**GROUP N MEMBERS:**

* VICTOR IGNATIUS - **BSCLMR158123**
* RICH BENJAMIN **- BSCLMR161123**
* AUGASTINE NJOROGE **- BSCLMR370922**
* JOYMELLA PHERRY - **BSCLMR158623**
* ENOCH MWENDWA **- BSCLMR160423**
* HERON KIPKIRUI - **BSCLMR231623**

**VOTING SYSTEM DOCUMENTATION**

**Project Plan for the Central Identification platform**

**Objective**

The primary objective of the project is to develop and implement a centralized identification platform that integrates biometric data such as fingerprints, eyes, faces, voices, DNA, and location of the Kenyan population. This platform will serve as a comprehensive database linked to various services including identity cards to access education, health, and social services.

**Project Scope**

The scope of the project includes:

•Designing and developing the central identification platform infrastructure.

•Collecting and storing biometric data from the Kenyan population.

•Integrating the platform with existing government systems and services.

• Implementing security measures to safeguard personal dataand encryption standardsto prevent unauthorized access.

**Resource Requirements**

* Human Resources: Project Manager, IT Specialists, Data Analysts, Security Experts, Training Personnel and cooperation and collaboration from public and government agencies and stakeholders.
* Infrastructure: Data Centers, Biometric Data Collection Equipment, Servers, Networking Equipment.
* Software: Biometric Data Processing Software, Security Software, Integration Tools.
* Budget: Funding for procurement, infrastructure setup, personnel salaries, public awareness campaigns, and ongoing maintenance.

**Methodologies**

* Agile Project Management: Agile methodologies can be effective for managing large-scale projects with evolving requirements. It allows for iterative development and frequent feedback loops, which can be beneficial when dealing with a project that involves integrating multiple technologies and data sources.
* Waterfall Project Management: Despite its traditional nature, the Waterfall methodology could be suitable for a project with well-defined stages and deliverables. Each phase could be carefully planned and executed, ensuring thoroughness and precision in the development process.
* Risk Management: Given the concerns raised about security breaches and privacy issues, implementing a robust risk management framework is essential. Techniques such as risk identification, assessment, mitigation, and monitoring should be integrated into the project management process to address potential threats effectively.
* Quality Management: Quality assurance and control should be prioritized throughout the project lifecycle by establishing clear quality metrics, conducting regular inspections and audits, and implementing continuous improvement processes to ensure the reliability, accuracy, and security of the centralized identification platform.

**Project Timeline**

* Duration: 1 month

**Key Milestones**

1. Planning (1 day)

2. Requirements Analysis(2 days)

3. System Design and Architecture (4 days)

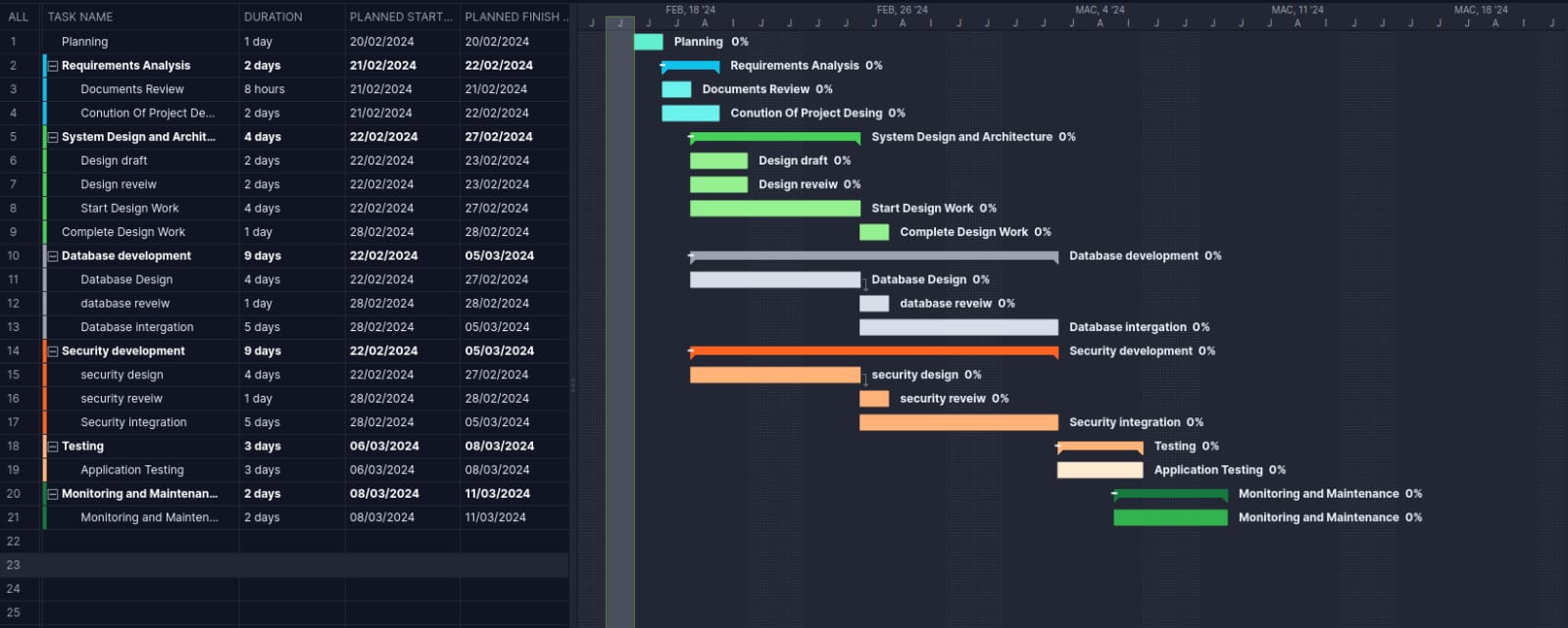
4. Database development (9 days)

5. Security development (9 days)

6. Testing (3 days)

7. Monitoring and Maintenance (2 days)

**GANTT Chart**



1. **SOFTWARE REQUIREMENTS SPECIFICATION (SRS)**

**Scope of project:** The scope of the project includes the development of a Python-based application that allows currently registered students to participate in campus nominations for yearly elections. The application will be developed utilizing the Flask web framework for Python. Flask is a lightweight and flexible framework that provides essential tools for building web applications. Its minimalistic design allows for greater flexibility and customization, making it suitable for the development of the voting platform with specific requirements. The application will provide safety features to ensure that each student can cast only one vote and maintain the principle of a secret ballot. Additionally, the application will integrate security measures to protect user data and address concerns regarding privacy breaches.

**Functional requirements**:

* **User authentication** - **Verification of Identity**: The authentication system shall validate the identity of users by requiring them to provide their university credentials. **Login Mechanism:** Users shall be presented with a login interface where they can input their username and password. **Registration Status Check**: Before allowing users to authenticate, the system shall verify whether they are currently registered students. This check ensures that only eligible students can participate in the elections. **Session Management**: Upon successful authentication, the system shall establish a session for the user, allowing them to interact with the application without needing to log in repeatedly during the same session.
* **Voting Process: User Authorization**: Only authenticated users shall be allowed to access the voting functionality of the application. Upon successful authentication, users will be granted permission to cast their votes. **Candidate Selection**: Each student shall have the opportunity to select one candidate for each available position in the election. The application shall present a list of nominated candidates for each position, allowing users to make their selections. **Vote Recording**: Once a user casts their vote, the application shall record it securely in the system's database. This recording ensures the integrity of the voting process and prevents duplicate or fraudulent votes. Vote **Modification Restriction:** After submitting their votes, users shall not be permitted to modify their selections. This restriction maintains the confidentiality and integrity of the voting process, preventing any attempts to alter votes after they have been cast. **Anonymous Voting**: The application shall enforce the principle of a secret ballot, ensuring that each user's vote remains anonymous. This anonymity fosters trust and integrity in the electoral process, allowing users to vote without fear of reprisal or coercion.
* **Candidate Nomination**: The application shall provide a mechanism for the nomination of candidates by eligible students. This mechanism includes a nomination form where students can input relevant details such as their name, position they are running for and a brief statement outlining their candidacy. Authorized administrators shall review and approve candidate nominations before they appear on the ballot.
* **Results Display**: After the voting period ends, the application shall display the election results. Results shall be displayed in a manner that preserves the anonymity of individual voters.

**NON-FUNCTIONAL REQUIREMENTS**

* **Security:** The application shall implement strong encryption mechanisms to protect user authentication credentials and voting data. Access controls shall be enforced to restrict unauthorized access to sensitive functionalities and data. Regular security audits and vulnerability assessments shall be conducted to identify and address potential security threats.

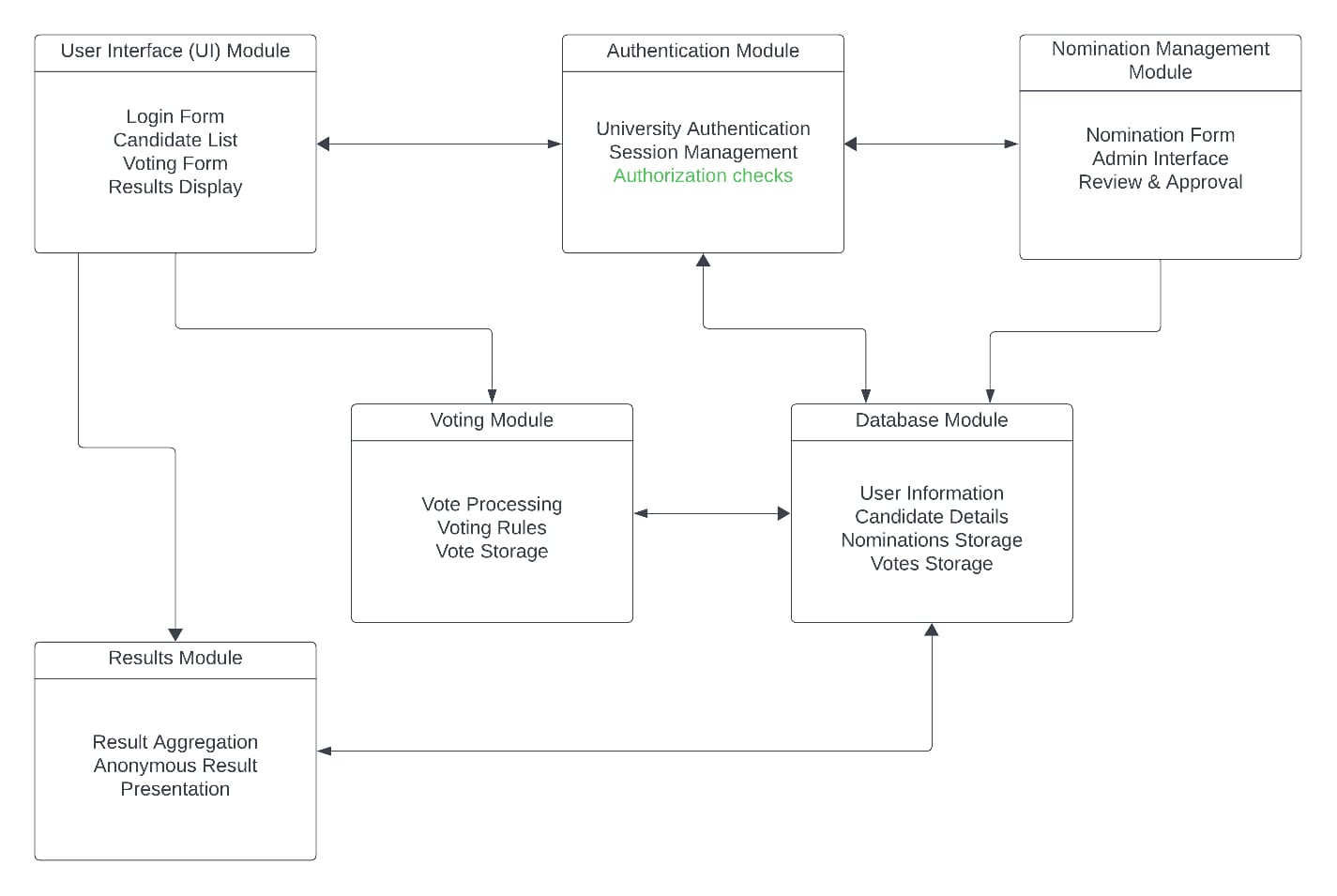


Figure 1 System module digram

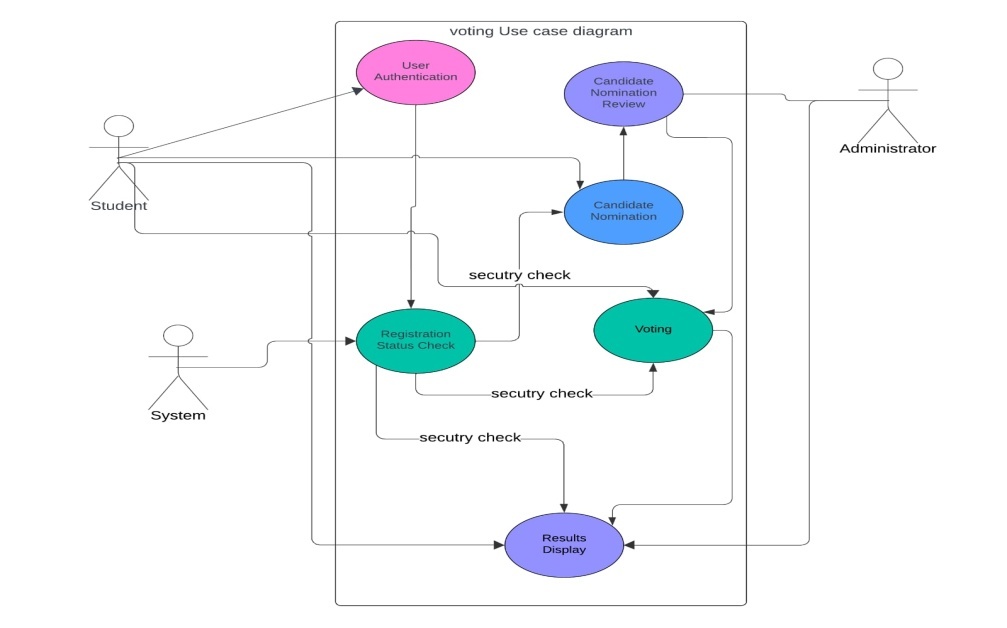


Figure 2 use case diagram

**Test Objectives:**.

**To confirm that every student is limited to one vote:**

• Usability: Assess the application's usability in terms of navigation, user interface, and feedback from users.

• Security: Confirm that the application has safeguards in place to stop illegal access, data breaches, and vote manipulation.

To guarantee that the campus nominations are limited to students who are actively enrolled.

•Performance: Evaluate the application's scalability and response times under both typical and heavy load conditions.

•Compatibility: Verify that the program works with various operating systems, browsers, and devices.

**To confirm that the secret ballot concept is upheld.**

•Accessibility: Check that the program complies with accessibility standards and is usable by people with impairments.

Test Strategies:

• **Unit Testing:** Test individual application module (functions, methods, classes) to ensure they work correctly in isolation.

• **Acceptance Testing:** Validate the application against user requirements and ensure it is ready for deployment.

**• Security Testing:** We will test the application's security and ensure that it protects the privacy of the students. This includes testing for vulnerabilities, input validation, and access controls.

• **Usability Testing:** Have users test the application to evaluate its ease of use and user satisfaction.

• **Performance Testing:** Test the application's performance under various load conditions to identify and address bottlenecks.

**Entry Criteria:**

These are the requirements that must be fulfilled before actual application testing can begin, such as finished development and the availability of test environments, in order for testing to start.

• The test environment should be configured and prepared for use.

• The application should be finished and prepared for testing.

• It is necessary to examine and approve the test cases and test plan.

• There shouldn't be any serious flaws in the program that might affect testing.

**Exit Criteria:**

Requirements that must be fulfilled, including passing every test case and hitting performance goals, in order to finish testing and go on to the next stage.

• Every test case needs to have been run through, executed and passed.

• Every problem that is rated as high or medium priority has to be repaired and retested.

• The application must fulfill all criteria, including functional and non-functional.

• The application has to be secure and safeguard students' privacy.

• The application need to be simple to use and easy on the eyes, i.e UI and UX (User Interface and User Experience.

• The application should be able to accommodate multiple logins for users and votes on the application.

Test Cases and Procedures:

1. Requirement; verify that only registered students can log in.

* Test Procedure:
* Register a new student.
* Attempt to log in with the new student's credentials.
* Verify that the login is successful.
* Log out.
* Attempt to log in with a non-registered student's credentials.
* Verify that the login is unsuccessful.

**Expected** **Result**: Only registered students can log in.

1. Requirement: Verify that each student can only cast one vote.

* Test Procedure:
* Log in as a registered student.
* Cast a vote.
* Log out.
* Log in again as the same student.
* Attempt to cast another vote.
* Verify that a message is displayed indicating that the student has already voted.

**Expected Result**: Each student can only cast one vote.

**THE TRACEABILITY MATRIX**

**Requirements and Project Artifacts:**

* Registered students' participation
* Each student casts one vote
* Secret ballot principle
* Python application development
* User-friendly application

**Authentication feature**= Database for student information and votes

**Establish Traceability Links:**

* Requirement 1 linked to Design Document Section 2.3, Source Code File authentication.py, Database Table Students.
* Requirement 2 linked to Source Code File voting.py, Database Table Votes.
* Requirement 3 linked to Design Document Section 4.5, Source Code File encryption.py.
* Requirement 4 linked to all Python source code files.
* Requirement 5 linked to Design Document Section 3.1, User Manual Section 2.
* Requirement 6 linked to Design Document Section 2.3, Source Code File authentication.py.
* Requirement 7 linked to Database Schema Document, Source Code Files for database operations.

**REFERENCES:**

**Sommerville, I. (2021). Software Engineering (9th ed.). Pearson Education Limited.**

**Sommerville, I. (2021). Software Engineering: A Practitioner's Approach (9th ed.). Pearson Education Limited.**