****

**ST. PAUL’S UNIVERSITY**

**(LIMURU CAMPUS)**

**FACULITY OF COMPUTER SCIENCE AND COMMUNICATION STUDIES**

**BCS 3106: SOFTWARE ENGINEERING II**

**ASSIGNMENT: INITIAL DOCUMENTS FOR SEMESTER PROJECT**

**GROUP E:**

**BSCLMR110623**

**BSCLMR151323**

**BSCLMR150723**

**BSCLMR328421**

**BSCLMR101721**

**BSCLMR106823**

**LECTURER(S): Cecilia Nanfuka**

**DATE OF SUBMISSION: 20th February, 2024**

**PROJECT PLAN**

**1.Requirements Gathering:**

**Interviews:** University council, administrators and students who have ever taken part in the election.

**Document review:**Election procedures & university student information databases.

**Research:**Similar candidates’ application forms in other universities and secure voting systems good enough to be adapted for the nomination process.

**2. System Design & Modelling:**

**Architecture diagram:**All components of a system as well as relationships between them.

**Data flow diagrams (DFDs):** Functions such as user functionality, nomination functionality and reporting functionality among others.

**Entity-relationship diagram (ERD):** User, nomination and position entities with attributes and relationships among these three entities.

**Use case diagrams:** Various users using multiple methods of interactions with a system

**3. Development & Testing:**

Selection of framework for Flask  
Secure coding practices like password hashing, input validation, data encryption  
Authentication and authorization through role based access control (RBAC)  
Unit testing, integration testing and system testingg

**4. Documentation & Delivery:**

User manual together with training materials for administrators and students.  
Technical documentation containing system architecture, API references and design decisions

Software delivery along with documentation to the University Council

**5. Deployment & User Training:**

a) Server deployment: Setting up safe environment with right access controls;  
  
b) User training: Conducting workshops, online tutorials, and support channels;  
  
c) Testing and monitoring: Sustaining seamless running of operations with due consideration to defects solving

**Software Requirements Specification (SRS)**

**1. Introduction:**

System name : University Nomination Application  
  
Purpose : To facilitate secure and efficient student nominations for annual campus elections.

**Target audience**: University Council, administrators, registered students.

**2. Functional Requirements:**

**User Roles and Permissions:**

**Student:**

• Check a series of the open nomination categories alongside their explanations.  
  
• Find eligible nominees by name or other important criteria.  
  
• Propose a nominee for each category (one nomination per category only).  
  
• Change or cancel an earlier nomination.

**Council members(admins):**

• Get information about any nominations for all categories.  
  
• Narrow down nominations through candidates, selection title or keyword.  
  
• Export nominations data for analysis purposes.  
  
• Closing and opening of nominations periods is done by admins including management of deadlines.  
  
• Admins are able to create, amend and delete user accounts (student & admin).

**Nomination Process:**

• Registered students can nominate themselves or other students for specific positions  
  
• Nominations include candidate information (name, program, platform)  
  
• A student may make one nomination per position only  
  
• The voting process is anonymous; therefore, no student or admin can see who nominated whom  
  
.• Secure storage of nomination data in a tamper-proof database

**Reporting:**

.• Nomination received reports by admins  
  
.• Tallies show candidate names and counts without disclosing voters  
  
.• Data in different file format like PDF/CSV can be exported to other systems

**3. Technical Requirements**

Programming language: Python Programming Language version 3.7 onwards   
Web framework: Flask   
Database: Secure as well as scalable database system like MySQL

Authentication: University system connect   
Security: Implement proper security measures including encryption, data protection and vulnerability patching.  
User interface: The application should be easy for students as well as council members to use on their own.

**4.Security Requirements**

Authentication: Authentication of a user with valid university credentials shall be required in order to access the application.  
  
Authorization: User roles must be clearly defined and access to functions must be restricted according to these roles.  
  
Data encryption: All critical information such as student IDs, and nominations should be encrypted in transit as well as at storage points.  
  
Password protection: Strong password policies should be enforced, including minimum length, complexity requirements, and regular password changes.

**5.Non-Functional Requirements**

Performance: Nominations’ demand response which is quick and efficient.  
  
Scalability: Application should handle numerous concurrent users during peak nomination periods.  
  
Usability: This software has an intuitive user interface that works for any kind of user. Instructions are clear; error messages are given.  
  
Accessibility: Works across various devices and browsers.

**Requirements Traceability Matrix (RTM):**

**Key Functionalities:**

1. **User Registration and Login:**

• Make sure that you have used secure access codes that are difficult to crack.  
  
• Consult with the University’s student information system to confirm whether a user is valid.

1. **Nominations:**

• Enable authorized users to make nominations in different categories.  
  
• Gather basic information about nominees, such as their qualification criteria.  
  
• Introduce moderation options so as to assess and approve nominations.

1. **Voting:**

• Present nominated students for each category to verify others.  
  
• The use of cryptographic techniques ensures that votes are unique and anonymous at the same time always being recorded and stored in non-modifiable database.

1. **Results:**

• Determine voting outcomes upon reaching a deadline  
  
• Safeguard individual ballot confidentiality while disclosing aggregated findings

1. **Security:**

• Use best practices for encryption in storage of data or transfer it over electronic media  
  
• Create access control processes across different user types  
  
• Update applications/libraries regularly considering security vulnerabilities addressed

**REFERENCES**

Pressman, R. S., & Dr, B. R. M. (n.d.). *Software Engineering: A Practitioner’s Approach*. McGraw-Hill Education.

Fowler, M. (n.d.). *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley Professional.

IEEE SA. (n.d.). *IEEE Standards Association*. IEEE Standards Association. <https://standards.ieee.org/standard/830-1998.html>

Wikipedia contributors. (2024, February 11). *Blind signature*. Wikipedia. <https://en.wikipedia.org/wiki/Blind_signature>

*Flask-Login — Flask-Login 0.6.3 documentation*. (n.d.). <https://flask-login.readthedocs.io/en/stable/>

*Cybersecurity Framework | NIST*. (2024, February 1). NIST. <https://www.nist.gov/cyberframework>

*Welcome to Flask — Flask Documentation (2.2.x)*. (n.d.). <https://flask.palletsprojects.com/en/2.2.x/>

*OWASP Top 10:2021*. (n.d.). <https://owasp.org/Top10/>