FEU Institute of Technology

COLLEGE OF ENGINEERING • COLLEGE OF COMPUTER STUDIES

COMPUTER SCIENCE DEPARTMENT

CS0053 - CS SPECIALIZATION 2 - PROGRAMMING TOOLS AND TECHNIQUE

CASE STUDY 1

Student Database System

Submitted by:

De Ramos, Joseph Michael J.
Gaudia, Jose Mari B.
Molina, Arjen Catherine P.
Monje, Marie Kristela Frances A.

TN32

October 3, 2023

I. Introduction

Purpose

This documentation serves as the blueprint for the development and implementation of a C++-based student database system tailored for Elmwood University. Elmwood, renowned for its century-long history and diverse student body encompassing 20,000 learners across various programs, is seeking a solution to address the shortcomings of its traditional paper-based student record-keeping system.

Elmwood University faced significant hurdles within its traditional system. Vital tasks like class registration, grading, and maintaining student records were time-consuming due to manual processes. Frequent errors cropped up in student records as multiple individuals manually entered the same data, leading to complications.

Accessibility to student information posed difficulties for teachers and staff, requiring them to be physically present in one location for retrieval. Moreover, concerns about the safety of their important data loomed large, given the risk of papers being misplaced or damaged. Consequently, the university was compelled to seek a more efficient and modernized approach to address these challenges.

Based on the client's outlined challenges, the programmers undertook the development of a solution known as the Student Database program, utilizing C++ as the programming language. This program serves as the pivotal instrument in facilitating Elmwood University's transformation from its paper-based student record-keeping system to a modern digital database system.

It has been designed to address and resolve the myriad challenges posed by the traditional system, offering a streamlined and efficient approach to managing student information, academic records, and administrative processes. Through the implementation of this program, Elmwood University seeks to enhance the overall operational efficiency, data accuracy, accessibility, and security, ushering in a new era of academic excellence and modernized information management.

Scope

The scope of this document serves as a guiding beacon for Elmwood University's ambitious journey toward an efficient and technologically advanced student data management system. This endeavor encompasses the complete overhaul of the existing system, redefining how the institution collects, organizes, and leverages student data.

However, it's important to delineate the boundaries of this document's scope. While it offers a bird's-eye view of the project's objectives, challenges, and outcomes, it does not delve into the intricate technical intricacies of system implementation. Instead, it takes a high-level approach, providing a strategic perspective on Elmwood University's transition to a digital student database system.

In essence, this document serves as a compass, guiding readers through the broad terrain of Elmwood's modernization journey, from identifying the problem to articulating the outcomes achieved. It outlines the landscape without venturing into the terrain, offering clarity on what lies ahead for the university.

Definitions, Acronyms, and Abbreviations

- UI - User Interface
- ERD - Entity Relationship Diagrams
- C++ - programming language
- RBAC - Role-Based Access Control
- GCC - GNU Compiler Collection
- RAM - Random Access Memory
- SSD - Solid-State Drive

References

C++ standard library headers. (n.d.). Cppreference.com. https://en.cppreference.com/w/cpp/header

Follow, G. (2016, September 27). Student data management in C++. GeeksforGeeks. https://www.geeksforgeeks.org/student-data-management-c/

Harwani, P. (2022, October 31). Student record management system in C++. Scaler Topics. https://www.scaler.com/topics/student-record-management-system-cpp/

Student management system in cpp with file handling - working code. (n.d.). NarendraAliani. https://narendraalianionline.blogspot.com/2015/10/student-management-system-in-cpp-with.html

Udacity Team (2021, May 7). How To Read From a File in C++ https://www.udacity.com/blog/2021/05/how-to-read-from-a-file-in-cpp.html

Yillié. (n.d.). MSP: A Student portal for a single semester of a single department utilizing C++/CLI WinForms and is based on NUCES FAST's flex portal.

(N.d.). Codespeedy.com. https://www.codespeedy.com/student-data-management-implementation-using-c/

Overview

The rest of the document goes into greater detail about Elmwood University's digital student database system. It underlines the product's relevance in upgrading administrative operations, removing obstacles, and boosting user efficiency and data security. The document goes over the system's capabilities, login/logout functionality, digital enrollment, grade management, and searching for student records.

It also describes the system's operating environment, limitations, assumptions, and dependencies during design and implementation. Additionally, it will also tackle the system features and requirements of the modernized database system of the university. It describes the system's performance metrics, user interface, and data management that greatly benefit the university’s efficiency and accuracy.

II. Overall Description

Product Perspective

The development and implementation of a digital student database system at Elmwood University represent a critical step in modernizing the institution's administrative processes and aligning them with contemporary technological advancements. This system is situated within the broader context of the university's operations, with the aim of enhancing efficiency, data accuracy, and security while providing better services to students and stakeholders.

Traditionally, Elmwood University relied on a paper-based student record-keeping system, which posed numerous challenges, including delays in processes, data inconsistencies, limited accessibility, and security concerns. The adoption of the new digital system addresses these issues comprehensively.

Within the educational landscape, this digital student database system places Elmwood University in a more competitive position. It allows the institution to meet the evolving expectations of students, faculty, and staff in an era characterized by digital transformation. By streamlining data entry and retrieval processes, the system not only reduces administrative burdens but also facilitates faster decision-making, ultimately improving the quality of education offered by the university.

Moreover, this system integrates seamlessly with other institutional modules, such as financial and housing systems. This integration enhances cross-functional efficiency, enabling better coordination among various university departments.In summary, the digital student database system is not an isolated application; it's a pivotal component that augments Elmwood University's operational capacity, data management, and services. It serves as a cornerstone in the institution's commitment to providing a contemporary and efficient learning environment.

By choosing a gradual transition strategy, Elmwood University ensures that the adoption of this system is a well-planned and manageable process. The extensive training provided to staff, faculty, and students demonstrates the institution's dedication to a smooth and successful integration of the new system. The outcomes, as detailed in the document, indicate significant improvements in efficiency, data accuracy, accessibility, and security, validating the system's relevance within the educational landscape and the institution itself.

Product Features

This section of the documentation delves into the essential features of the Student Database Program created for Elmwood University. These features are strategically designed to address the shortcomings of the traditional paper-based record-keeping system, offering a comprehensive solution that aligns with Elmwood University's challenges.

1. **Login/Logout System with Role-Based Access Control**
    - This system allows users to log in and log out of a digital platform or application.
    - It employs role-based access control (RBAC) to determine the user's role (staff, faculty, or student) based on their login information and directs them to the appropriate sections or features of the system accordingly.
2. **Student Enrollment Portal**
    - Digital platform that facilitates the enrollment process for both existing and new students.
    - It typically allows students to register for courses, update personal information, and complete necessary enrollment tasks.
3. **Grade Management System**
    - Digital tool used by educational institutions to record, calculate, and manage students' grades and academic performance.
    - It helps instructors and administrators track student progress and generate reports on grades and assessments.
4. **Student Record Search and Reporting System**
    - This system provides quick search capabilities for accessing individual student records, and it also includes batch processes for generating class or course reports.
    - It helps streamline administrative tasks related to student data management and reporting.

Operating Environment

Elmwood University's new digital student database system is C++-driven and operates in a stable technical environment built for efficiency and security. The system runs on servers with a stable and secure operating system, as well as hardware requirements adapted to a diversified 20,000-student community. This shall contain enough space to store student information, academic records, and administrative data.

Design and Implementation Constraints

- **Time Constraint:** The programmers were given a short period of time to create a database system for Elmwood University. Additionally, 5 features were proposed to be implemented in the system. 3 weeks to create an efficient database system may not be enough for the team.
- **Programming Language Constraint:** The required programming language to implement the system is C++. Although the programmers have a basic knowledge of the chosen language, they may encounter problems in creating the system as they have not yet mastered C++.
- **User Adaptation:** Switching from a paper-based student record system to a database system may be hard for some students, teaching, and non-teaching personnel. It may take some time for the target users to familiarize themselves with the new system.
- **Data Privacy:** Switching from a paper-based system to a digital system may cause data loss or data corruption. The system will hold the personal information of all the students of Elmwood University which may raise a legal constraint that the team and the university must adhere to.
- **Scalability:** There is a possibility that Elmwood University will continue to accept new students which may cause a constraint to the system.
- **Lack of Specification:** Due to a lack of specific information provided by the client regarding the courses they teach and their grading system, we may encounter challenges in ensuring that the courses listed in our program accurately align with the client's curriculum and grading criteria.
- **Limited Client Interaction**: Given the absence of direct client interactions, our team is currently facing difficulties in obtaining real-time feedback and engaging in a more hands-on collaboration with clients.

Assumptions and Dependencies

- **Data Availability**: The necessary student information, including personal information, enrollment records, and academic performance data, will be available and accurate for migration into the student database system. Any discrepancies or missing data may impact the system's functionality.
- **User Training**: Assuming that Elmwood University will provide adequate training and resources for staff, faculty, and students to effectively use the digital student database system. The successful adoption of the system relies on users' proficiency.
- **Infrastructure Availability**: Assuming that the planned hardware, software, and network infrastructure required to support the system will be procured and maintained as per the project schedule.
- **Third-Party Services**: Any third-party services or integrations required for the system, such as external authentication services or payment gateways, will remain accessible and functional throughout the project's duration.
- **Compliance with Regulations**: Assuming that the project will adhere to all relevant legal and regulatory requirements, including data privacy laws and educational standards. Failure to meet compliance standards may result in legal and operational challenges.
- **Data Migration:** The successful implementation of the digital student database system depends on the accurate and timely migration of existing paper-based student records into the new system. Any delays or errors in this process could affect system functionality.
- **Hardware Procurement:** The project is dependent on the timely procurement and installation of the necessary hardware components, including servers and networking equipment, to support the system's operation.
- **Software Dependencies:** The system relies on specific software components, including the chosen database management system and web server software. The availability and compatibility of these software components are critical to the system's functionality.
- **External Services:** Any third-party services or APIs integrated with the system, such as authentication services or payment gateways, are dependencies. The stability and reliability of these services are essential for the system's operation.
- **Regulatory Changes:** The project is dependent on maintaining compliance with evolving legal and regulatory requirements. Any changes in regulations may necessitate updates to the system to ensure ongoing compliance.

III. System Features and Requirements

Functional Requirements			
Features	**Inputs**	**Processing**	**Output**
Main Menu	- Enter the option (enrollment page and login page)	- Redirect users based on the option selected	- Can end up in Enrollment, Login, or Exit the program
Enrollment Menu	- Fill-out information - Upload Documents - Submit Application - Go back to Main Menu	-Submits the information of the applicant -Submit documents	-Saves the applicant information and document
Faculty Menu	- Input Student Grades - Search the Student Record -Generate Class Report	- Searches for the section entered by the user - Determines if the section exists or not - Displays the list of sections	- Confirms student grades before updating the system - Save recent transactions
Admin Menu	- Create user - Remove user - Create course - Find existing course/ section - Create Section - Add faculty to section - Add student to section - Logout	- Create a new user with provided details - Removes the user based on the email - Retrieves information based on the user input - Searches for existing course by the Course ID - Determines if the section exists and find the student/faculty by their ID	- Confirmation of the recent transaction - Updates and adds to the system
Student Menu	- Submit the documents of the student - See information - Logout	- Retrieve documents - Retrieve the information of the student	- Display the Student Information - Save submitted documents
Staff Menu	-Display Applicants -Search Student Records -Generate Class/Course Report	-Retrieve Applicants -Retrieve Student from Records -Retrieve Class/Course Report	- Display the Applicants pending for approval -Display -Display the generated Class/Course Report

Login Menu	-Enter email and password	-Checks the for an account with matching email and password	- Redirects user based on their email
Create User Menu	-Create a new user by entering the user type and the credentials of the user.	- Retrieves user information and credentials	- Saves the user credentials and adds them to the system

Non-Functional Requirements

The C++ Student Database System is designed to deliver outstanding non-functional qualities to meet the university's objectives effectively:

1. **Performance Metrics:**
    - The system will exhibit remarkable performance improvements, with enrollment and grade submissions becoming 80% faster than the previous processes.These performance enhancements will result in a significant reduction in administrative workload, improving overall efficiency.
2. **Data Accuracy:**
    - The system will boost data accuracy, leading to a remarkable 95% reduction in data conflicts.This ensures the maintenance of accurate and consistent student records, critical for academic operations.
3. **Efficiency and Quick Data Operations:**
    - The system will feature highly efficient data entry and retrieval procedures, contributing to streamlined administrative tasks. Centralized data storage will further enhance efficiency, allowing quick access to relevant information.
4. **Security Measures:**
    - Rigorous security safeguards have been implemented to protect sensitive student data from breaches and loss. Role-based access control ensures that only authorized personnel have access to sensitive data, enhancing data security.
5. **Data Recovery:**
    - An automated backup process is in place to safeguard data integrity.In the event of system malfunctions or data loss, this process will ensure the swift recovery of all stored data, minimizing disruptions.

These non-functional requirements underscore the system's commitment to improving performance, data accuracy, efficiency, security, and data recovery capabilities, aligning with the university's goals and ensuring a robust and reliable software solution.

User Interface Requirements

The digital application at Elmwood University has significantly improved the user experience for students, teachers, and administration. This program has introduced a user-friendly Digital Enrollment Portal that allows students to enter their personal information and submit required documents conveniently from their homes. Faculty members have gained the ability to directly input students' grades into the system, which are then reflected in the database.

Enhanced data security is ensured through role-based access control, limiting students to view-only access for their grades. Additionally, the application incorporates wireframed automatic backup and recovery mechanisms to maintain data integrity and system resilience. These transformative features have reshaped how users engage with academic records and administrative tasks, resulting in increased operational efficiency, data integrity, accessibility, and security for Elmwood. This, in turn, has enriched students' academic journeys and fostered administrative excellence.

Main Menu

1. **User Authentication:**
    - Users must provide valid credentials (e.g., email and password) to access the system.
    - Failed login attempts should display an error message.
    - After successful login, users are redirected to specific menus based on their roles (student, faculty, staff, admin).

Student Menu

1. **View Personal Information:**
    - When the student selects this option, the system displays the following information:
        - Student ID
        - Full Name (First Name, Middle Name, Last Name)
        - Email Address
        - Enrollment Date

- Any other relevant personal details

2. **View Enrolled Courses:**

    - When the student selects this option, the system retrieves and displays a list of courses in which the student is enrolled.
    - The list includes information such as Course ID, Course Name, and Section Name.

3. **View Grades:**

    - When the student selects this option, the system retrieves and displays the student's grades for each enrolled course.
    - The grades should include details such as Course ID, Course Name, Section Name, and the respective grade obtained.
    - Students cannot modify their grades.

4. **Logout:**

    - When the student selects this option, they are logged out of the system and returned to the main menu.

Staff Menu

1. **Applicant List Display:**

    - Staff can view a list of applicants for enrollment.
    - The list includes relevant information about each applicant.
    - Staff can enter the Applicant ID to approve an applicant.
    - Upon approval, a success message is displayed.
    - If the applicant is not found, an error message is displayed.

2. **Student Search:**

    - Staff can search for student records by entering the student's email.
    - The system displays student information, including Student ID, Full Name, and enrolled sections.
    - Staff can view the grade for each section in which the student is enrolled.

3. **Generate Class Report**

    - Section Selection:

        - Staff can enter the Section ID to generate a class report.If the section is found, a class report is displayed.

    - Course and Section Selection:

        - Staff can enter the Course ID to find a course.

        - If the course is found, its details are displayed.

        - Staff can enter the Section ID to which they want to assign a faculty member.

4. **Faculty Assignment:**

    - Staff can enter the faculty's email to assign them to the selected section.

    - Upon assignment, a success message is displayed.

    - If the faculty or section is not found, an error message is displayed.

Admin Menu

1. **Create User:**

    - Admin can select the type of user to create (Student, Faculty, Staff, Admin).

    - Admin provides user details such as First Name, Middle Name, Last Name, Email, and Password.

    - User details must be validated, and any errors should be displayed.

    - Upon successful creation of a user, a success message is displayed.

2. **Remove User:**

    - Admin can search for a user by entering the user's email.

    - Admin is prompted to confirm the removal of the selected user.

    - Confirmation should be entered as "Y" or "N."

    - If confirmed, the user is removed, and a success message is displayed.

3. **Create Course:**

    - Admin enters the Course Name to create a new course.

    - Upon successful creation of a course, a success message is displayed.

4. **Find Existing Course/Section:**

- Admin can search for existing courses or sections by entering the Course/Section ID.
- If found, details of the course or section are displayed, including Course Name and enrolled sections.

5. **Create Section:**
    - Admin can enter the Course ID to associate a new section with an existing course.
        - Admin enters the Section Name for the new section.
        - Upon successful creation of a section, a success message is displayed.
6. **Add Faculty/Student to Section:**
    - Admin can enter the Course ID to find a course.
    - If the course is found, its details are displayed.
    - Admin can enter the Section ID to which they want to assign a faculty member or student.
    - Admin enters the email or ID of the faculty or student to assign them to the selected section.
    - Upon assignment, a success message is displayed.
    - If the faculty or student is not found, an error message is displayed.

Enrollment Menu

1. **Enrollment Page:**
    - The enrollment page displays a form where users can fill out their enrollment information.
    - Users are required to enter the following information:
        - First Name
        - Middle Name
        - Last Name
    - The entered information is displayed on the screen.
2. **Fill-out Information:**
    - Users can select the "Fill-out Information" option (choice 1).

- Upon selection, the user is prompted to enter their first name, middle name, and last name.
- Users can enter their information, and the entered values are displayed on the screen.

3. **Upload Documents:**
    - Users can select the "Upload Documents" option (choice 2) to upload documents.
    - The code provides a confirmation prompt, allowing users to decide whether they want to proceed with document uploading.
    - If confirmed, the code simulates the uploading process and stores timestamps of the uploaded documents.
4. **Submit Application:**
    - Users can select the "Submit Application" option (choice 3) to submit their enrollment application.
    - Upon selection, the system creates a new applicant with the provided first name, middle name, and last name.
    - A success message is displayed, indicating that the application has been successfully submitted.
5. **Back to Main Menu:**
    - Users can select the "Back to Main Menu" option (choice 4) to return to the main menu.
    - Upon selection, the enrollment process is canceled, and the user is taken back to the main menu.
6. **Input Validation:**
    - The system should perform input validation to ensure that the user provides valid information for first name, middle name, and last name.
    - If any of the fields are left empty, the system should display an error message and prompt the user to enter the missing information.

Clear Console and Pause

- The application provides a mechanism to clear the console screen for a cleaner interface.
- After certain actions or messages, users are prompted to press a key (Enter or any key) to continue.

Hardware and Software Requirements

To run the C++ student database program on computers and laptops, you will need the following:

- Operating Systems (Windows, macOS, or Linux distribution)
- C++ compiler (e.g. GCC)
- RAM (4GB or more)
- Storage Space (SSD)

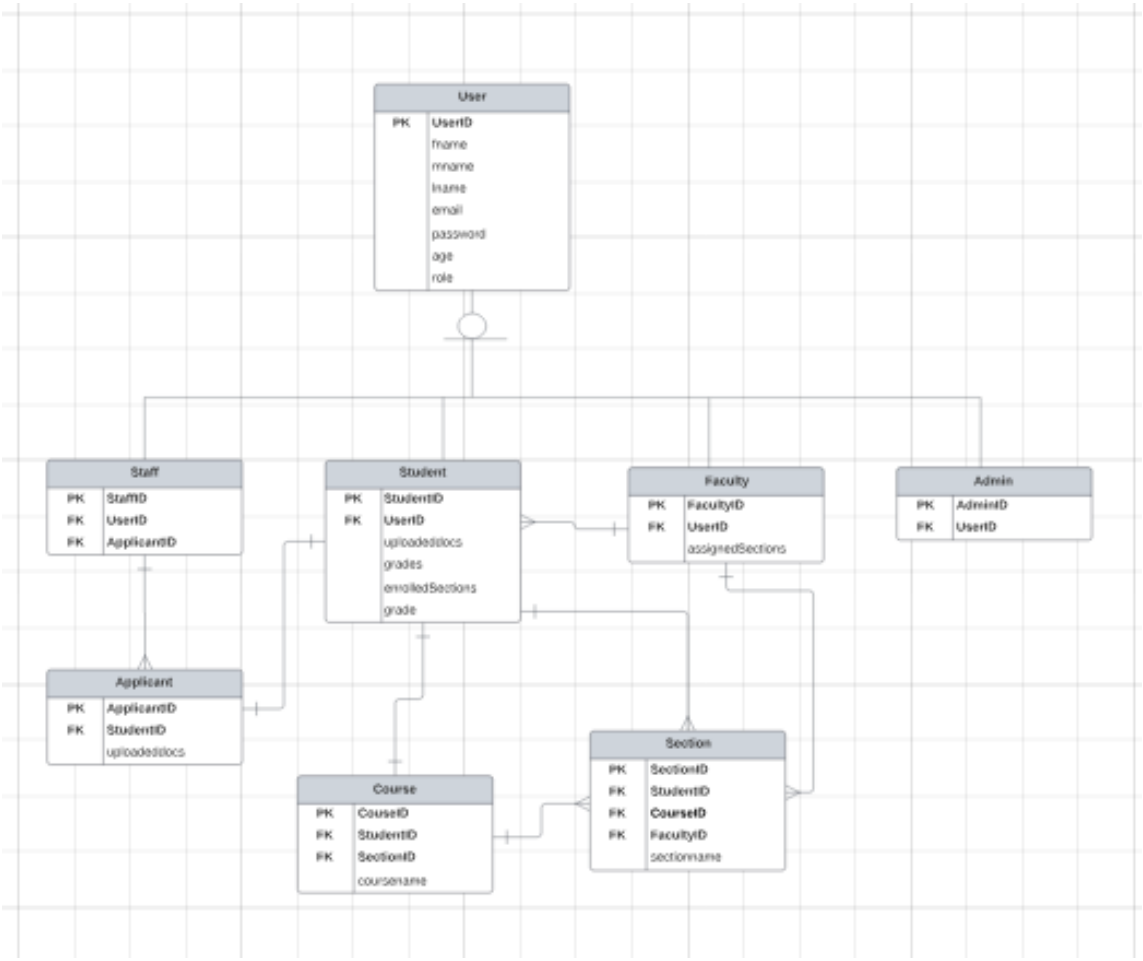
These requirements ensure smooth operation and efficient management of student records, providing a seamless experience for users.

Data Management or Database Requirements

Elmwood University is modernizing its data management operations by implementing a digital student database system designed with the powerful capabilities of C++ programming, specifically file-handling techniques for data storage. This comprehensive system aims to efficiently concentrate and manage the enormous and diverse data sets of around 20,000 undergraduate, graduate, and doctorate students. The users may easily find their relevant information by searching their own information, like student ID or name, allowing for faster and more accurate access to student records.

IV. System Models

Entity Relationship Diagrams



V. Testing

Feature: Enrolling as a New Student

Preconditions:

1. The user is a new student at Elmwood University.

Test Steps:

1. Navigate to the "Enrollment Page".
2. Choose fill out information.
3. Input needed information.
4. Click submit application to save data.

Expected Results:

1. After submitting application, the user information will be stored in applicants.json

Postconditions:

1. The applicant ID must increment by 1.
2. The user must wait for the application to be approved before they can log-in to the system.

Variations:

1. If the user enters the wrong details they can choose to fill out information again to override the mistake they made.

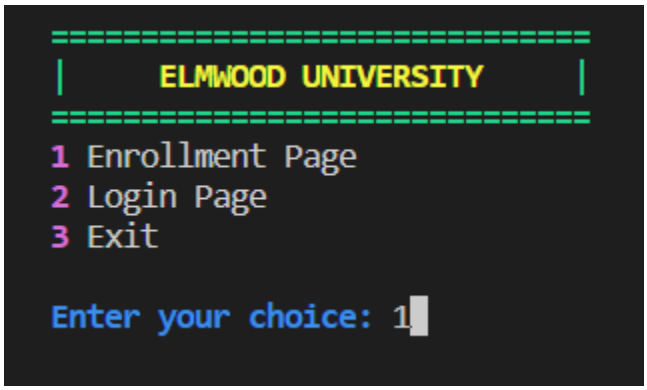


Figure 1.1 Navigate to "Enrollment Page"

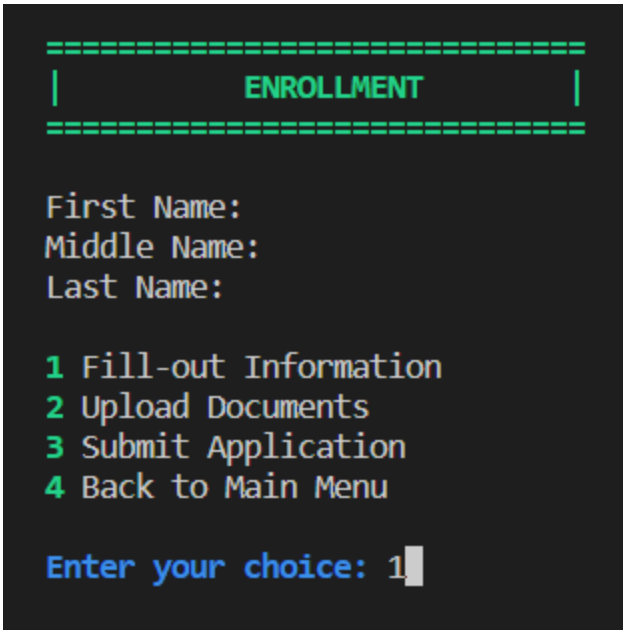


Figure 1.2 Choose "Fill-out Information"

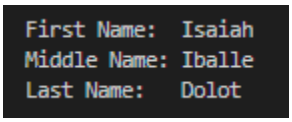


Figure 1.3 Input needed information

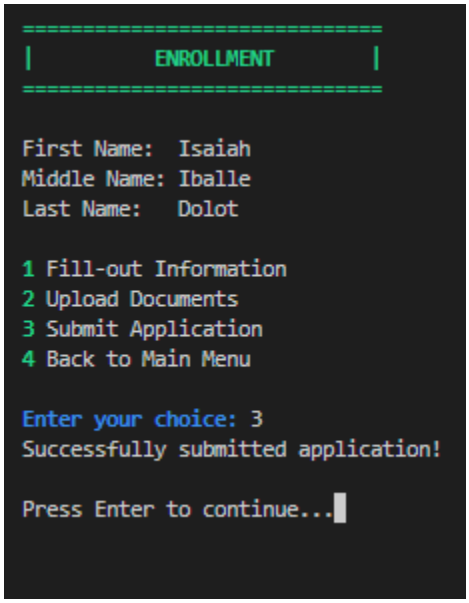


Figure 1.4 Click submit application to save data.

Feature: Encoding Grades as a Faculty Member

Preconditions:

1. The user is a professor at Elmwood University.
2. The user has already existing information stored in the system.

Test Steps:

1. Navigate to the "Login Page".
2. Input email and password.
3. Choose to input grades
4. Select a section from the assigned sections list.
5. Input grade.

Expected Results:

1. After encoding the grade should be stored in the students.json file.
2. Students will be able to view their grades.

Postconditions:

1. The grade should appear when the professor choose to generate class report.

Variations:

1. Choose Generate Class Report without inputting the grade first. No grades shall be displayed.

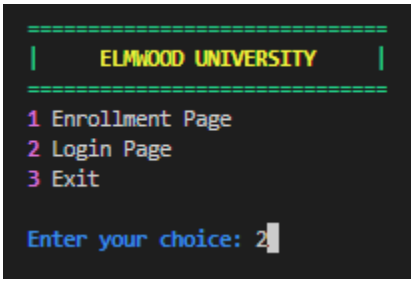


Figure 2.1: Navigate to "Login Page"

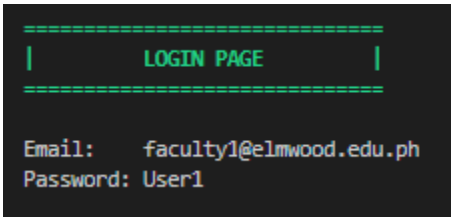


Figure 2.2 Input email and password

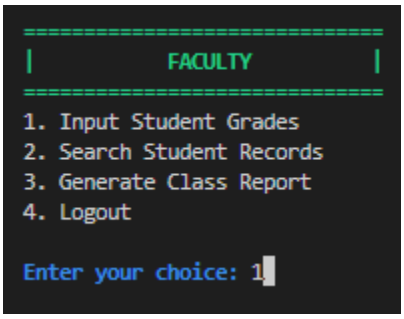


Figure 2.3 Choose to input grades

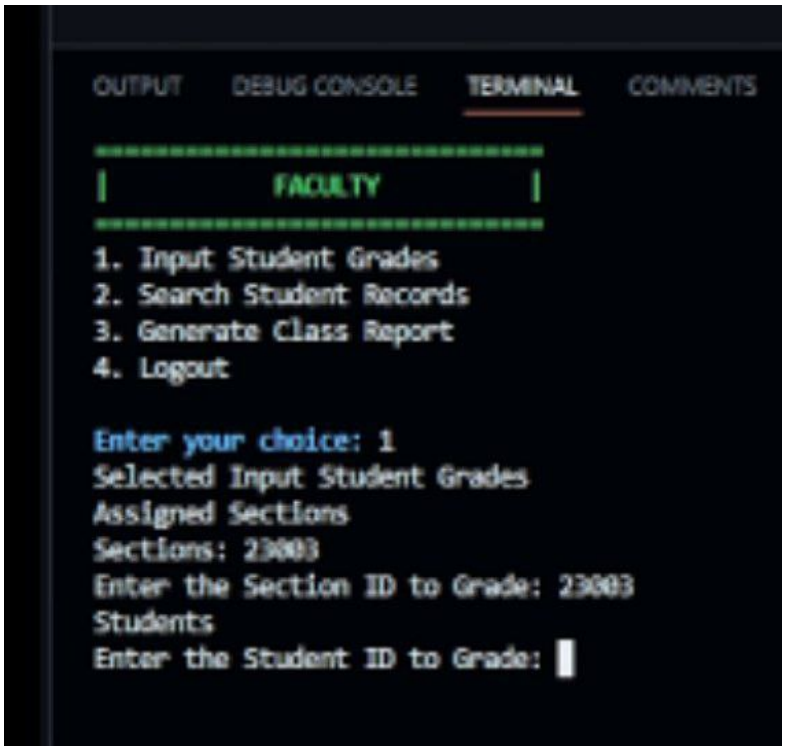


Figure 2.4 Select section

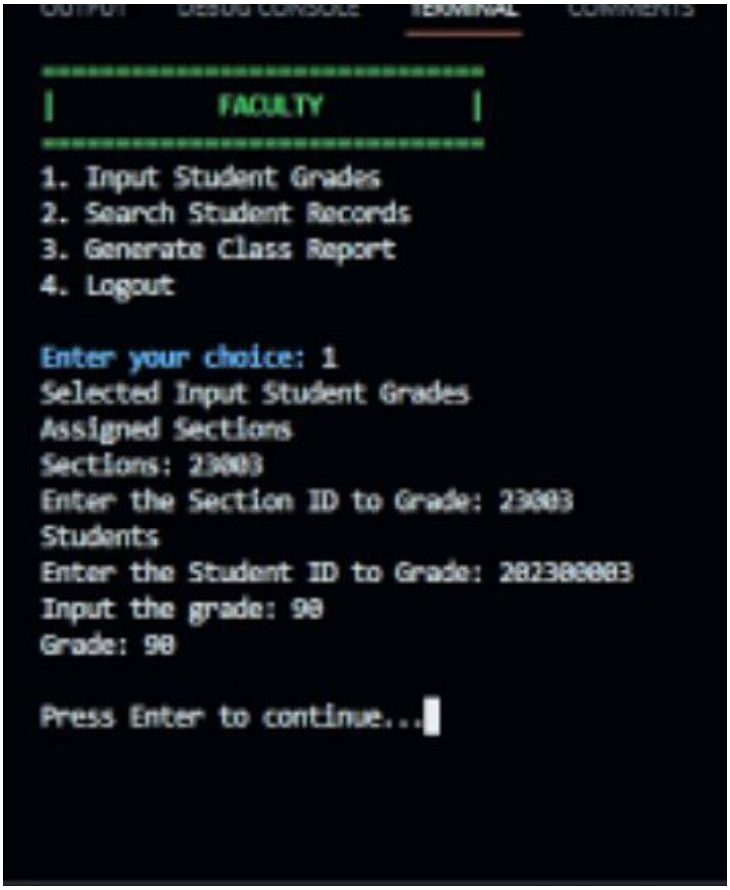


Figure 2.5 Input grades

Feature: Creating a New Course

Preconditions:

1. The user is part of the administration at Elmwood University.
2. The user has already existing information stored in the system.
3. The user has already logged in to the system.

Test Steps:

1. Choose “Create a Course”
2. Enter Course Name
3. Go back to the Admin menu and Logout.

Expected Results:

1. Course shall be stored in courses.json

Postconditions:

1. CourseID shall increment by 1.

Variations:

1. Choose to find an existing course instead of creating: Enter courseID to check if course exists. If it exists, the course name and related section will be displayed.

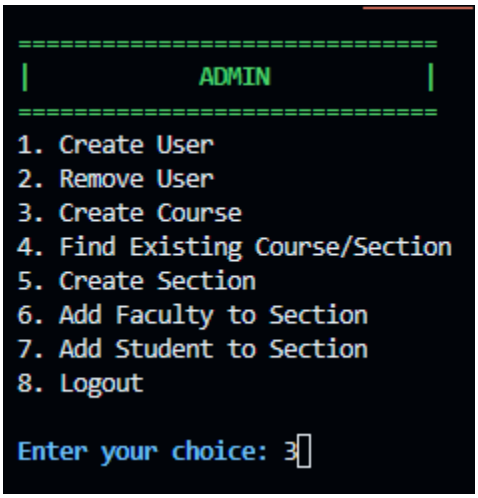


Figure 4.1 Choose Create Course

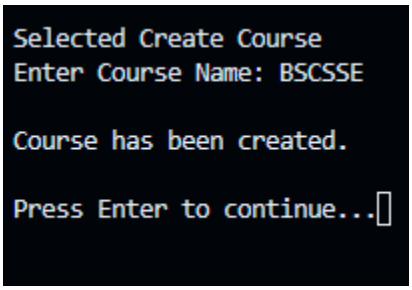


Figure 3.2 Enter Course Name

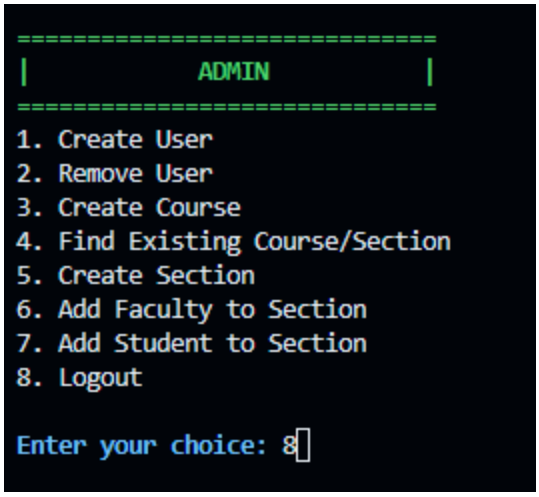


Figure 3.3 Logout

Feature: Displaying List of Applicants

Preconditions:

1. The user is a staff member at Elmwood University.
2. The user has already existing information stored in the system.
3. The user has already logged in to the system.

Test Steps:

1. Choose “Display Applicants”
2. Input Applicant ID.
3. Go back to the Staff menu and Logout.

Expected Results:

1. The applicant that was chosen will be moved to students.json file.
2. Their information will be removed from the applicants.json file.

Postconditions:

1. StudentID shall increment by 1.
2. ApplicantID of the approved student must not be used again by other users.

Variations:

1. **Search Student Records:** Instead of choosing display applicants, try to test another feature from the staff menu

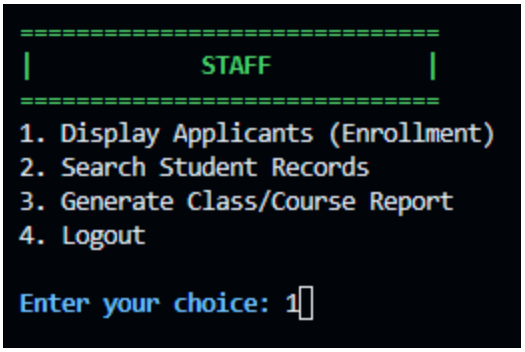


Figure 4.1 Choose "Display Applicants"

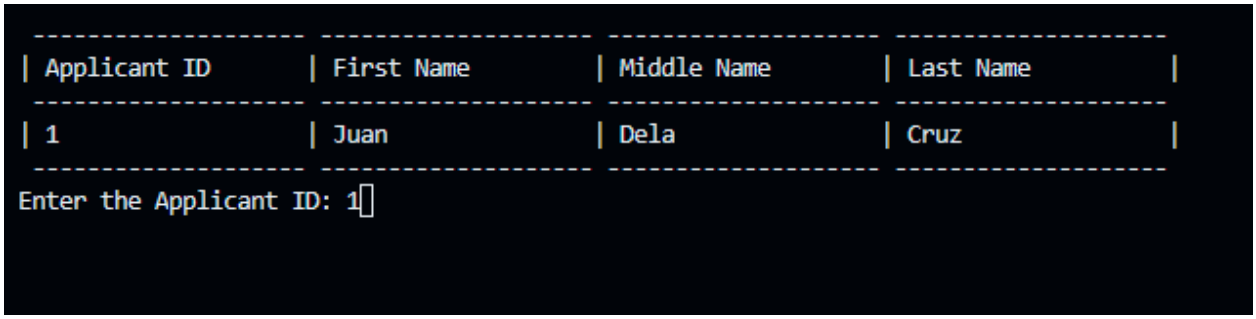


Figure 4.2 Input applicant ID

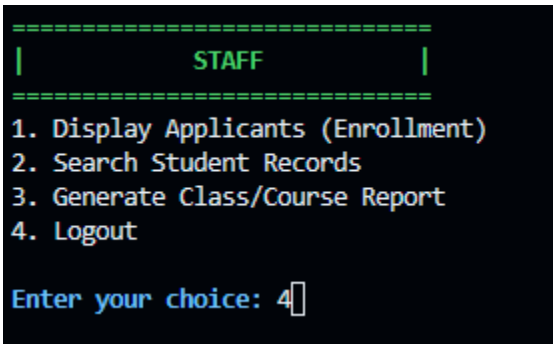


Figure 4.3 Logout