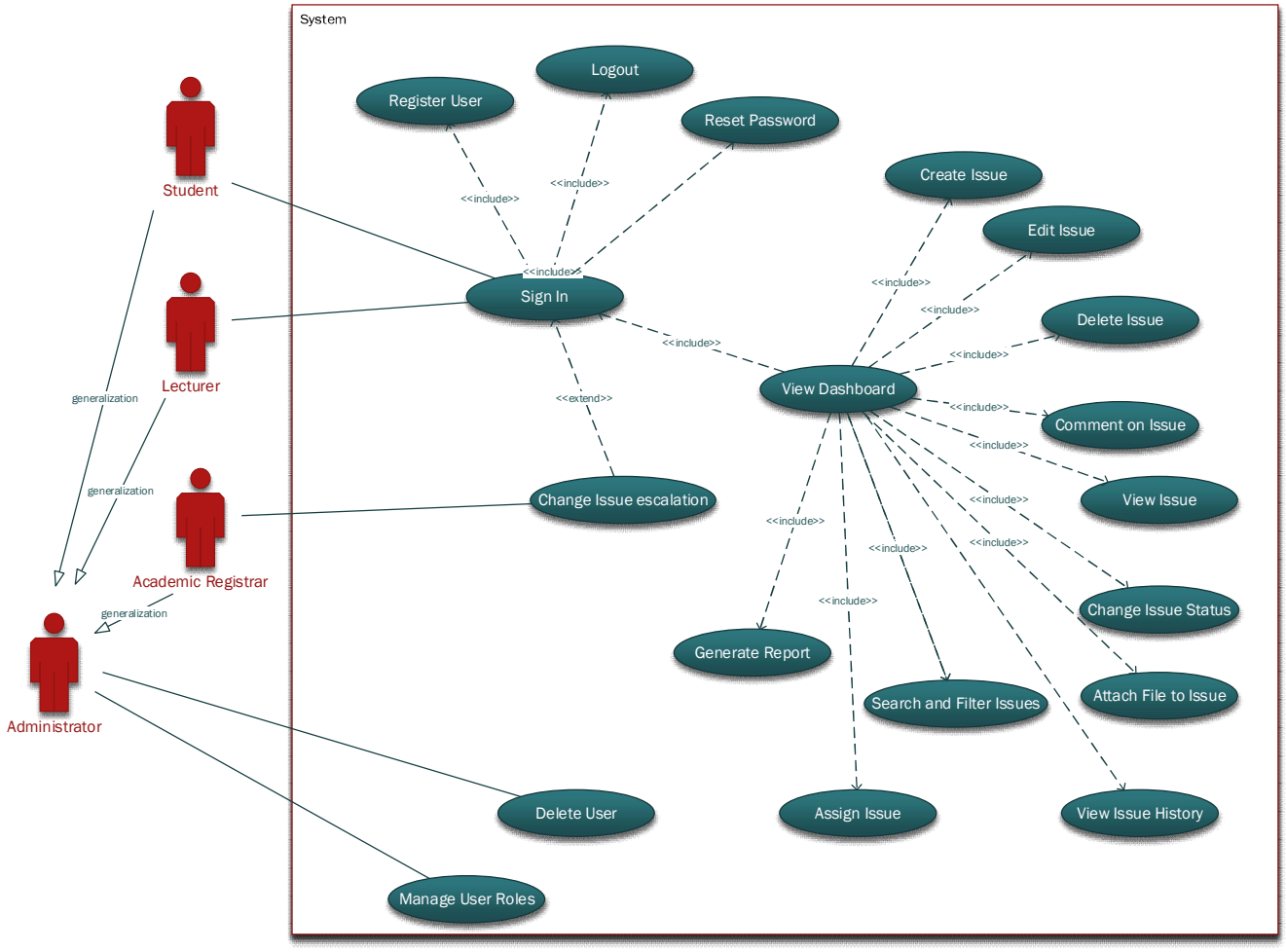
7. If notifications fail, the system retries sending them.

Post Conditions:

- Users receive timely updates on their issues.

2.1.1. User Stories

https://github.com/Jetlee-lab/GroupProject-Repo



USER STORIES CONCEPT

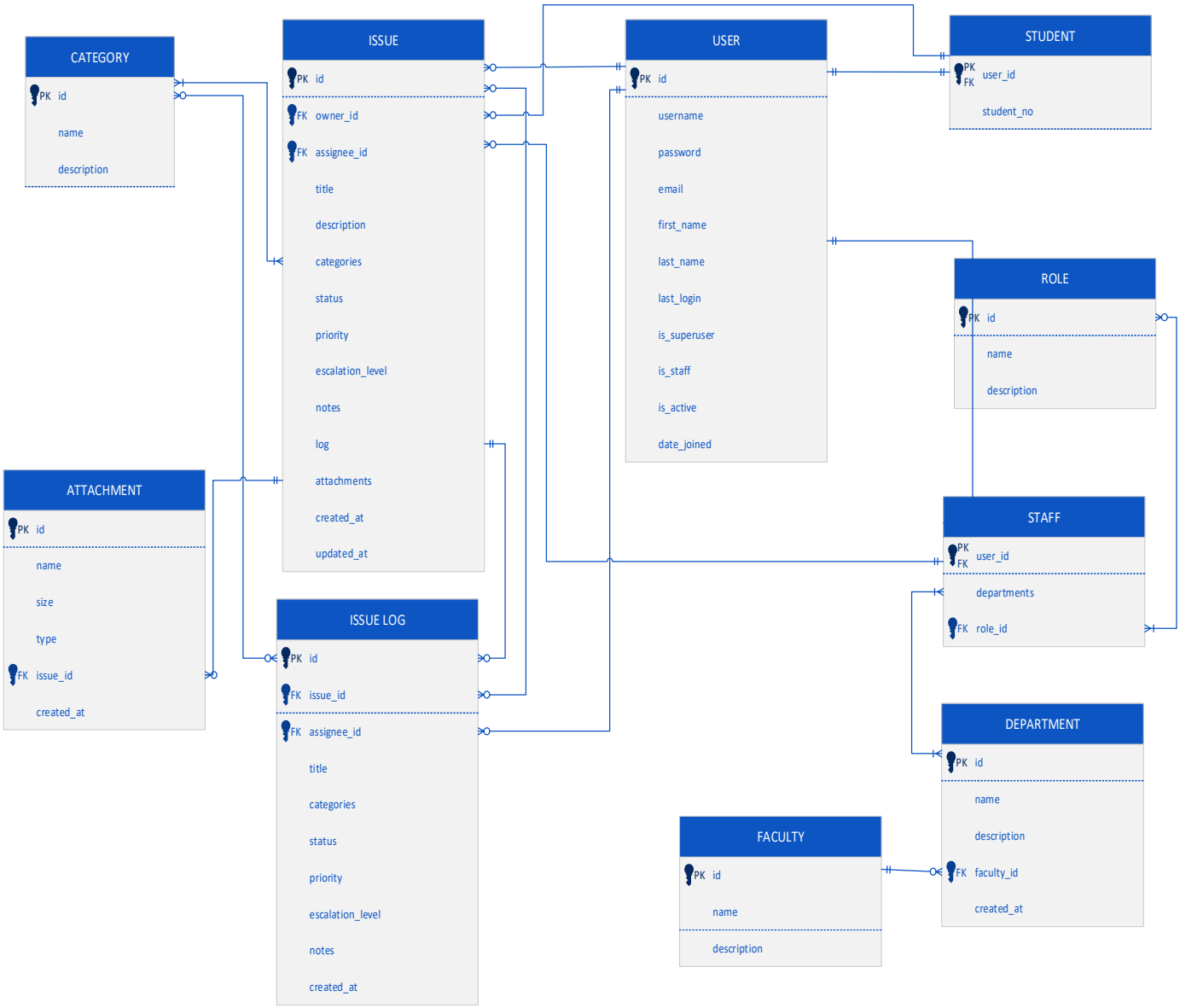
https://github.com/Jetlee-lab/GroupProject-Repo

As a student, I want	As a lecturer, I want	As an Academic Registrar, I want
❖ To create an account so that I can login and access the system	❖ To login to the system Securely so that I can access my dashboard.	❖ To login to the system Securely so that I can access the system.
❖ To Login securely so that I can access my dashboard.	❖ To filter and sort issues by course, student or issue type so that I can manage them efficiently	❖ To view all academic issues submitted so that I can oversee the resolution process.
❖ To submit an academic issue	❖ To respond to student issues by providing solutions, explanations, or necessary actions.	❖ To assign or reassign issues to lecturers if they are misdirected.
❖ To view the status of my submitted issue so that I can track its progress	❖ To escalate issues to the academic registrar if they require high-level intervention.	❖ To track unresolved and escalated issues so that I can take appropriate actions.
❖ To edit or update my submitted issue before it is reviewed to provide additional details	❖ To view my history of resolved issues so that I can refer back to previous cases.	❖ To view system analytics and reports. (e.g., issue trends, resolution rates) so that I can make data – driven decisions.
❖ To withdraw a submitted issue if I no longer need assistance	❖ To receive notifications when students respond to my feedback or when new issues are assigned to me.	❖ To set deadlines for issue resolution to ensure timely response.
❖ To chat, or communicate with the lecturer or registrar regarding my issue so that I can provide clarification.	❖ To track issue resolution metrics (e.g., response time, number of resolved issues) for performance evaluations.	❖ To send reminders and notifications to students and lecturers regarding pending issues.
❖ To upload supporting documents e.g., medical proof when submitting an issue.		❖ To update policies and guidelines for handling academic issues
❖ To view history of my past issues so that I can reference them if needed		❖ To manage user accounts and permissions to ensure data security.

2.1.2. Functional Requirements

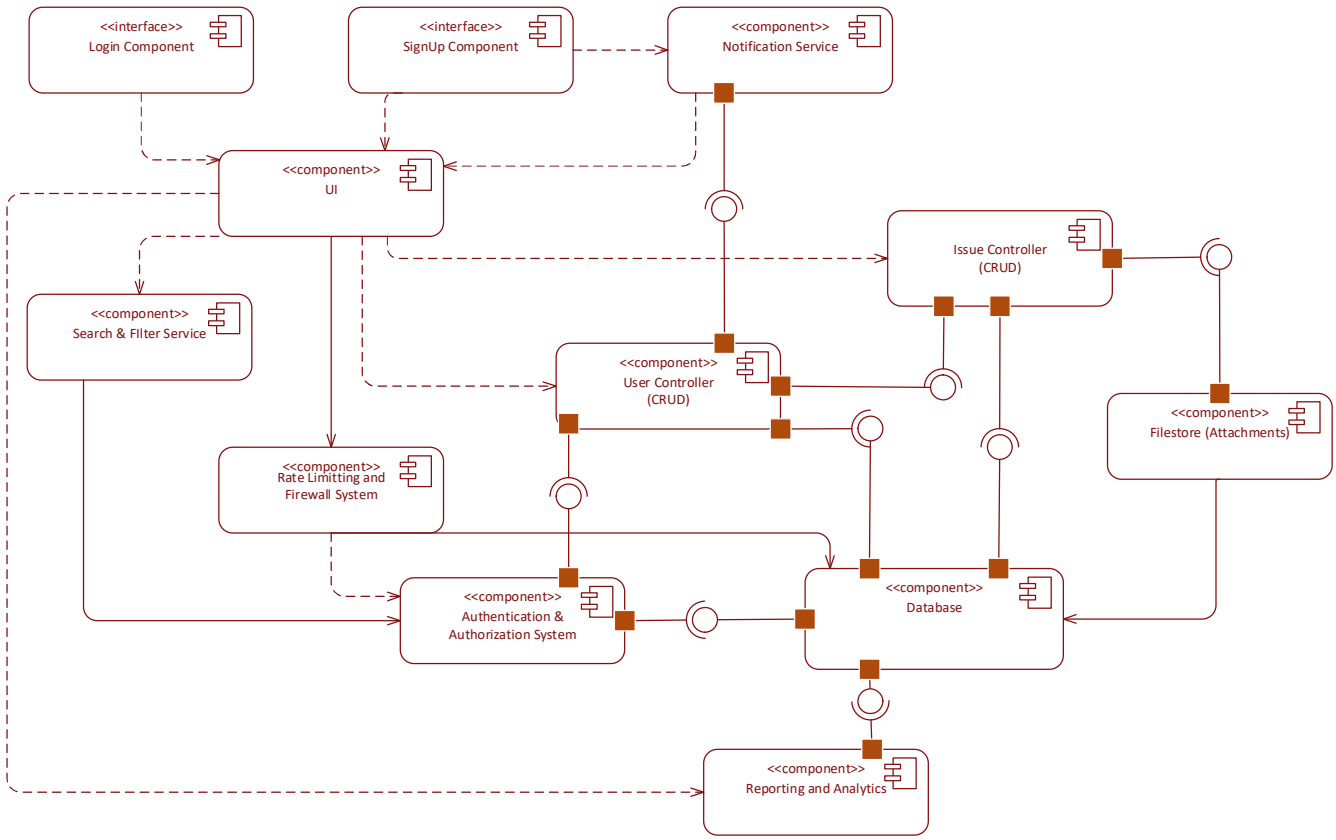
Database Schema (E.R.D)

https://github.com/Jetlee-lab/GroupProject-Repo



2.1.3. UML (Components) diagram.

https://github.com/Jetlee-lab/GroupProject-Repo



3) CHAPTER THREE

3.1. Methodology Introduction

This methodology section outlines the approach, techniques, and tools used to develop the Academic Issue tracking System (AITS) for Makerere University. The selection of an appropriate software development model, programming languages, frameworks, and database management system ensures that the system is efficient, scalable, and user – friendly. This chapter will provide an overview to the development model, data collection techniques, system implementation strategy, and testing procedures to be followed in the development process.

3.2. Developmental Model

For the **Academic Issue Tracking System (AITS)**, We will adapt the **Agile development model** due to its flexibility, iterative nature, and ability to accommodate changes based on user feedback.

❖ Justification for using Agile

- Allows continuous user feedback from students, lecturers and administrators

- Ensures incremental delivery, meaning that essential features are developed and tested first.
- Accommodates changes in system requirements during the development process.
- Provides a high-quality product by continuously testing and refining features.

❖ **Agile Development phases for AITS.**

Phase	Description
Requirement Gathering & Analysis	• Identifying user needs and documenting functional & nonfunctional requirements.
Planning & Design	• Designing system architecture, including UML diagrams, ERD, wireframes, and database schema.
Implementing & Coding	• Developing the system backend using Django, frontend using react.js, and database using PostgreSQL.
Testing & debugging	• Conducting unit testing, integration testing, and user acceptance testing (UAT)
Deployment	• Deploying the system to a cloud server (AWS/Azure) and integrating it with Makerere University's existing platforms.
Maintenance & Updates	• Ensuring continued support, bug fixes, and feature enhancements.

References

- Sommerville, I. (2015). Software Engineering (10th ed.). Pearson.
- Pressman, R. S. & Maxim, B. (2020). Software Engineering: A practitioner's Approach (9th ed.). McGraw-Hill.
- Makerere University IT Policies and Guidelines, Retrieved from: https://mak.ac.ug
- Agile Alliance. (2023). "Introduction to Agile Software Development." Retrieved from: https://www.agilealliance.org

Appendices

Appendix A: Acronyms Used in this document

Acronym	Full Form
AITS	Academic Issue Tracking System
UML	Unified Modeling Language
ERD	Entity-Relationship Diagram
UAT	User Acceptance Testing
API	Application Programming Interface
RBAC	Role-Based Access Control

Project Budget

The estimated budget covers hardware, software, human resources, and other miscellaneous costs needed to develop the AITS at Makerere University as shown below.

Item	Description	Estimated Cost (UGX)
Hardware	• Servers (cloud or premises) • Developer Workstations (2 high-end laptops) • Internet & Hosting Fees (1 year)	• **4,500,000** • **8,000,000** • **2,000,000**
Software	• Django (Open – Source) • React.js (open – Source) • PostgreSQL Database • GitHub (Private Repositories) • SSL certificate	**Free** **Free** **Free** • **1,200,000** • **500,000**
Human Resources	• Developer (2 for 3 months) • UI/UX Designer • System Tester	• **9,000,000** • **2,500,000** • **1,500,000**
Other Expenses	• Training for staff and students • Documentation & Reports • Contingency (10% of cost)	• **3,000,000** • **1,000,000** • **3,720,000**
Total Estimated Budget	• Overall System Development & Deployment	• **37,920,000**