Project Report

Project: **StudentBuddy**

Only for course Teacher						
		Needs Improvement	Developing	Sufficient	Above Average	Total Mark
Allocate mark & Percentage		**25%**	**50%**	**75%**	**100%**	**25**
Understanding	**3**					
Analysis	**4**					
Implementation	**8**					
Report Writing	**10**					
					Total obtained mark	
Comments						

Semester: Spring …….. / Fall 2024

Group: 01

Names: UTHSOB CHAKRABORTY, Md. Abdullah Saeem, Utsha Dey

Student ID's: 232-35-621, 232-35-648, 232-35-459

Batch: 41 **Section:** K2

Course Code: SE133 Course Name: Software Development Capstone Project

Course Teacher Name: Maidul Islam

Designation: Lecturer

Submission Date: 30 /11/2024

Table of Contents

Chapter 1 (Introduction)

StudentBuddy is a multi-functional CLI-based application designed to assist students with their academic productivity. It integrates a **Note Manager** for managing class notes and a **Schedule Manager** for organizing schedules and detecting empty time slots. The system supports multi-user functionality, ensuring secure and private access for each user through a robust account-based system.

1.1 System Overview

The **StudentBuddy** system is a CLI-based productivity tool designed for students, combining robust user authentication with functional modules for managing notes and schedules. This section provides an overview of the system, its components, and how they interact to deliver an intuitive and secure experience.

Key Components

1. **User Management System**: Handles user registration and authentication, ensuring secure access to the application.
2. **Note Manager**: Enables users to manage their academic or personal notes effectively.
3. **Schedule Manager**: Helps users organize and manage their academic schedules.

1.2 Features

1. **User Account Management**:
    - Secure and unique user authentication with username and password.
    - Automatically generated unique account numbers for users.
    - Persistent storage of user data in a file.
2. **Note Management**:
    - Add, edit, delete, and list notes specific to each user.
    - Save and retrieve notes from persistent storage.
3. **Schedule Management**:
    - Add, view, and delete class schedules.
    - Detect empty time slots for effective time utilization.
    - Persistent storage of schedules.

1.3 Why are we developing this?

We are developing the **StudentBuddy** system to address the common challenges students face in managing their academic tasks effectively. Many students struggle with organizing notes, planning schedules, and making the best use of their time, often leading to stress, missed deadlines, and inefficient study habits. By integrating features like personalized note management and the innovative Empty Class Slot Detector, this system empowers students to identify free time, plan their schedules efficiently, and stay on top of their academic responsibilities. The goal is to provide a user-friendly, secure, and productive tool that enhances time management, reduces stress, and helps students achieve their academic and personal goals seamlessly.

Chapter 2 (Stakeholders)

2.1 What is Stakeholder?

Stakeholders in a software system are individuals, groups, or organizations that are impacted by the system's development, use, or existence. They have a vested interest in the system's success and can influence its requirements, design, and functionality.

2.2 Who are the Stakeholders for this system?

1. Primary Stakeholders

These are individuals or groups directly involved with or benefiting from the project:

1. **Students**
    - **Role:** End-users of the tool.
    - **Interest:**
        - Efficient management of notes and schedules.
        - Personalized features for improved productivity.
    - **Needs:**
        - A user-friendly, reliable, and secure tool.
        - Features that cater to academic requirements like time management, note-taking, and class scheduling.
2. **Developers**
    - **Role:** Creators and maintainers of the tool.
    - **Interest:**
        - Building a functional, modular, and scalable application.
        - Continuously improving the tool to match user expectations.
    - **Needs:**
        - Clear requirements and feedback from end-users.
        - Proper version control, documentation, and testing resources.

---

2. Secondary Stakeholders

These are individuals or entities indirectly benefiting from the project:

1. **Educational Institutions**
    - **Role:** Promoters or adopters of the tool.
    - **Interest:**
        - Encouraging students to use tools for better academic organization.
        - Reducing dependency on paper-based schedules and notes.
    - **Needs:**
        - Access to features that can be recommended to students.
        - Scalability for institution-wide adoption if needed.
2. **Parents and Guardians**
    - **Role:** Supporters or influencers in students' productivity.
    - **Interest:**
        - Ensuring students stay organized and focused on academics.
    - **Needs:**
        - Assurance that the tool is effective and secure.

Chapter 3 (Requirement Analysis)

3.1 What is Requirement Analysis?

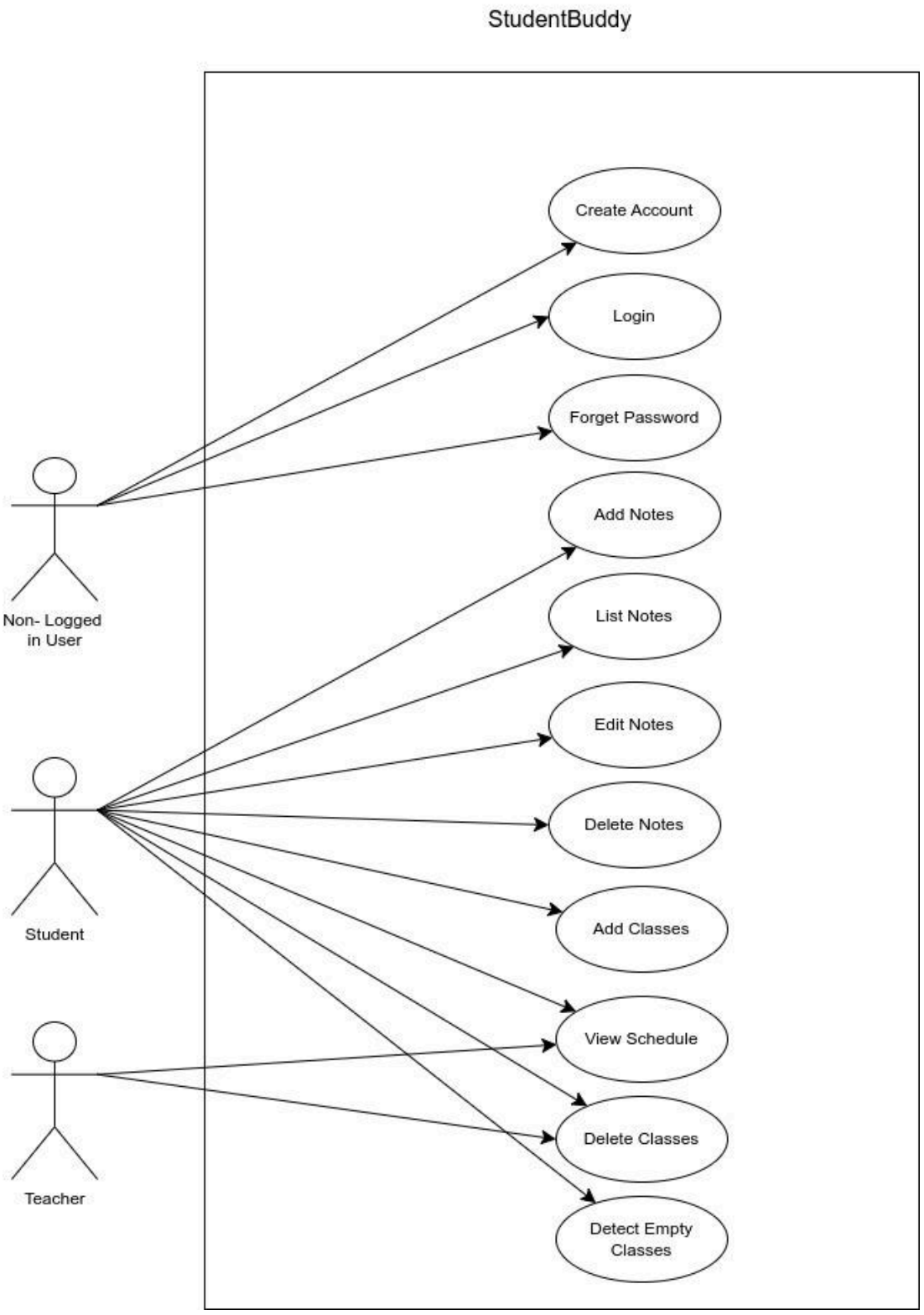
Requirements Analysis is a fundamental stage in software development that focuses on determining the needs, expectations, and conditions that a new or modified product or system should fulfill. This process considers the often conflicting requirements of various stakeholders involved in the project.

3.2 Functional Requirement Table:

FR	Feature	Description	Stakeholder
FR 01	Create an account	A non-logged user must create an account to use the system, such as accessing personalized features like note management, scheduling, and tracking. After creating an account, they are considered a registered user.	Non-logged User
FR 02	Login	A registered user can log in using their username and password. Upon logging in, they gain access to their personal data, including notes and schedules. Login ensures secure and user-specific operations.	User
FR 03	Forget Password	The user can reset their password if they forget it by modifying user.txt	User
FR 04	Add Notes	Users can create and save new notes. Each note is linked to the user's account number for personalization and secure access.	User
FR 05	List Notes	Users can view all notes they have saved. Notes are displayed in chronological order or by category to ensure quick access.	User
FR 06	Edit Notes	Users can modify existing notes by notes id to ensure information stays up-to-date and relevant.	User
FR 07	Delete Notes	Users can permanently remove notes they no longer need.	User

FR 08	Add Classes	Users can add class schedules to their personalized timetable. This helps users keep track of their academic or other schedules efficiently.	User
FR 09	View Schedule	Users can view their entire schedule, showing all planned classes or events in a clear and accessible format.	User
FR 10	Delete Classes	Users can remove specific class entries from their schedule when they are no longer needed.	User
FR 11	Detect Empty Slots	The system scans the user's schedule to identify free time slots, helping them plan and utilize their time effectively.	User
FR 12	Save User Data	The system automatically saves user account data, notes, and schedules in external files (users.txt, notes/, schedule/). This ensures all user information is persistent across sessions.	System

Chapter 4 (Use Case Diagram)



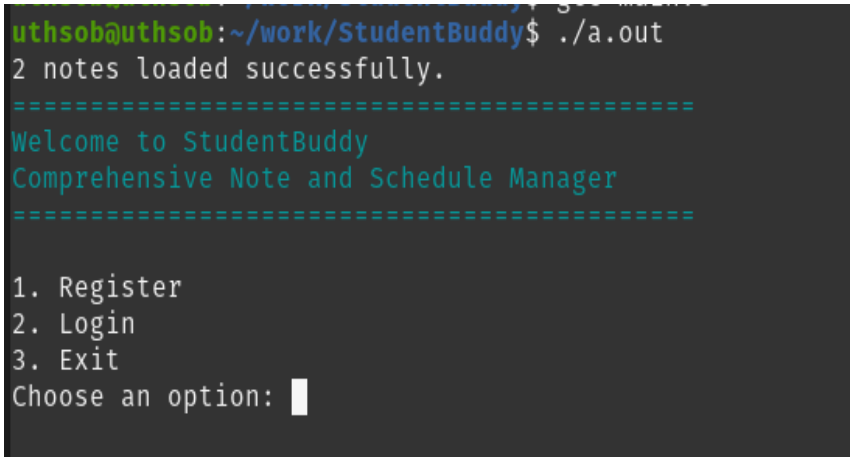
Chapter 5 (Feature Description)

5.1 Feature Description for Non-logged Users

A list of features for Non-Logged Users:

View Welcome Page:

- Non-logged users can access a welcome interface, which includes a brief introduction to the system.



Register:

- Non-logged users can create an account by providing a username and password.
- After registration, they become logged users and gain access to personalized features.

Login:

- Non-logged users can login to the system with their account credentials.

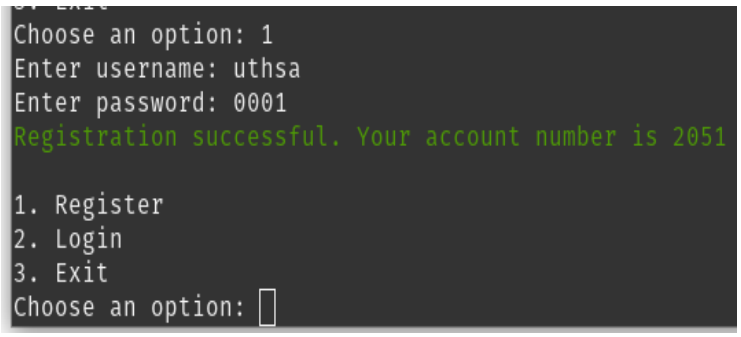
Exit:

- Non-logged users can safely exit the system through a dedicated option.

5.1.2 Feature Showcase for Non-logged Users (Register)

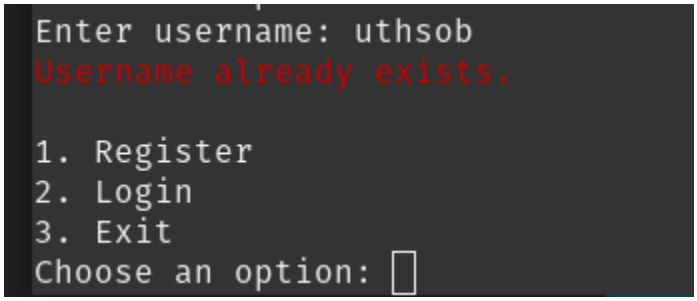
5.1.2.1 Successful registration:

- If a user gives a valid username then with a password it generates an account for the user with a unique account number; This unique account number is used to fetch all data for individual users.



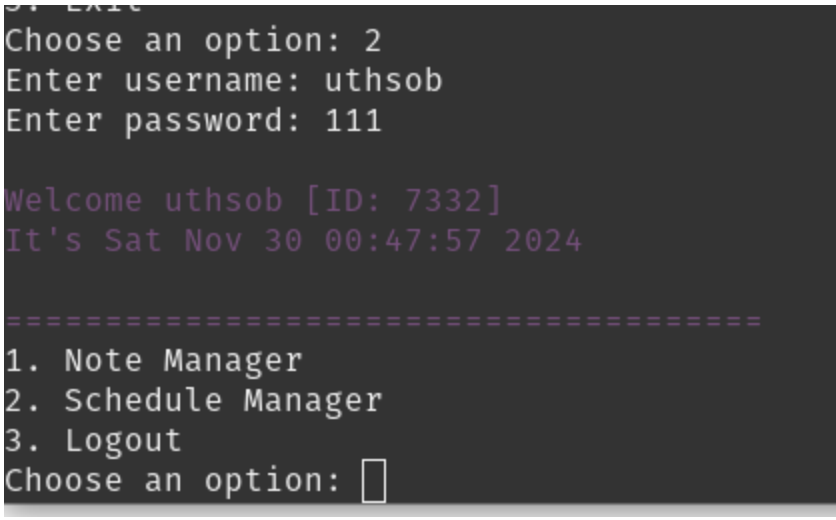
5.1.2.2 Failed Registration

- If a user provides an existing username then it will show an error message and won't register a new account.

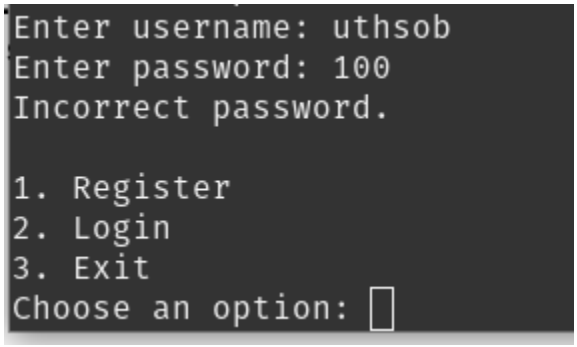


5.1.3 Feature Showcase for Non-logged Users (Successful Login):

- Users can login with their existing account username and password.
- It will be well welcome with <Username> <[ID:]> and current time in 24H format.



5.1.4 Feature Showcase for Non-logged Users (Failed Login):

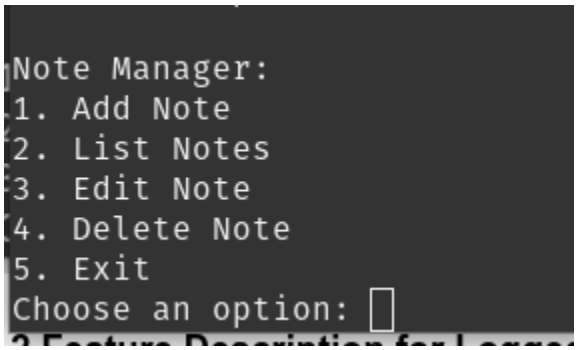


5.2 Feature Description for Logged Users

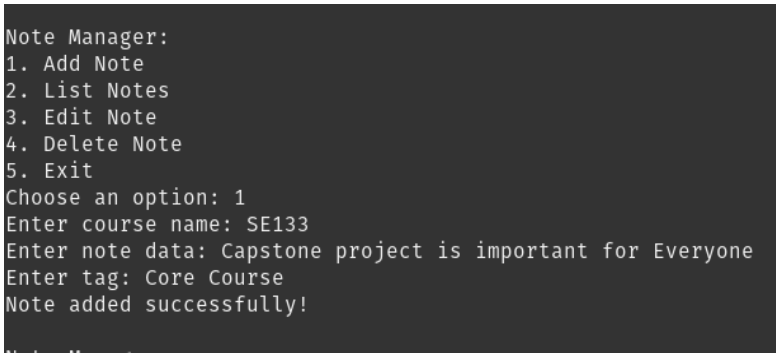
5.2.1 Feature Description for Logged Users (Welcome Screen)

- Users will be able to see and use Note Manager, Schedule Manager and logout from the main screen.

5.3 Feature Description - Note Manager



5.3.1 Feature Description - Note Manager (Add Note)



5.3.1 Feature Description - Note Manager (List Notes)

- User specific notes affiliated with account number
- Gets and uses date automatically
- Unique individual id for each notes

```
Listing notes for account: 7332
========================================
ID: 1
Course: Bil
Date: 2024-11-25
Note: Something is better?
Tag: anything
========================================
ID: 2
Course: Something
Date: 2024-11-25
Note: Not something
Tag: Maybe
========================================
ID: 3
Course: SE133
Date: 2024-11-30
Note: Capstone project is important for Everyone
Tag: Core
========================================
```

5.3.1 Feature Description - Note Manager (Edit Note):

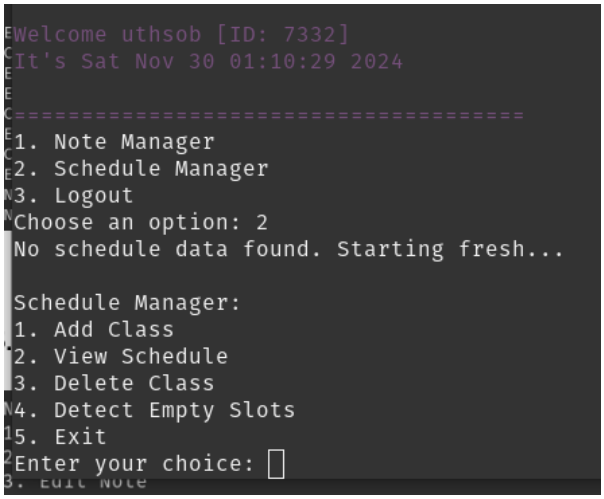
```
Choose an option: 3
Enter the ID of the note you want to edit: 1

Editing Note ID: 1
Current Course: Bil
Enter new Course (or press Enter to keep current): Current Date: 2024-11-25
Enter new Date (YYYY-MM-DD) (or press Enter to keep current):
Current Note Data: Something is better?
Enter new Note Data (or press Enter to keep current): It's important
Current Tag: anything
Enter new Tag (or press Enter to keep current): Core
Notes saved successfully!
Note updated successfully!
```

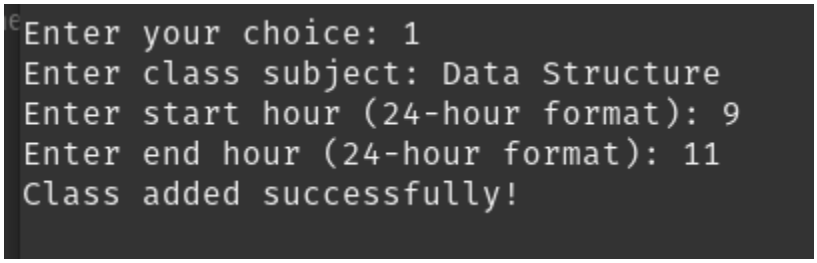
5.3.1 Feature Description - Note Manager (Delete Note):

```
Note Manager:
1. Add Note
2. List Notes
3. Edit Note
4. Delete Note
5. Exit
Choose an option: 4
Enter the ID of the note you want to delete: 2
Note with ID 2 deleted successfully!
```

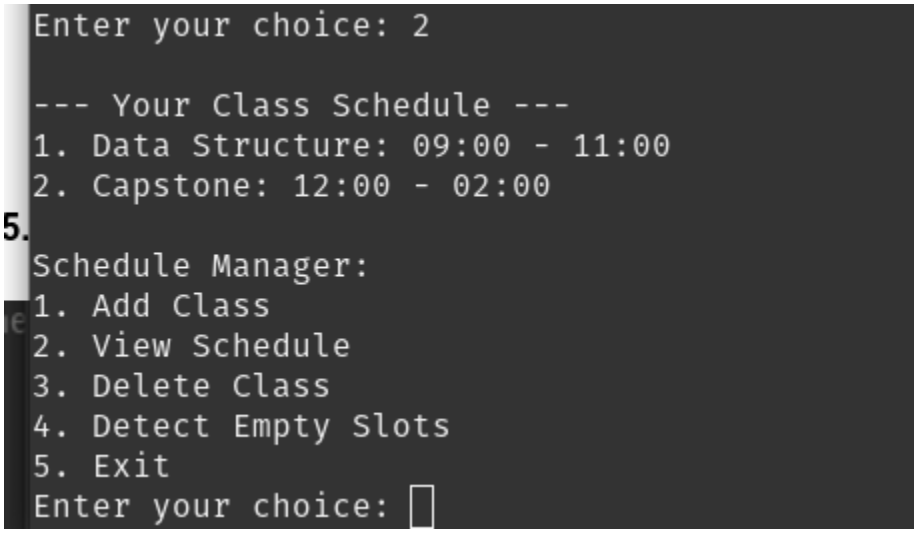
5.4 Feature Description - Schedule Manager:



5.4.1 Feature Description - Schedule Manager (Add Class):

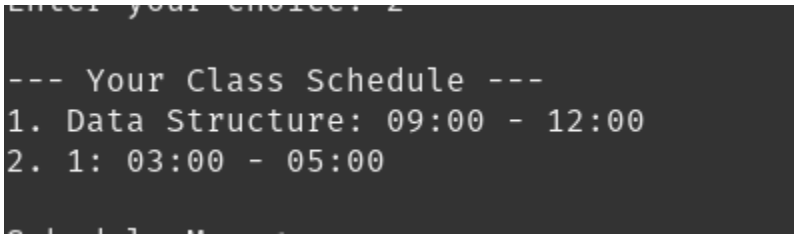


5.4.2 Feature Description - Schedule Manager (View Class):

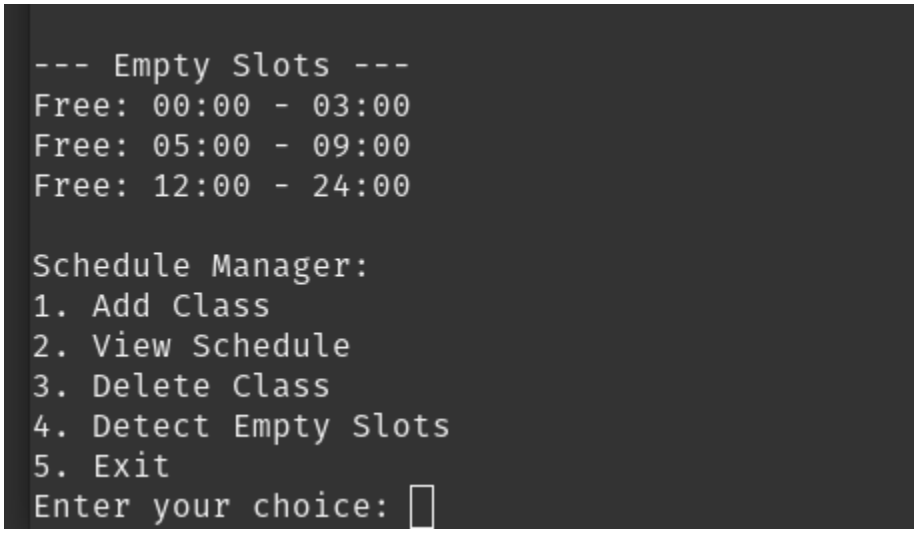


5.

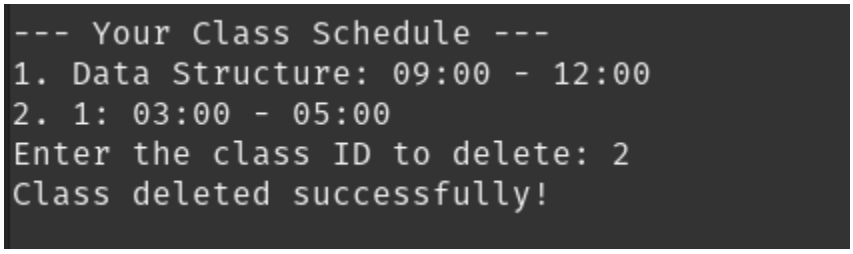
5.4.3 Feature Description - Schedule Manager (View Class):



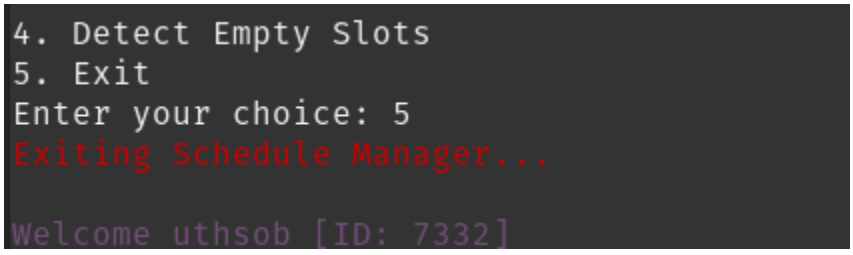
5.4.4 Feature Description - Schedule Manager (Detect Empty Slots):



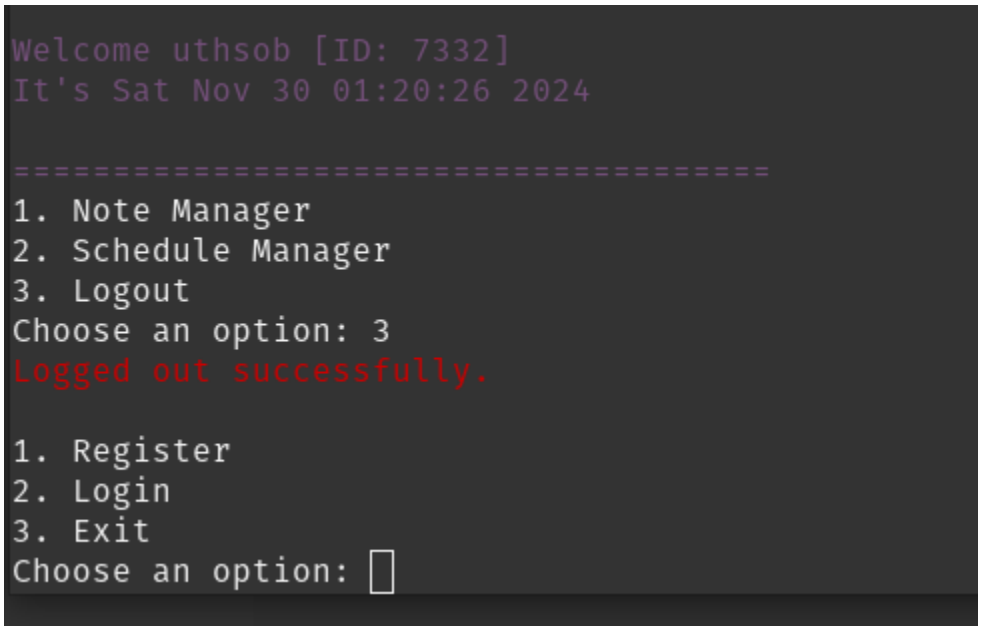
5.4.5 Feature Description - Schedule Manager (Delete Class):



5.4.6 Feature Description - Schedule Manager (Exit):



5.5 Feature Description - Logout



Chapter 6: Testing Analysis for StudentBuddy

6.1 Overview of Testing

Testing is a critical phase in software development to ensure that the system meets functional and non-functional requirements. The StudentBuddy CLI tool underwent rigorous testing to validate its features, performance, and usability. This chapter highlights the testing process, methodologies, outcomes, and key observations.

6.2 Objectives of Testing

1. Functional Validation: Ensure all features work as intended.
2. Non-Functional Evaluation: Assess usability, performance, and reliability.
3. Defect Identification and Resolution: Detect bugs and ensure they are fixed before deployment.
4. User Experience Assurance: Confirm that the system is intuitive and user-friendly.

6.3 Testing Methodologies

1. **Unit Testing:**
   Each module, such as authentication, note management, and schedule management, was tested individually to ensure its correctness.

2. **Integration Testing:**
   Interactions between modules (e.g., linking user accounts to notes and schedules) were validated to ensure smooth data flow.

3. **System Testing:**
   The complete system was tested in a fully integrated environment to verify that all components work together seamlessly.

4. **Regression Testing:**
   Verified that new features or fixes did not introduce bugs in existing functionalities.

5. **User Acceptance Testing (UAT):**
   Simulated real-world usage scenarios to confirm the system meets end-user expectations.

6.4 Testing Outcomes

6.4.1 Functional Testing

- User Management:
    - Account creation, login, and data storage were validated successfully.
    - Password validation and duplicate username checks worked as expected.
- Note Management:
    - Adding, listing, editing, and deleting notes performed flawlessly.
    - User-specific notes were securely stored and retrieved.
- Schedule Management:
    - Adding, viewing, deleting classes, and detecting empty slots worked as intended.
    - Timetable updates dynamically reflected user actions.

6.4.2 Non-Functional Testing

- Performance:
    - The system performed efficiently, with negligible latency during operations.
    - File I/O operations, such as saving and loading user data, were optimized for speed.
- Usability:
    - The CLI interface was intuitive and easy to navigate for users with minimal technical expertise.
    - Color-coded feedback improved user interaction and error understanding.
- Reliability:
    - The system maintained data integrity, ensuring no data was lost or corrupted during testing.
- Security:
    - User authentication successfully restricted unauthorized access to personal data.

6.4.3 Regression Testing

- Ensured that modifications to one module (e.g., adding features to note management) did not negatively impact other functionalities.

6.4.4 User Acceptance Testing

- Users confirmed that the system met their expectations for managing notes and schedules.
- Positive feedback was received on ease of use and intuitive navigation.

6.5 Defects Identified and Resolved

1. **Error Handling:**
    - Issue: Improper feedback for invalid user inputs (e.g., invalid menu options).
    - Resolution: Added meaningful error messages and guided prompts.
2. **File Corruption:**
    - Issue: User data files occasionally corrupted when the application was abruptly terminated.
    - Resolution: Implemented atomic file operations to prevent incomplete writes.
3. **Duplicate Notes:**
    - Issue: Users could accidentally save duplicate notes.
    - Resolution: Added validation to alert users about duplicates before saving.

6.6 Documentation and Reporting

- Test Cases:
  Detailed test cases were created for each feature, including input scenarios and expected outcomes.

- Defect Logs:
  All defects were logged with detailed descriptions, reproduction steps, and resolution statuses.

- Test Reports:
  Comprehensive reports summarized testing activities, including results, identified issues, and resolutions.

6.7 Summary of Testing

The **StudentBuddy** system successfully passed all functional and non-functional tests, demonstrating its readiness for deployment. Key features, such as user authentication, note management, and schedule handling, were validated for accuracy and efficiency. The tool's performance, reliability, and user experience exceeded expectations, ensuring a robust and user-friendly application.

This testing analysis ensures that **StudentBuddy** is reliable, secure, and ready for real-world use. Future monitoring and updates will address any unforeseen issues and adapt to user feedback for continuous improvement.

Chapter 7: Conclusion

7.1 Key Achievements

The StudentBuddy CLI Tool has been successfully developed as a robust and user-friendly application tailored to enhance student productivity. It incorporates essential features such as note management, schedule management, and user authentication. These features work seamlessly to provide a secure, efficient, and intuitive experience for users. The system has passed rigorous testing, ensuring it meets both functional and non-functional requirements, making it a reliable solution for its intended purpose.

7.2 Good Features

The following features contribute significantly to the system's success:

1. Personalized User Experience:
    - Notes and schedules are tied to unique user accounts, ensuring secure and personalized data handling.
2. User-Friendly CLI Interface:
    - The intuitive navigation and color-coded feedback enhance the user experience.
3. Comprehensive Note Management:
    - Features for adding, listing, editing, and deleting notes empower users to organize information effectively.
4. Efficient Schedule Management:
    - Tools to add classes, view schedules, and detect empty slots promote better time management.
5. Modular Design:
    - The system's modular architecture facilitates maintainability and scalability.

7.3 Limitations

Despite its strengths, the current system has some limitations:

1. Plain Text Password Storage:
    - Passwords are stored in plain text, posing a security risk. Future iterations should implement hashing.
2. File-Based Storage:
    - The reliance on text files for data storage limits scalability compared to database systems.

3. Basic Error Handling:
    - Error handling is functional but can be expanded for better user guidance.
4. No Multi-User Support:
    - The system currently supports single-user sessions at a time, restricting real-time collaboration.

7.4 Future Enhancements

To overcome the current limitations and further enhance functionality, the following improvements are proposed:

1. Enhanced Security:
    - Implement password hashing and encryption for better data protection.
2. Database Integration:
    - Transition from file-based storage to a database system for improved scalability and performance.
3. Advanced Features:
    - Include options for exporting notes and schedules as PDFs, setting reminders, and integrating with external calendars.
4. Multi-User Support:
    - Add multi-user capabilities to allow collaboration among students.
5. Mobile Compatibility:
    - Develop a mobile version of the system to extend accessibility.

7.5 Final Remarks

The development of StudentBuddy showcases the successful application of software engineering principles to create a practical and impactful solution for student productivity challenges. While it serves as a robust foundation, the proposed future enhancements will ensure its continued relevance and effectiveness in evolving academic and technological landscapes. This project not only highlights technical proficiency but also reflects a commitment to improving user experiences through thoughtful design and continuous iteration.