Software Requirements Specification

for

The Student Assistant: AI-Enabled Academic and Financial Management

Version 1.0

Sukkur IBA University

Prepared by 20F-33

06 November, 2023

Supervisor Approval

_____

Dr. Asif Khan

Table of Contents

Table of Figures

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to outline the requirements and specifications for the Student Assistant, which is the result of our final year project. This SRS document pertains to Version 1.0. It comprehensively defines the scope of the Student Assistant, encompassing all features and functionalities developed as part of our project. The SRS details the requirements for the entire system, including its constituent subsystems, ensuring a clear understanding of the project's objectives and functionality.

The Student Assistant aims to address the diverse needs of university students and faculty by providing a unified and intelligent platform for academic and financial management. This SRS serves as a critical reference point for all stakeholders involved in the development, ensuring that the final product aligns with the specified requirements and user expectations.

1.2 Document Conventions

Font: The entire document is written in a legible, serif font such as Times New Roman, with a standard font size (14pt) for the main text, 16pt for Heading 2 and 18pt for the Heading 1for readability.

Highlighting: Important terms, titles, and emphasis may be highlighted using bold text.

Section Numbering: Sections and subsections are numbered hierarchically, with each level represented by a number (e.g., 1, 1.1, 1.2, 2, 2.1, 2.1.1, and so on). This numbering aids in document navigation and cross-referencing.

Tables and Lists: Information presented in tables or lists is intended to enhance readability and comprehension.

Hyperlinks: Hyperlinks, when used, are in blue and underlined to distinguish them from regular text.

Acronyms and Abbreviations: Acronyms and abbreviations are defined upon first use, and the full term is provided in parentheses. Thereafter, the acronym or abbreviation may be used in the document.

1.3 Intended Audience and Reading Suggestions

This SRS document is intended to be a comprehensive reference for a diverse range of stakeholders involved in the Student Assistant project. The document caters to the following types of readers:

- **Developers:** Developers (Members of the FYP Project Team) will find detailed information about the project's functional requirements, system architecture, and design constraints in this document.

- **Supervisor:** Supervisor will benefit from understanding the project's scope, constraints, and dependencies as well as the schedule outlined in the Gantt chart in referenced proposal document.

- **Testers:** Testers will use this document to ascertain the functional requirements and performance criteria necessary for testing the Student Assistant.

- **Documentation Writer:** Writer will find information about user documentation requirements in this document, helping them create user guides and manuals.

- **Marketing Staff:** Marketing staff of Sukkur IBA can gain insights into the product's features and user classes, which can aid in crafting marketing materials.

- **Users:** While not typically direct readers of the SRS, users may refer to this document to understand the system's capabilities and constraints.

- **Reviewing Committee:** Committee members shall review the document to check for any inconsistencies as per the standards set by the University.

Document Organization:

The SRS is organized into sections, with each section addressing a specific aspect of the project.

It begins with an Introduction, providing an overview of the document's purpose and the product it covers.

The subsequent sections follow a logical sequence, covering aspects such as the project's scope, functional and nonfunctional requirements, external interfaces, and other essential details.

Readers are encouraged to start with the overview sections, which provide a high-level understanding of the project, and then delve into sections most pertinent to their roles and interests.

The sequence for reading the document, starting with the overview sections and proceeding to sections most pertinent to each reader type, is as follows:

- **Introduction:** For a broad understanding of the project's purpose and scope.
- **Overall Description:** Useful for project managers, developers, and testers to grasp the project's context and constraints.
- **External Interface Requirements:** Relevant to developers and testers dealing with user interfaces, hardware, and software interactions.
- **System Features:** Valuable for developers and testers to understand functional requirements.
- **Use Cases:** Valuable for any reader to understand the functions of the system.
- **Other Nonfunctional Requirements:** Pertinent to developers, testers, and project managers, providing insights into performance, security, and quality attributes.

- **Other Requirements:** For additional details that may not fit into the previous sections.

- **Appendices:** Resources such as a glossary.

1.4 Product Scope

The Student Assistant is a comprehensive software solution designed to cater to the diverse needs of university students and faculty members. Its purpose is to provide an integrated platform that combines academic and financial management with the assistance of a virtual AI-driven companion. The key objectives and goals of the Student Assistant project are as follows:

- **Academic Management:** The Student Assistant aims to simplify academic life by offering features such as timetable extraction from university emails, centralized access to course information, assignment deadlines, and exam schedules. It intends to streamline academic planning and organization.

- **Financial Management:** This software aspires to help students effectively manage their finances. It will provide tools for budget tracking, expense management, and financial goal setting. The objective is to promote financial responsibility among the student community.

- **AI-Powered Assistance:** The virtual AI assistant embedded in the Student Assistant is designed to enhance the user experience. It will offer personalized academic guidance, financial insights, reminders, and general assistance. The goal is to create an intelligent companion that provides valuable support.

- **Efficiency and Convenience:** The Student Assistant project seeks to offer convenience and efficiency in academic and financial management, reducing the administrative burden on students and faculty. It is aligned with the goal of simplifying complex processes and providing users with more time for their academic pursuits.

Benefits:

The implementation of the Student Assistant is expected to bring forth several benefits:

- ✓ Enhanced User Experience: Students and faculty will have a user-friendly, consolidated platform to manage their academic and financial affairs.
- ✓ Improved Academic Performance: Access to timely academic information and the support of an AI assistant can enhance academic performance and reduce stress.
- ✓ Financial Responsibility: Students will develop financial planning skills and manage their budgets more effectively, preparing them for future financial challenges.
- ✓ Operational Efficiency: The university can benefit from streamlined administrative processes and enhanced communication with students.

While this project is within the academic context of a university, it can be linked to broader educational institution goals. These might include improving the overall student experience, promoting academic success, and preparing students for their future careers. The Student Assistant aligns with these goals by providing a solution that enhances both the academic and financial aspects of students' lives, ultimately contributing to their success and well-being.

For a more detailed description of the project's scope and objectives, please refer to the Student Assistant Project Proposal document.

1.5 References

Student Assistant Project Proposal Document

Author: FYP 20F-33 Team
Version: 1.0
Date: 20 September, 2023
Source: https://github.com/20F-33-FYP-Team-Sukkur-IBA-University/Proposal
Description: The Project Proposal Document provides a detailed overview of the project's vision, background, objectives, and scope. It offers a more comprehensive perspective on the Student Assistant project.

2. Overall Description

2.1 Product Perspective

The Student Assistant stands as a self-contained and innovative software solution developed as part of our final year project. It is not a member of a product family, nor is it intended as a direct replacement for any existing systems. Instead, it serves as a unique and standalone product designed to address specific needs within the university environment.

Product Context:

The Student Assistant exists within the context of university life, targeting both students and faculty. It aims to enhance the academic and financial management experiences of users by providing a consolidated platform for these purposes.

External Interfaces:

The Student Assistant interfaces with external systems in the following ways:

- **User Interfaces:** The software provides user interfaces for students and faculty to interact with the system. These interfaces enable users to access academic schedules, track budgets, and interact with the virtual AI assistant.

- **Hardware Interfaces:** While the Student Assistant primarily operates as a software solution, it may interact with standard hardware components such as personal computers, smartphones, and tablets.

- **Software Interfaces:** The system may have software dependencies, particularly when it comes to AI components or external data sources, which it communicates with as needed.

Subsystem Interconnections:

The Student Assistant comprises several interrelated subsystems that work cohesively to deliver its functionalities:

- **Academic Management Subsystem:** This subsystem includes features like timetable extraction, academic information management, and email extraction using AI.

- **Financial Management Subsystem:** This component allows for budget tracking and financial management.

- **AI Assistant Subsystem:** The virtual AI assistant is a central part of the Student Assistant, offering personalized assistance to users.

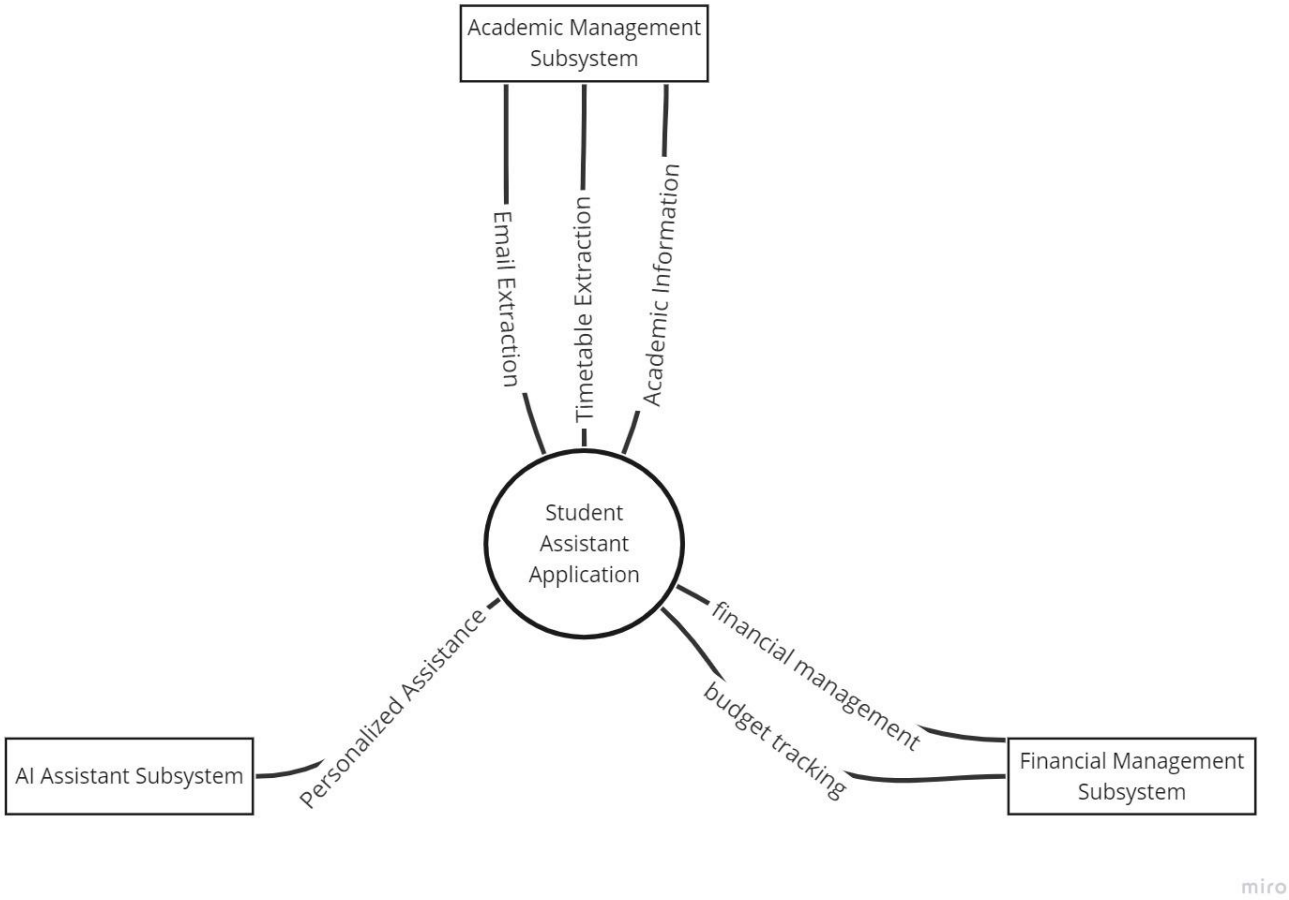

Figure 1 Context Diagram

2.2 Product Functions

❖ Academic Management Functions:

- Timetable extraction from university emails.
- Email Summaries
- Centralized access to course information.
- Deadlines and exam schedules management.

❖ Financial Management Functions:

- Budget tracking and expense management.
- Financial goal setting and tracking.
- Expense categorization and analysis.
- Generate financial reports and summaries.
- Data visualization for easy comprehension.

❖ AI Assistant Functions:

- General user assistance.
- Personalized academic and financial insights.
- Task reminders and notifications.
- Integration with academic and financial features.

❖ User Profile Management:

- User registration and login.
- Profile customization and personalization.
- Security and privacy settings.

❖ Communication and Notifications:

- Notification system for important academic dates and financial milestones.
- Integration with university announcements and communication channels.

❖ Search and Information Retrieval:

- Search functionality for academic resources, courses, and financial records.
- Quick access to academic and financial information.

❖ **User Support and Help:**

- Provide user support through the AI assistant.
- Access to a knowledge base for common queries and issues.
- Report and resolve technical problems.

❖ **Backup and Data Recovery:**

- Automatic backup of academic and financial data.
- Data recovery in case of system failures or data loss.

2.3 User Classes and Characteristics

Students:

✧ Frequency of Use: Daily or weekly.
✧ Product Functions: Use academic management and financial management features extensively.
✧ Technical Expertise: Varies from basic to advanced.
✧ Educational Level: Typically undergraduates (Semester 1 to 8) or postgraduates.
✧ Experience: May be experienced with academic tools and personal finance management.

Administrators:

✧ Frequency of Use: Periodic for system maintenance and support.
✧ Product Functions: Ensure system integrity, troubleshoot issues.
✧ Technical Expertise: Advanced, with knowledge of system administration.
✧ Educational Level: Varied but often with technical backgrounds.
✧ Experience: Experienced in system administration.

Support and Help desk:

- ✧ Frequency of Use: As needed for user support.
- ✧ Product Functions: Assist users in troubleshooting and resolving issues.
- ✧ Technical Expertise: High, with proficiency in the Student Assistant.
- ✧ Educational Level: Typically technically educated.
- ✧ Experience: Experienced in user support.

Developers and Technical Team:

- ✧ Frequency of Use: During development and maintenance.
- ✧ Product Functions: Work on system maintenance and updates.
- ✧ Technical Expertise: Advanced, with knowledge of software development.
- ✧ Educational Level: Technical backgrounds in computer science or related fields.
- ✧ Experience: Experienced in software development.

While all user classes are important, student class is the primary user class for this product. They are the core users who will rely on the academic and financial management features. Other user classes, like administrators, support, and developers, play vital roles in system maintenance and user support.

2.4 Operating Environment

- ❖ **Hardware Platform:**
  - ■ Android-based smartphones and tablets with a minimum of 2GB of RAM and dual-core processors with optimal speed.
- ❖ **Operating System and Versions:**
  - ■ Android 8.0 (Oreo) or later.
- ❖ **Development Tools:**

- Android Studio for Android app development using Kotlin.
- Python programming language for data analysis and AI components.
- Java Runtime Environment (JRE) for the AI component.

❖ **Data Analysis and AI Integration:**

- Python libraries for data analysis and machine learning, such as NumPy, Pandas, TensorFlow, and scikit-learn.
- Integration with Python-based AI models and components.

❖ **Email Extraction:**

- Integration with the Gmail API for email extraction from Gmail accounts, requiring OAuth2 authentication.
- Use of ChatGPT AI for summarisation of emails.

❖ **Internet Connectivity:**

- Reliable and secure internet connectivity is essential for real-time features and updates, as well as for interfacing with external services.

❖ **Screen Resolutions:**

- The application is designed to support a range of screen resolutions to accommodate various Android devices.

❖ **Permissions:**

- The application will request necessary permissions, such as access to email for the Gmail API, device storage for data management, and network access for communication.

The Student Assistant is built specifically for the Android operating system, intended for use on Android-based smartphones and tablets. However, if the project is completed on optimistic estimation, the product shall also be developed for IOS based Operating Systems using XCode development Environment.

It makes extensive use of Python for data analysis and AI components, enabling advanced analysis and AI-driven features within the application. The integration with the Gmail API ensures efficient email extraction from Gmail accounts. The reliable availability of internet connectivity is essential to maintain real-time features and keep the application up to date.

2.5 Design and Implementation Constraints

❖ Corporate and Regulatory Policies:

The project must adhere to the corporate policies of the university, including data security and privacy regulations. Compliance with relevant data protection and privacy laws is mandatory.

❖ Hardware Limitations:

The application's performance will be influenced by the hardware capabilities of the user's device. To ensure usability on a wide range of Android devices, the application must be optimized for devices with varying memory and processing capabilities.

❖ Interfaces to Other Applications:

The integration with the Gmail API for email extraction is subject to the availability and functionality of this external service. Any changes or limitations in the Gmail API could affect the application's email extraction capabilities.
Specific Technologies and Tools:

The use of Kotlin for Android development, Python for data analysis and AI components, and Java Runtime Environment (JRE) for the AI component are predetermined technologies.

❖ Parallel Operations:

The application should handle parallel operations efficiently, ensuring responsiveness and smooth user experience, especially when multiple features are in use simultaneously.

❖ **Security Considerations:**

Robust security measures must be implemented to protect user data and ensure secure communication. Encryption and secure data storage practices are mandatory.

❖ **Design Conventions and Programming Standards:**

The project will adhere to established design conventions and programming standards to maintain code consistency and readability. The development team will follow best practices for coding and documentation.

❖ **Maintenance Responsibility:**

While the development team will build the application, the customer's organization (university) will be responsible for maintaining the delivered software after deployment. This requires the development team to provide comprehensive documentation and support during the handover.

❖ **Compliance with Google Play Store Policies:**

If the application is intended for distribution through the Google Play Store, it must adhere to Google's policies and guidelines for app submission and updates.
These design and implementation constraints are crucial in guiding the development process and ensuring that the Student Assistant meets regulatory and organizational requirements. The development team will work within these constraints to create a secure, reliable, and efficient software solution.

2.6 User Documentation

❖ **User Manuals:**

The Application is generally easy for user of any expertise to use the application and considering the user would be an undergraduate or graduate student, the user manual is **not needed**.

❖ Tutorials and Help Guides:

The tutorial would be given at the end of the project on the use of application. The AI Assistant will be given general information about the application for help guide.

❖ Release Notes:

Release notes will accompany each version of the application, documenting updates, new features, and changes. Users will be informed about improvements and enhancements.

❖ Feedback and Contact Information:

Information on how users can provide feedback, report issues, and suggest improvements will be included. Contact details for customer support and feedback channels will also be readily available.

❖ Delivery Format:

The user documentation will primarily be delivered in digital format within the application. This approach ensures easy access for users directly from their mobile devices.

❖ Standards:

The user documentation will follow standard practices for clarity, conciseness, and user-friendliness. It will be designed to provide an intuitive and user-friendly experience.

The user documentation components are tailored to help users maximize their experience with the Student Assistant. They will be easily accessible within the application, offering clear and concise guidance to enhance user proficiency.

2.7 Assumptions and Dependencies

❖ Assumptions:

- **Third-Party Components:** It is assumed that third-party components and libraries used in the development process will be compatible with the application. Any unforeseen incompatibilities or changes in third-party components could affect the project.

- **Stable Gmail API:** We assume that the Gmail API will remain accessible and stable throughout the development and operational phases of the application. Any changes or unavailability of the Gmail API may impact the email extraction feature.

- **Consistency in Operating Environment:** We assume that the Android operating environment, including hardware and software, will remain consistent and continue to support the application. Changes in the Android ecosystem that affect compatibility may require adjustments.

- **User Data Privacy Compliance**: It is assumed that the university's data protection and privacy policies, as well as relevant laws and regulations, will not undergo significant changes that would necessitate a reevaluation of the application's data handling and security measures.

- **University's Email Format:** It is assumed that the email formats for the courses and timetable will not undergo significant changes that destruct the assumed development.

❖ **Dependencies:**

- **Gmail API Integration:** The project is dependent on the successful integration of the Gmail API for email extraction. Any disruptions or changes to the Gmail API may impact the email extraction functionality.

- **Third-Party Libraries:** The use of third-party libraries, particularly for Python-based data analysis and AI components, creates a dependency on the availability and compatibility of these libraries.

- **Data Connectivity:** The application depends on a reliable internet connection to provide real-time features and updates. Any disruptions in internet connectivity could affect the user experience.

- **Android Operating System:** The project is dependent on the Android operating system and its versions remaining compatible with the application. Changes to Android that affect compatibility must be addressed.

- **University Policies and Regulations:** The project's compliance with university data protection, privacy policies, and relevant laws is a critical dependency. Any changes in these policies may necessitate adjustments to the application.

Note: Some dependencies, such as Gmail API integration and third-party libraries, are also assumptions to a certain extent, as they depend on external factors that the project team cannot control.

3. External Interface Requirements

3.1 User Interfaces

The Student Assistant will feature a user-friendly interface that prioritizes ease of navigation and intuitive interaction for users. The following user requirements have been identified for the application's user interfaces:

- **Login/Signup Screens:**

  - A simple and secure login and sign up screens allowing users to access their personalized accounts or make new accounts respectively.

Figure 3 Proposed Login

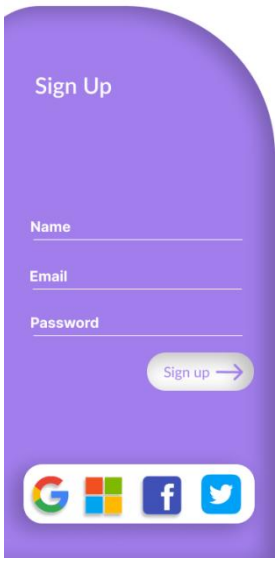

Figure 2 Proposed Signup

❖ Dashboard/Landing Page:

- A user-friendly dashboard displaying key information, and quick access to essential features.

Landing Page Prototype:

Figure 4 Proposed Landing Page

❖ Timetable Extractor:

- A clear and organized display of the extracted timetable from the university mail.
- Intuitive filtering and search options for easy access to specific classes and schedules.

Timetable Prototype:

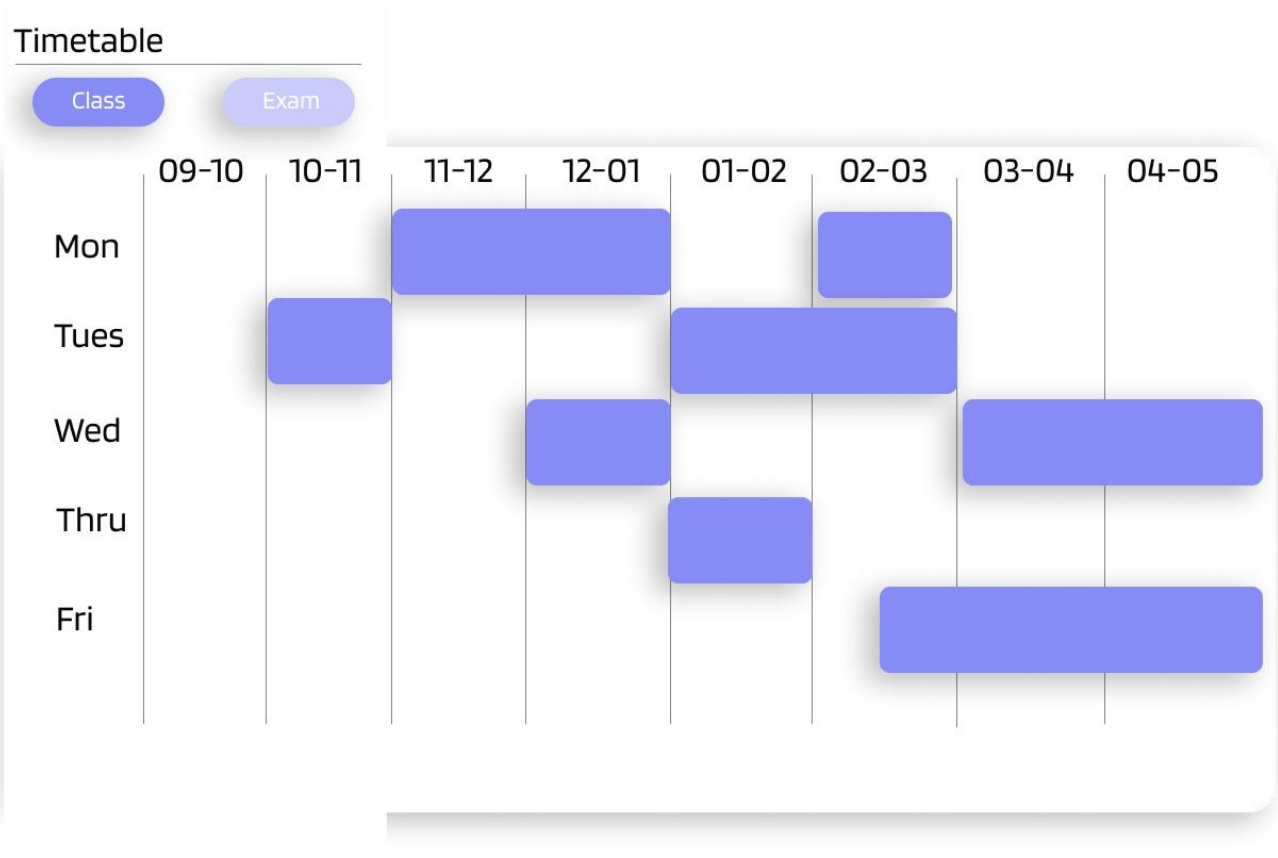

Figure 5 Proposed Timetable

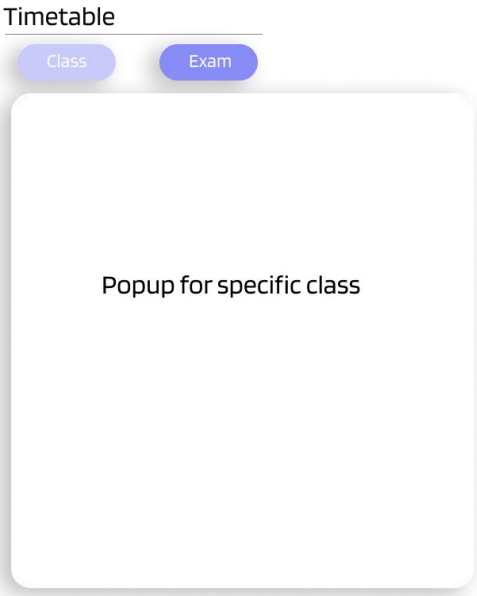

Figure 6 Proposed Timetable Popup

❖ Financial Tracker:

- An interactive financial tracker displaying expenses, budgets, and financial goals in a visually informative manner.
- Visualization of data and prediction of expenses.

Financial Tracker Prototype:

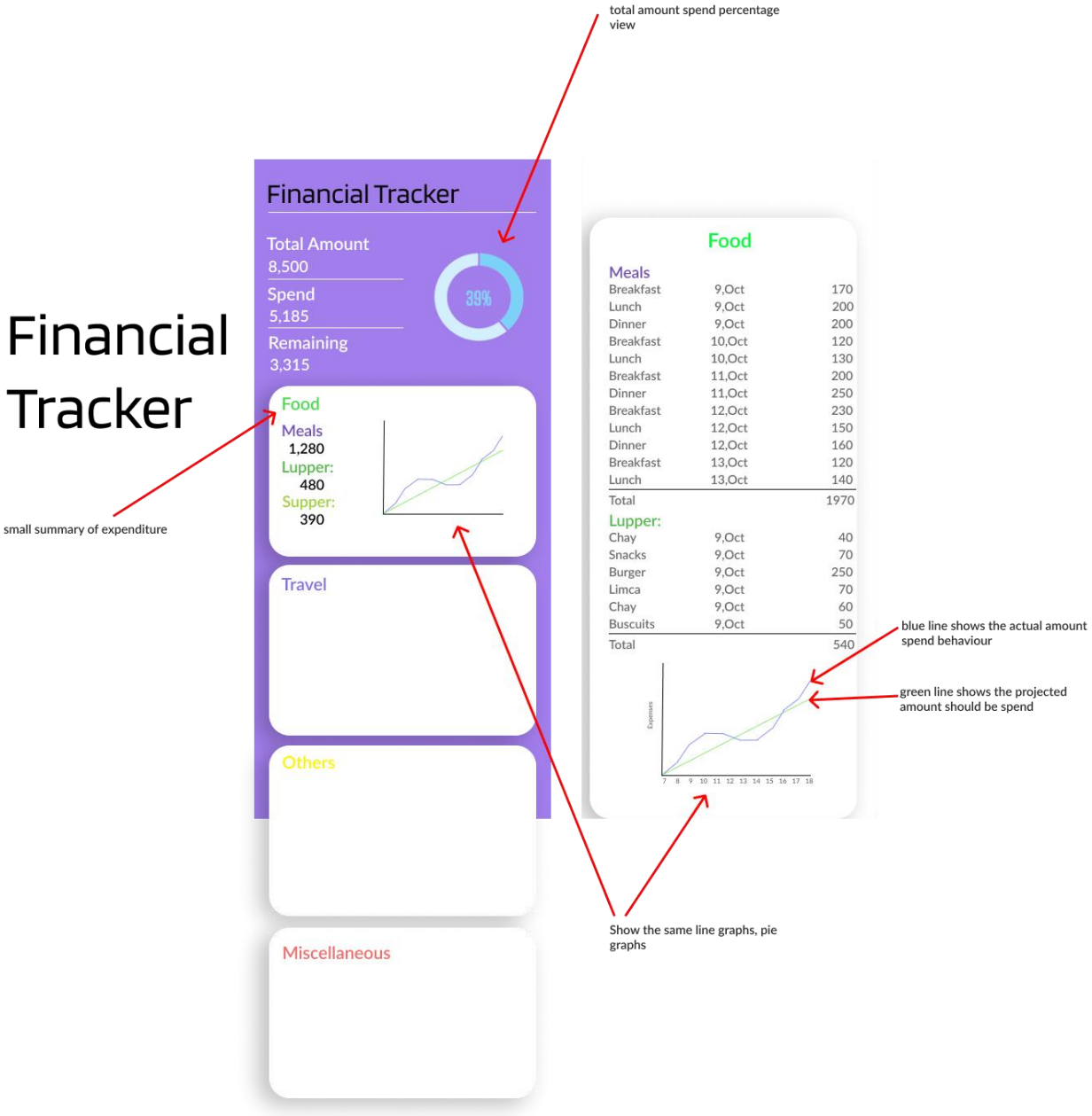

Figure 7 Proposed Financial Tracker

❖ Virtual AI Assistant:

- A conversational and responsive virtual AI assistant for intuitive user interactions.
- Natural language processing for seamless communication and task execution.

Virtual AI Assistant Prototype:

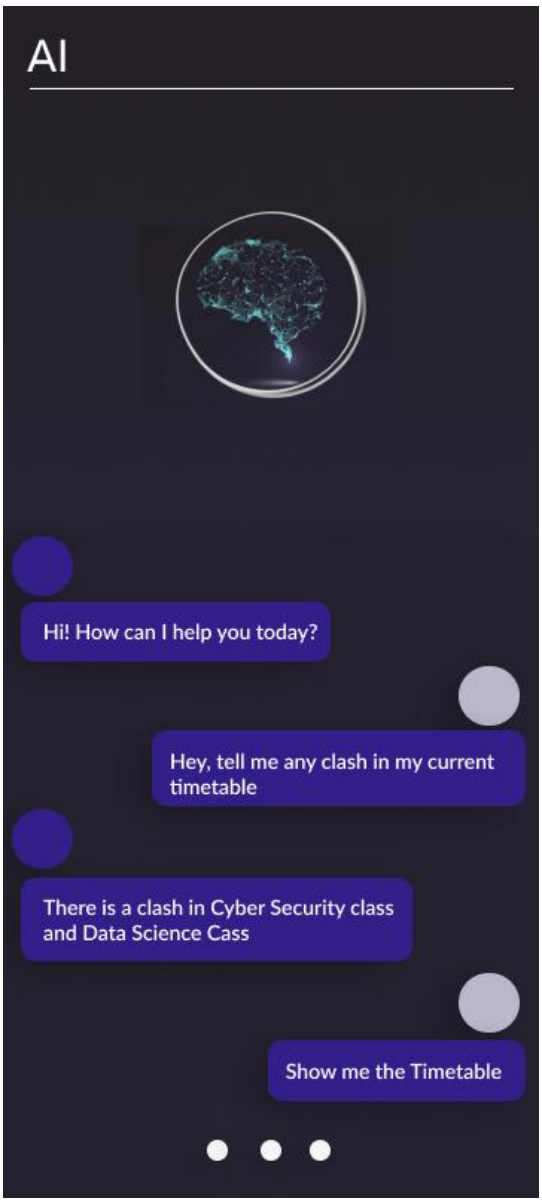

Figure 8 Proposed Virtual AI Assistant Interface

❖ Email Summarizer:

- An interface that presents summaries of emails, providing key details without requiring users to open the full emails.
- Options to mark emails for further review or action.
- Options to manually update the information.

Email Summarizer Prototype:

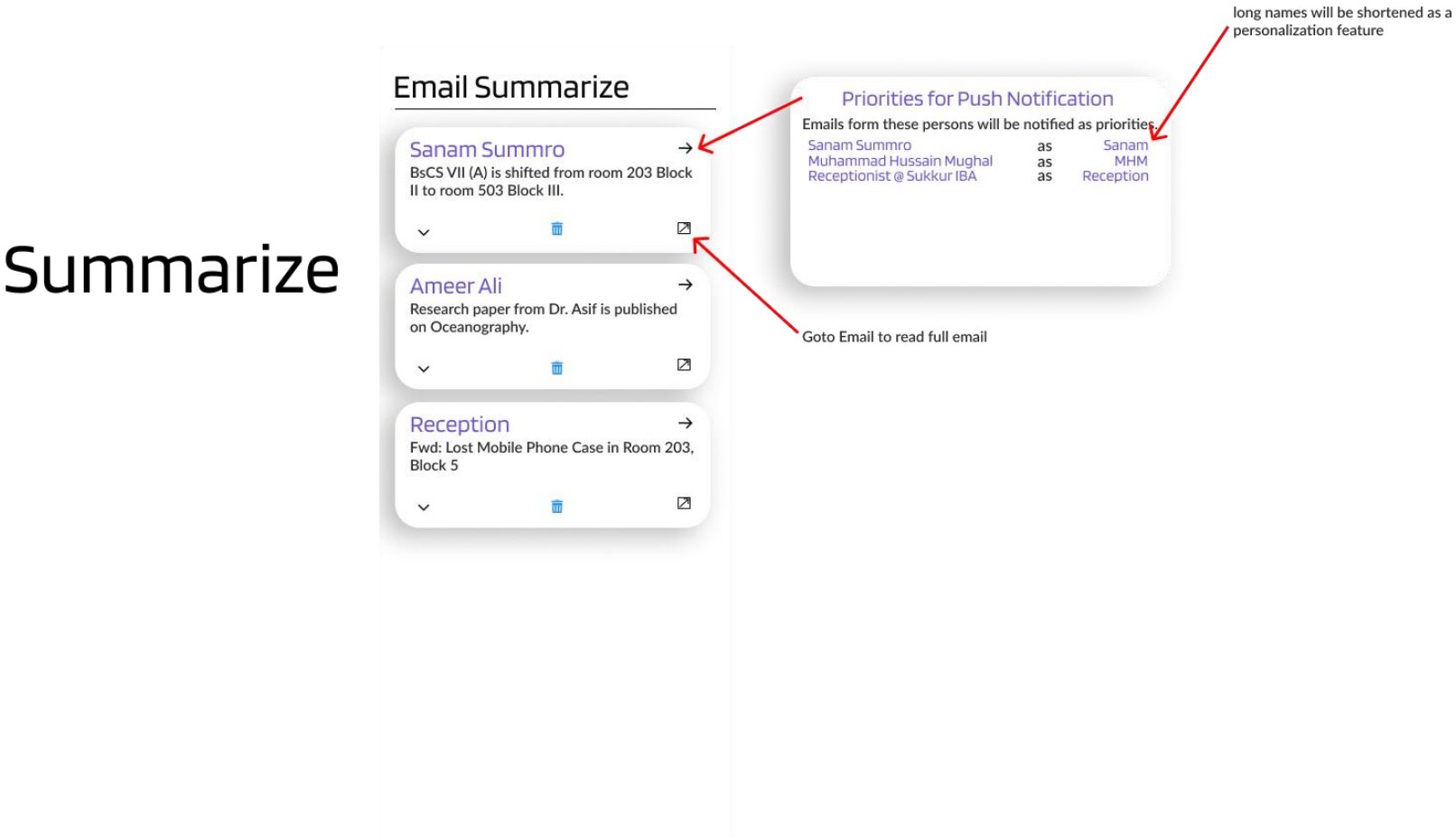

Figure 9 Proposed Email Summarizer

❖ **Settings and Preferences:**

- An easily accessible settings menu allowing users to customize application preferences and security settings.
- Options for data backup, and notification preferences.

❖ **Error Handling and Messaging:**

- Clear and concise error messages displayed in a consistent format to guide users in resolving issues.

❖ **Responsive Design:**

- The user interface should be responsive, adapting to different screen sizes and orientations, to ensure usability on a variety of devices.

❖ **Navigation:**

- Navigation bar shall be at the bottom of the screen to switch among the modules.

Informative prompts and notifications for updates. The user interface design will adhere to modern design principles, ensuring a cohesive and visually appealing experience for users. It will be intuitive, responsive, and consistent across different sections of the application, bringing up a seamless user journey.

3.2 Hardware Interfaces

The Student Assistant is designed to interact with the hardware components of the system in a manner that ensures optimal performance and functionality. The hardware interfaces include:

- **Device Compatibility:**

The Student Assistant is compatible with a wide range of Android-based hardware devices, including smartphones and tablets. The application will adapt to different screen sizes and resolutions.

- **Biometric Authentication (Optional):**

Devices equipped with biometric authentication capabilities, such as fingerprint scanners or facial recognition, can leverage these features for secure login and access.

- **Data Storage:**

The application interacts with the device's internal storage and external storage options to manage data, such as financial records.

- **Internet Connectivity:**

The application requires access to the device's network interfaces to ensure internet connectivity. This enables real-time features and communication with external services.

- **Camera and Microphone (Optional):**

In cases where the application uses features like image scanning or voice commands, access to the device's camera and microphone is necessary. Users will be prompted to grant permission for these hardware components.

- **Sensors (Optional):**

If the application integrates sensor data, such as location information from GPS, it will require access to the device's sensors.

- **Bi-Directional Communication:**

The application communicates with the hardware components for both data input and output. For example, it reads user input through touchscreen interaction and provides visual and auditory output through the device's screen and speakers.

❖ **Communication Protocols:**

The application utilizes standard communication protocols, such as HTTP/HTTPS for web-based services, SMTP for email, and OAuth2 for Gmail API integration, to interact with external servers and services.

The Student Assistant is designed to work seamlessly with various hardware components while ensuring a consistent and user-friendly experience across different devices. It adapts to the hardware characteristics to provide users with a reliable and efficient utility.

3.3 Software Interfaces

The Student Assistant interacts with several software components and interfaces to ensure seamless functionality and data exchange. These software interfaces include:

❖ **Operating System:**

The application is designed to operate on Android operating systems (OS) version 8.0 (Oreo) and later. It utilizes OS-specific services and APIs for various functions.

❖ **Development Tools:**

Android Studio is used for Android app development using Kotlin. Python programming language, along with libraries for data analysis and AI components, is utilized for specific functionalities.

❖ **Java Runtime Environment (JRE):**

The AI component of the application requires Java Runtime Environment for execution. This component interacts with the application for AI-driven features.

❖ Gmail API:

The application interfaces with the Gmail API to extract and summarize emails. It uses OAuth2 for user authentication and data access.

❖ Internet Connectivity:

The application relies on internet connectivity for real-time features, data synchronization, and communication with external services. It uses standard communication protocols, such as HTTP/HTTPS, SMTP, and OAuth2, for these interactions.

❖ Data Storage:

The application may utilize Amazon Web Services (AWS) for data storage and processing, depending on configuration. Data storage and retrieval involve interactions with cloud-based databases and services.

❖ Libraries and Components:

The application may use third-party libraries and components for specific functionalities. These components include but are not limited to machine learning libraries, email parsing libraries, and data visualization tools.

❖ External APIs (Optional):

If the application integrates with external services, such as university information systems or financial institutions, it will interact with their APIs, complying with their protocols and security measures.

❖ User Interfaces:

The application provides user interfaces for user interaction and input. It communicates with the user through the user interfaces, including login screens, dashboards, email summaries, financial trackers, and the virtual AI assistant.
The Student Assistant seamlessly interfaces with various software components and APIs to deliver its comprehensive features. Data sharing occurs within the application and with external services, maintaining data integrity and security throughout the process.

3.4 Communications Interfaces

❖ **Internet Communication:**

The application relies on internet connectivity for several functions, including email extraction, financial data synchronization, and interaction with external services. It uses standard communication protocols, such as HTTP/HTTPS, for data transfer.

❖ **Email Communication:**

The application communicates with the Gmail API for email extraction and summarization. It follows the OAuth2 protocol for secure user authentication and access to Gmail data.

❖ **Data Synchronization:**

Data synchronization between the application and external cloud-based storage or processing services (e.g., Amazon Web Services) is achieved through secure, encrypted channels.

❖ **Security and Encryption:**

All data transmissions are secured through encryption protocols to safeguard user data and privacy. HTTPS is used for secure web-based communications, and OAuth2 ensures secure email API access.

❖ **Data Transfer Rates:**

The application aims to provide efficient data transfer rates, particularly when syncing financial data or accessing email summaries. Performance considerations are taken into account to optimize data transfer rates.

❖ **Synchronization Mechanisms:**

Synchronization with external services, databases, or cloud storage is implemented through robust synchronization mechanisms to ensure data consistency across devices and platforms.

❖ Message Formatting:

Messages and data exchanged conform to standard data formats and structures to enable seamless integration with external systems and services.

❖ Communication Standards:

The application adheres to widely accepted communication standards, such as HTTP, HTTPS, and OAuth2, ensuring compatibility and interoperability with external systems and services.

❖ Real-time Communication:

Real-time communication features, such as chat interactions with the virtual AI assistant, involve instantaneous data exchange with minimal latency.
The Student Assistant places a strong emphasis on secure and efficient communication, employing encryption, standardized protocols, and synchronization mechanisms to facilitate reliable and responsive interactions with external systems and services.

4. System Features

Functional Hierarchy of Student Assistant:

Student Assistant

- Main Module
  - User Authentication
  - User Profile Management
  - Dashboard/Landing Page

- Timetable Management

  - View Timetable
  - Add Class
  - Edit Class
  - Delete Class

- Financial Tracker
  - Expense Tracking
  - Budget Management
  -Expense Suggestion

- Email Management
  - Email Summarization
  - Email Filtering
  - Email-Based Updates

- Virtual AI Assistant
  - Chat Interaction
  - Information Retrieval

4.1 User Authentication

4.1.1 Description and Priority

Description: The User Authentication feature is responsible for ensuring secure access to the application. It allows users to log in to their accounts, protecting their data and maintaining privacy. This feature is critical to the overall system's security and user identity management.

Priority: High

4.1.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Successful Login

Stimulus: The user enters a valid username and password and clicks the "Log In" button.

Response: The system validates the credentials, and if they are correct, the user is granted access to the application's main dashboard.

Stimulus/Response Sequence 2: Invalid Credentials

Stimulus: The user enters an incorrect username or password and attempts to log in.
Response: The system detects the invalid credentials and displays an error message. The user is prompted to re-enter the correct login information.

Stimulus/Response Sequence 3: Forgotten Password

Stimulus: The user clicks the "Forgot Password" link.
Response: The system guides the user through the password recovery process, which typically involves sending a password reset link to the user's registered email address.

4.1.3 Functional Requirements

UA-01: User Authentication

The system shall provide a user authentication mechanism that requires a valid username and password.
The system shall hash and securely store user passwords in the database.
The system shall enforce password complexity rules, including a minimum length and the use of alphanumeric characters and symbols.

UA-02: Account Lockout

The system shall implement an account lockout policy to prevent brute force attacks. After a specified number of failed login attempts, the account shall be temporarily locked, and the user shall be notified.

UA-03: Password Recovery

The system shall allow users to recover their passwords through a secure email-based process.
When a user requests a password reset, the system shall generate a time-limited reset link and send it to the user's registered email address.

UA-04: Secure Session Management

The system shall manage user sessions securely, using best practices such as session tokens and timeouts to protect against session hijacking.

4.2 User Profile Management

4.2.1 Description and Priority

Description: The "User Profile Management" feature allows users to update and manage their personal information, such as name, contact details, and profile picture. It empowers users to maintain accurate and current profiles within the system.

Priority: Medium

4.2.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Profile Update

Stimulus: The user selects the "Edit Profile" option and makes changes to their profile information.
Response: The system accepts the updated information, validates it, and reflects the changes in the user's profile.

Stimulus/Response Sequence 2: Profile Picture Change

Stimulus: The user uploads a new profile picture.
Response: The system uploads and displays the new profile picture, ensuring it meets specified size and format requirements.

Stimulus/Response Sequence 3: Invalid Input

Stimulus: The user attempts to update their profile with missing or invalid information.
Response: The system identifies the invalid input and prompts the user to provide complete and valid profile details.

4.2.3 Functional Requirements

UP-1: Profile Editing

The system shall provide an interface for users to edit their profile information, including their name, contact details, and other relevant data.

UP-2: Input Validation

The system shall validate user input when updating their profile to ensure completeness and adherence to specified data formats.
In the case of missing or invalid information, the system shall display appropriate error messages.

UP-3: Profile Picture Management

The system shall allow users to upload and change their profile pictures.
Uploaded profile pictures shall adhere to specified size and format requirements.

UP-4: Profile Display

The system shall accurately display the user's profile information, including the updated details and profile picture.

4.3 Landing Page (Dashboard)

4.3.1 Description and Priority

Description: The "Landing Page" serves as the user's initial point of interaction with the application. It provides an overview of key information, such as upcoming classes, financial summaries, and notifications. This feature offers a quick and informative start to the user's experience.

Priority: High

4.3.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: User Login

Stimulus: The user logs in to the application.
Response: The system directs the user to the Landing Page, where relevant information is displayed.

Stimulus/Response Sequence 2: Upcoming Classes Display

Stimulus: The user views the Landing Page.
Response: The system fetches and displays a list of the user's upcoming classes or events.

Stimulus/Response Sequence 3: Financial Summary Display

Stimulus: The user accesses the Landing Page.
Response: The system retrieves and shows a summary of the user's financial data, including expenses, budgets, and account balances.

4.3.3 Functional Requirements

LP-1: User Login Redirect

After a successful login, the system shall direct the user to the Landing Page.

LP-2: Upcoming Classes Display

The Landing Page shall show a list of the user's upcoming classes or events, including details such as class name, date, time, and location.

LP-3 Financial Summary Display

The Landing Page shall provide a summary of the user's financial information, including expenses, budgets, and account balances.

LP-4 Notification Display

The Landing Page shall display relevant notifications, such as system updates or event reminders.

4.4 View Time Table

4.4.1 Description and Priority

Description: The "View Time Table" feature allows users to access and display their class schedules and events in a user-friendly format. Users can view their timetable for the day, week, or month, making it convenient to plan their activities.

Priority: Medium

4.4.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Daily Timetable

Stimulus: The user selects the "View Daily Timetable" option.
Response: The system displays the user's class schedule for the current day, including class names, times, and locations.

Stimulus/Response Sequence 2: Weekly Timetable

Stimulus: The user chooses the "View Weekly Timetable" option.
Response: The system presents a weekly overview of the user's classes, allowing them to plan their activities for the upcoming week.

4.4.3 Functional Requirements

VT-1: Timetable Display

The system shall provide options for users to view their timetable for the day, week, and month.
Users shall be able to select specific date ranges for viewing their timetable.

VT-2: Class Information

The system shall display class details, including class names, times, locations, and instructors.
Users shall be able to click on a class to view additional details or make updates.

VT-3: Navigation

Users shall have navigation controls for moving between days and weeks.
The system shall offer a user-friendly interface for easy navigation within the timetable.

4.5 Add Class

4.5.1 Description and Priority

Description: The "Add Class" feature enables users to include new classes or academic activities in their timetable. Users can input essential class details such as the course name, instructor, time, and location, facilitating effective schedule management.

Priority: High

4.5.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Adding a Class

Stimulus: The user selects the "Add Class" option and inputs class details, including the course name, instructor, time, and location.
Response: The system stores the new class information and updates the user's timetable accordingly.

Stimulus/Response Sequence 2: Conflict Detection

Stimulus: The user attempts to add a class that conflicts with an existing schedule.

Response: The system identifies the conflict and notifies the user, suggesting alternative time slots or adjustments.

4.5.3 Functional Requirements

AD-1: Class Information Input

The system shall provide a user interface for users to input class details, including the course name, instructor, time, and location.
Users shall be able to add new classes with complete and accurate information.

AD-2: Conflict Resolution

The system shall check for scheduling conflicts when users attempt to add new classes.
In the case of conflicts, the system shall offer solutions or alternative options to resolve the issue.

AD-3: Timetable Update

After successful addition, the system shall update the user's timetable to reflect the newly added class.
The timetable shall accurately display the new class details and schedule changes.

4.6 Edit Class/Exam

4.6.1 Description and Priority

Description: The "Edit Class/Exam" feature allows users to make modifications to existing class or exam details within their timetable. Users can update information such as the course name, instructor, time, or location, exam ensuring that their schedule remains accurate and up-to-date.

Priority: High

4.6.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Editing a Class/Exam

Stimulus: The user selects the "Class or Exam" option for a specific class/exam within their timetable and makes changes to class/Exam details.
Response: The system updates the class/exam information, and the user's timetable reflects the modifications.

Stimulus/Response Sequence 2: Conflict Detection

Stimulus: The user attempts to edit a class/exam in a way that creates a scheduling conflict.
Response: The system identifies the conflict and notifies the user, offering suggestions to resolve the issue.

4.6.3 Functional Requirements

EC-1: Class Modification

The system shall provide a user-friendly interface for users to edit class details, including the course name, instructor, time, and location.
Users shall be able to update class information accurately.

EC-2: Conflict Resolution

The system shall check for scheduling conflicts when users attempt to edit class details or exam details.
In the case of conflicts, the system shall offer solutions or alternative options to resolve the issue.

EC-3: Timetable Update

After successful editing, the system shall update the user's timetable to reflect the modified class/exam details.

The timetable shall accurately display the updated class information and any schedule changes.

4.7 Delete Class/Exam

4.7.1 Description and Priority

Description: The "Delete Class/Exam" feature allows users to remove classes or academic activities(e.g., Exams) from their timetable when they are no longer relevant or required. Users can select the class/exam they wish to delete and confirm the removal, keeping their schedule up-to-date.

Priority: Medium

4.7.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Deleting a Class/Exam

Stimulus: The user selects the "Delete Icon" option for a specific class/Exam within their timetable and confirms the deletion.
Response: The system removes the class/Exam from the user's timetable, adjusting the schedule accordingly.

Stimulus/Response Sequence 2: Confirmation Prompt

Stimulus: The user initiates the deletion process.
Response: The system displays a confirmation prompt to ensure that the user intends to delete the selected class or Exam.

4.7.3 Functional Requirements

DC-1: Class or Exam Deletion

The system shall provide a user interface for users to delete classes or Exams from their timetable.
Users shall be able to select and confirm the deletion of classes/Exams.

DC-2: Confirmation Prompt

When users choose to delete a class/exam, the system shall display a confirmation prompt to prevent accidental deletions.
Users shall confirm the deletion action before the class/exam is removed.

DC-3: Timetable Update

After successful deletion, the system shall update the user's timetable to reflect the removal of the class.
The timetable shall accurately display the adjusted schedule without the deleted class/exam.

4.8 Expense Tracking

4.8.1 Description and Priority

Description: The "Expense Tracking" feature allows users to record and manage their expenses, helping them maintain financial control. Users can log various types of expenses, categorize them, and track their spending over time.

Priority: High

4.8.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Adding an Expense

Stimulus: The user selects the "Add Expense" option and enters expense details, including the amount, category, date, and description.
Response: The system records the expense, calculates new totals, and updates the user's financial summary.

Stimulus/Response Sequence 2: Expense Categorization

Stimulus: The user categorizes an expense, such as marking it as a "food" or "transportation" expense.
Response: The system assigns the expense to the specified category, allowing users to track spending by category.

Stimulus/Response Sequence 3: Expense Summary

Stimulus: The user accesses the expense tracking module.
Response: The system displays a summary of the user's expenses, including total spending, categorized expenses, and expense trends
.

4.8.3 Functional Requirements

ET-1: Expense Recording

The system shall provide a user-friendly interface for recording expenses, including fields for amount, category, date, and description.
Users shall be able to add new expenses and specify expense details.

ET-2: Expense Categorization

The system shall allow users to categorize expenses into predefined categories or create custom categories.
Users shall be able to assign expenses to the selected categories.

ET-3: Expense Tracking

The system shall calculate and display a summary of the user's expenses, including total spending, categorized expenses, and spending trends over time.
Users shall have access to visual representations of their spending data, such as charts and graphs.

ET-4: Expense Modification

Users shall have the ability to edit or delete recorded expenses.
The system shall recalculate totals and update the financial summary accordingly.

4.9 Budget Management

4.9.1 Description and Priority

Description: The "Budget Management" feature empowers users to set financial goals, create budgets, and monitor their spending to ensure they stay within their financial limits. Users can define budget categories, allocate funds, and receive alerts when approaching budget limits.

Priority: High

4.9.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Creating a Budget

Stimulus: The user selects the "Create Budget" option and defines budget categories, allocation amounts, and spending limits.
Response: The system saves the budget settings, and the user's financial summary is updated with the budget information.

Stimulus/Response Sequence 2: Expense Tracking Against Budget

Stimulus: The user records expenses or income.
Response: The system automatically compares expenses to budget categories and provides alerts when the user is approaching or exceeding budget limits.

Stimulus/Response Sequence 3: Budget Visualization

Stimulus: The user accesses the budget management module.

Response: The system displays visual representations of budget categories, actual spending, and the progress towards financial goals.

4.9.3 Functional Requirements

BM-1: Budget Creation

The system shall provide a user-friendly interface for creating budgets.
Users shall be able to define budget categories, allocate funds, and set spending limits for each category.

BM-2: Expense Tracking Against Budget

The system shall continuously track expenses and income and compare them to budget categories.
Users shall receive real-time alerts when approaching or exceeding budget limits.

BM-3: Budget Visualization

The system shall display budget categories, actual spending, and progress towards financial goals using charts and graphs.
Users shall have access to visual representations of their budget data.

BM-4: Budget Modification

Users shall have the ability to modify existing budgets, including adding, deleting, or editing budget categories and allocation amounts.
The system shall update the financial summary and spending alerts accordingly.

4.10 Expense Guide

4.10.1 Description and Priority

Description: The "Expense Guide" feature offers intelligent and automated expense recommendations to users, making it easier for them to manage their finances. It provides suggestions on potential expenses based on historical data, current financial status, and user preferences.

Priority: Medium

4.10.2 Stimulus/Response Sequences

Stimulus: The user opens the Financial Tracker application.
Response: The system analyzes the user's financial data, such as income, previous expenses, and financial goals.

4.10.3 Functional Requirements

EG-1: The system shall collect and store the user's financial data securely.
EG-2: It should analyze the user's financial history to identify spending patterns.
EG-3: The system shall consider the user's financial goals and budget constraints.
EG-4: It will generate personalized expense recommendations, including categories and approximate amounts.
EG-5: Users should be able to view and accept or reject these suggestions.
EG-6: If accepted, the system shall provide details for the suggested expense, such as vendor, date, and purpose.
EG-7: The system will monitor and learn from user interactions to improve the accuracy of future expense recommendations.

4.11 Email Summarization

4.11.1 Description and Priority

Description: The "Email Summarization" feature employs AI algorithms to provide users with concise summaries of their email messages. Users can reduce the time spent on reading and processing emails, making it easier to manage their email communications.

Priority: Medium

4.11.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Email Detection and Summarization

Stimulus: The system automatically detects incoming email messages and assesses their relevance to the user's interests or priorities.
Response: If an email is determined to be of interest to the user, the system processes the email content using AI algorithms and provides a brief summary of the email's key points.

4.11.3 Functional Requirements

ES-1: Email Summarization

The system shall offer users the ability to summarize email messages using AI-based algorithms.
Users shall be able to generate summaries for individual email messages.

ES-2: Summarization Accuracy

The system shall prioritize accuracy in email summarization, ensuring that key information is included in the summaries.
Summarized emails shall be coherent and meaningful to users.

ES-3: Summarization History

The system shall maintain a history of summarized emails for users to reference.
Users shall have access to past summaries for quick email review.

4.12 Email Filtering

4.12.1 Description and Priority

Description: The "Email Filtering" feature enables users to apply filters and rules to their email messages, automatically organizing and prioritizing incoming emails. Users can categorize emails, mark them as important, or route them to specific folders based on predefined criteria.

Priority: High

4.12.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Creating Email Filters

Stimulus: The user accesses the email filtering settings and creates rules or criteria for filtering incoming emails.
Response: The system saves the user's filter settings, and incoming emails are filtered according to the specified criteria.

Stimulus/Response Sequence 2: Automatic Email Categorization

Stimulus: An incoming email matches the criteria of a predefined filter.
Response: The system automatically categorizes the email and applies any specified actions (e.g., given high priority).

Stimulus/Response Sequence 3: User-Defined Filters

Stimulus: The user configures custom filters, defining criteria such as sender, subject keywords, or recipient.
Response: The system processes incoming emails, applying the user's custom filters for organization and prioritization.

4.12.3 Functional Requirements

EF-1: Filter Creation

The system shall provide a user-friendly interface for creating and managing email filters.

Users shall be able to define filter criteria, actions, and folder destinations.

EF-2: Automatic Email Categorization

The system shall automatically apply predefined filters to incoming emails that match the specified criteria.
Users shall have emails categorized and organized according to their preferences.

EF-3: User-Defined Filters

The system shall allow users to create custom filters with criteria based on sender, subject, recipient, or other attributes.
Users shall have the flexibility to define custom rules for email organization.

EF-4: Filter Management

Users shall be able to edit, delete, or disable filters as needed.
The system shall adapt to changes in filter settings.

4.13 Email-Based Updates

4.13.1 Description and Priority

Description: The "Email-Based Updates" feature empowers users to automatically update their timetables in response to emails from the program officer. The system summarizes relevant emails, presents the information to the user for verification and modification, and with user permission, it automatically updates the timetable.

Priority: High

4.13.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: Timetable Update from Program Officer Email

Stimulus: An email from the program officer with timetable updates or changes is received.
Response: The system processes the email content, summarizes the timetable updates, and presents them to the user for review.

Stimulus/Response Sequence 2: User Verification and Confirmation

Stimulus: The user reviews the summarized timetable updates, makes any necessary changes or corrections, and provides confirmation to proceed with the updates.
Response: The system awaits user confirmation before implementing the timetable changes.

Stimulus/Response Sequence 3: Automatic Timetable Update

Stimulus: Upon receiving user confirmation, the system automatically updates the timetable with the approved changes.
Response: The timetable is updated, reflecting any modifications or additions specified by the user.

4.13.3 Functional Requirements

EU-1: Email-Based Timetable Summarization

The system shall process emails from the program officer containing timetable updates.
It shall summarize the timetable changes, including class rescheduling, venue changes, or additional classes.

EU-2: User Review and Verification

The system shall present the summarized timetable updates to the user for review.
Users shall be able to modify the timetable information or correct inaccuracies in the summary.

EU-3: User Confirmation for Updates

The system shall await user confirmation before implementing any timetable changes.
Users must explicitly approve the changes, ensuring their consent.

EU-4: Automatic Timetable Update

Upon receiving user confirmation, the system shall automatically update the timetable with the approved changes.
The updated timetable shall accurately reflect the user's preferences and the confirmed changes.

4.14 Chat Interaction with Virtual AI Assistant

4.14.1 Description and Priority

Description: The "Chat Interaction" feature allows users to engage in conversational interactions with the Virtual AI Assistant. Users can ask questions, request information, and receive assistance through a natural language chat interface.

Priority: High

4.14.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: User Initiated Chat

Stimulus: The user initiates a chat session with the Virtual AI Assistant by sending a message or query.
Response: The Virtual AI Assistant responds to the user's message, providing information or assistance based on the query.

Stimulus/Response Sequence 2: Contextual Understanding

Stimulus: The user engages in a conversation with the Virtual AI Assistant, asking questions or providing context.
Response: The system's AI algorithms maintain context and provide relevant responses, demonstrating an understanding of the ongoing conversation.

Stimulus/Response Sequence 3: Task Completion

Stimulus: The user requests the Virtual AI Assistant to perform specific tasks, such as setting reminders, retrieving information, or initiating actions.
Response: The Virtual AI Assistant executes the requested tasks and informs the user of the outcome.

4.14.3 Functional Requirements

CI-1: Natural Language Processing

The system shall incorporate natural language processing capabilities for understanding and generating human-like responses in chat interactions.
Users shall be able to communicate with the Virtual AI Assistant using conversational language.

CI-2: Contextual Understanding

The system shall maintain context during chat interactions, allowing the Virtual AI Assistant to comprehend and respond contextually.
Users shall have seamless, context-aware conversations with the AI.

CI-3: User Assistance

The system shall provide assistance and support to users by answering questions, offering guidance, and offering information.
Users shall benefit from reliable and informative responses from the Virtual AI Assistant.

4.15 Information Retrieval

4.15.1 Description and Priority

Description: The "Information Retrieval" feature enables users to request and receive information from the Virtual AI Assistant. Users can ask questions, seek explanations, and access data on various topics.

Priority: High

4.15.2 Stimulus/Response Sequences

Stimulus/Response Sequence 1: User Information Query

Stimulus: The user initiates an information query by sending a question or request to the Virtual AI Assistant.
Response: The AI processes the query, retrieves relevant information, and provides a response to the user.

Stimulus/Response Sequence 2: Multilingual Information Retrieval

Stimulus: The user submits a query in their preferred language.
Response: The Virtual AI Assistant processes queries in multiple languages, ensuring a diverse user base can access information.

Stimulus/Response Sequence 3: Knowledge Expansion

Stimulus: The AI Assistant updates its knowledge base with new information or corrections based on user interactions.
Response: The system continuously improves its information retrieval capabilities and knowledge accuracy.

4.15.3 Functional Requirements

IR-1: Natural Language Processing for Queries

The system shall support natural language processing to understand and interpret user queries accurately.
Users shall be able to ask questions in plain language, enhancing ease of use.

IR-2: Multilingual Support

The system shall accommodate queries in multiple languages, including but not limited to English, providing a globally accessible information retrieval experience.
Users shall have the flexibility to interact in their preferred language.

IR-3: Knowledge Base Management

The system shall maintain a knowledge base containing diverse and up-to-date information.
The Virtual AI Assistant shall continuously expand its knowledge and update existing data based on user interactions.

IR-4: User Feedback Integration

The system shall encourage user feedback on the accuracy and quality of provided information.
Users shall be able to report inaccuracies, which the system will consider for knowledge base improvements.

5. Use Cases

5.1 Manage Email Filtering

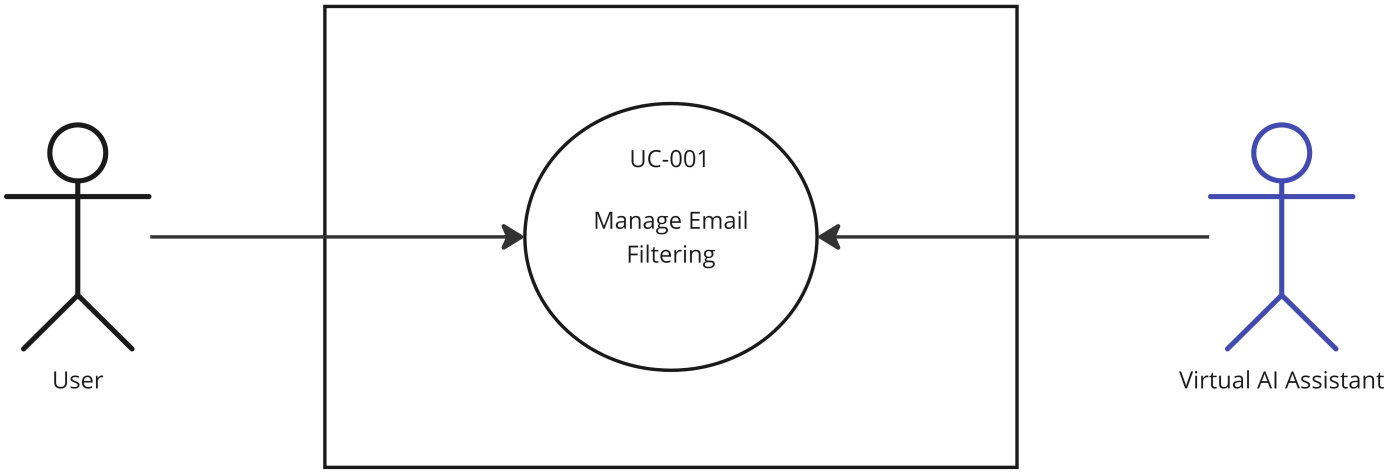

Figure 10 Use Case 1

Description

The "Manage Email Filtering" use case is about allowing users to create, modify, and manage email filtering rules within the Student Utility App. These rules enable users to automatically categorize and organize incoming emails based on specific criteria, such as sender, subject, or keywords, improving email organization and management. The use case provides control over how emails are processed, making it easier for users to focus on relevant information.

Use case Id: U C - 0 0 1		
Use case Id:	UC-001	
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Virtual AI Assistant***		
Feature:	Email Management	
Pre-condition:	- User is logged in to the application. - User has access to their email inbox. - Virtual AI Assistant is active and operational.	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	User selects the "Manage Email Filtering" option within the email management section	The system displays the current list of filtering rules, if any.
2.	-User chooses to add a new filtering	-The system updates the list of filtering rules with

	rule or edit an existing one. -User defines filtering criteria, which may include sender, subject, keywords, or other attributes. -User specifies the action to be taken when an email matches the filtering criteria (e.g., move to a specific folder, mark as important). -User saves the filtering rule.	the new rule or changes. - The system applies the filtering rules to incoming emails, automatically categorizing and organizing them according to the defined criteria.

Alternate Scenarios:

1a:
Action:
1. User cancels the creation or modification of a filtering rule.

Software Reaction:
1. The system discards any changes and returns to the list of filtering rules.

2a:
Action:
1. User selects an existing filtering rule for editing.
2. User modifies the filtering criteria or action.
3. User deletes the selected filtering rule.

Software Reaction:
1. The system updates the rule with the new criteria or action.
2. The system removes the deleted rule from the list.

Post Conditions

Step#	***Description***
1.	The user has successfully initiated the "Manage Email Filtering" use case, and the list of filtering rules is displayed.

Use Case Cross referenced	None

5.2 Email Actions and Timetable Updates

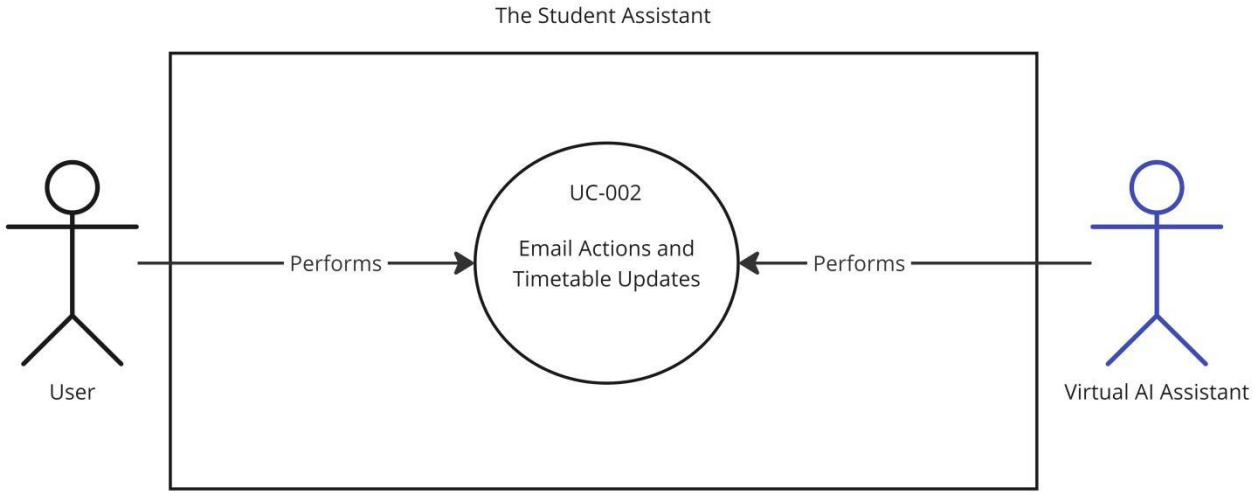

Figure 11 Use Case 2

Description:

This use case enables users to perform actions on emails and, when necessary, update the timetable in response to email contents.

Use case Id: U C - 0 0 2		
Use case Id:	UC-002	
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Virtual AI Assistant***		
Feature:	Email Management	
Pre-condition:	- User is logged in to the application. - User has access to their email inbox. - Virtual AI Assistant is active and operational.	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	User selects an email in their inbox to perform actions (e.g., open).	The system executes the selected email action as requested.
2.	User receives an email from an academic authority, such as the program officer, with timetable-related	The system processes the email to extract timetable-related information. The Virtual AI Assistant presents the extracted

	updates. The Virtual AI Assistant detects the email and offers to update the user's timetable based on its contents.	information to the user for review and confirmation. If the user approves the changes, the system automatically updates their timetable accordingly.
Alternate Scenarios:		
1a: *Action:* *1. User cancels or closes the email action before completion.* *Software Reaction:* *1. The system cancels the email action, returning to the previous state.* ***2a:*** *Action:* *1. User receives an email with timetable updates but decides not to apply the changes.* *Software Reaction:* *1. The system retains the original timetable without any modifications.*		
Post Conditions		
Step#	***Description***	
1.	The user has successfully executed the email action as intended.	
2.	The user's timetable is updated according to the changes approved in response to the email.	
Use Case Cross referenced	None	

5.3 Email Summarization

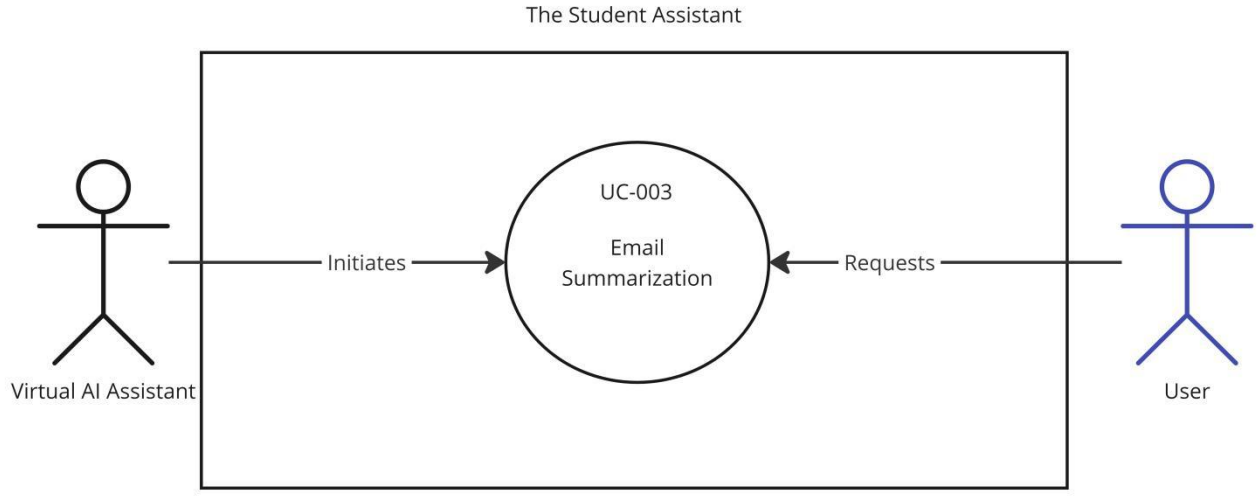

Figure 12 Use Case 3

Description:

This use case involves the automatic detection of emails, the assessment of their relevance to the user, and the generation of email summaries when deemed necessary.

Use case Id: U C - 0 0 3		
Use case Id:	UC-003	
Actors: Primary Actor: Virtual AI Assistant Secondary Actor: User		
Feature:	Email Management	
Pre-condition:	-The Virtual AI Assistant is operational. -The user has access to their email inbox.	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	The Virtual AI Assistant continuously monitors the user's email inbox. When a new email arrives, the system assesses its content and relevance to the user's interests.	If the email is determined to be of interest, the system proceeds to the email summarization process. If the email is not of interest, the system does not initiate summarization.
2.	For emails deemed relevant, the system analyzes the content. The system generates a concise and informative summary of the email content.	The system provides the user with the email summary.
Alternate Scenarios:		
1a: *Action:* *1. The system misclassifies an email's relevance.* *Software Reaction:* *1. The user can provide feedback and correct the system's assessment.* ***2a:*** *Action:* *1. The user receives an email summary but requires further details.* *The user can request the system to provide the full email content.* *Software Reaction:* *1. The system displays the complete email content to the user.*		
Post Conditions		
Step#	***Description***	

1.	The system continuously monitors the user's email inbox and assesses email relevance.
2.	The system provides email summaries for relevant emails.
Use Case Cross referenced	None

5.4 Information Retrieval

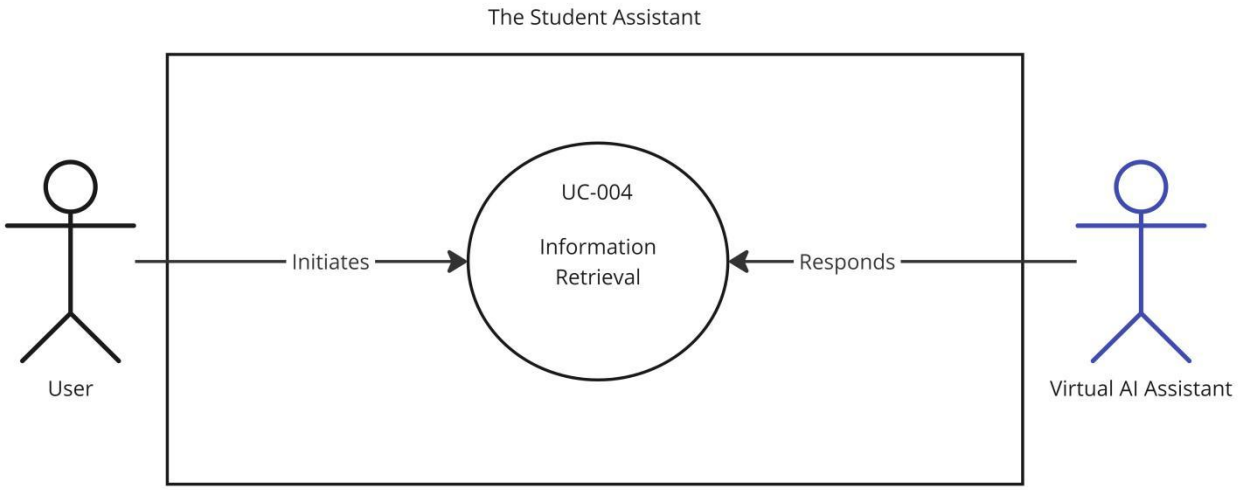

Figure 13 Use Case 4

Description:

This use case involves the Virtual AI Assistant retrieving information in response to user queries and providing relevant responses.

Use case Id: U C - 0 0 4	
Use case Id:	UC-004
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Virtual AI Assistant***	
Feature:	*Virtual AI Assistant*

Pre-condition:	-The Virtual AI Assistant is operational. -The user has access to the Virtual AI Assistant.

Scenarios

Step#	***Action***	***Software Reaction***
1.	The user initiates a query or request by interacting with the Virtual AI Assistant.	The Virtual AI Assistant processes the user's query. The system searches for relevant information in its database or external sources.
2.	The system identifies relevant information based on the query and sources. The system generates a response to the user's query.	The Virtual AI Assistant provides the user with a response, which may include text, voice, or multimedia content.

Alternate Scenarios:

1a:
Action:
1. The system cannot find relevant information.
Software Reaction:
1. The Virtual AI Assistant informs the user that it couldn't locate the requested information.

2a:
Action:
1. The user requests additional information or clarification.

Software Reaction:

1. The system responds with further details or explanations as requested.

Post Conditions

Step#	***Description***
1.	The user initiates a query, and the system searches for relevant information.
2.	The system provides the user with a response to their query.

Use Case Cross referenced	None

5.5 Manage Expenses

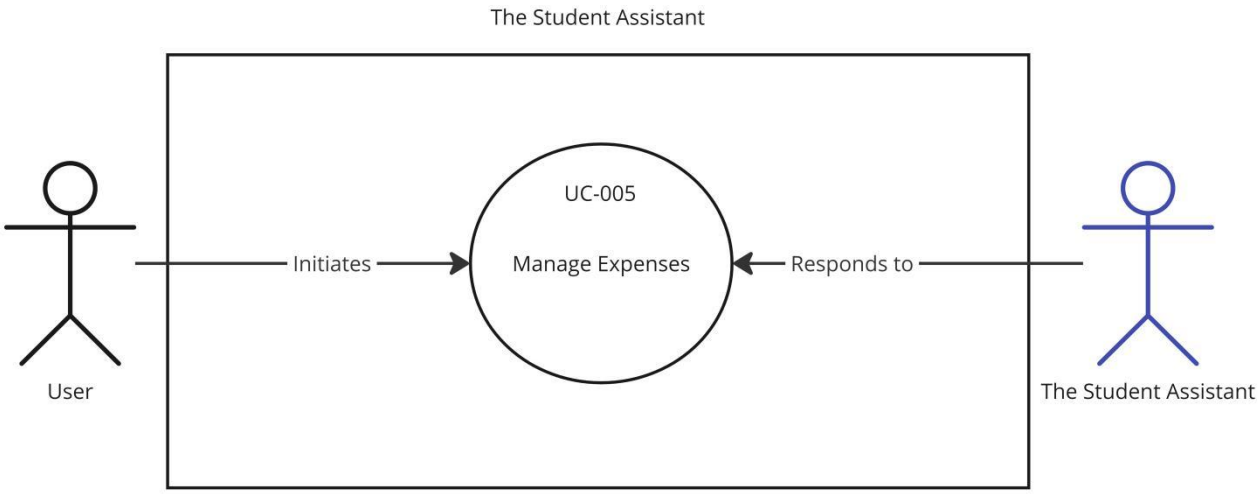

Figure 14 Use Case 5

Description:

This use case involves the management of expenses by users, allowing them to record, categorize, and track their financial transactions within the app.

Use case Id: U C - 0 0 5		
Use case Id:	UC-005	
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Student Assistant App***		
Feature:	*Financial Tracker*	
Pre-condition:	The user has registered and logged into the Student Assistant App. The user has access to the Financial Tracker feature.	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	The user initiates the "Manage Expenses" use case. The user selects the option to record a new expense.	The system displays a form for entering expense details, including the date, amount, category, and description. The user enters the expense information and submits it.
2.	The system stores the expense information in the user's financial records. The system updates the user's financial	The user is provided with a confirmation of the expense entry, and their updated financial balance is displayed.

	balance based on the recorded expense.	
Alternate Scenarios:		
1a: *Action:* *1. The user enters incorrect or incomplete expense information.* *Software Reaction:* *1. The system prompts the user to correct the information and resubmit the expense entry.* ***2a:*** *Action:* *1. The user wishes to edit or delete an existing expense entry.* *Software Reaction:* *1. The user can select an expense from their records and edit or delete it.*		
Post Conditions		
Step#	***Description***	
1.	The user records a new expense, and the system updates the financial records and balance.	
2.	The user can view and manage their expenses, including editing and deleting entries.	
Use Case Cross referenced	None	

5.6 Budget Management

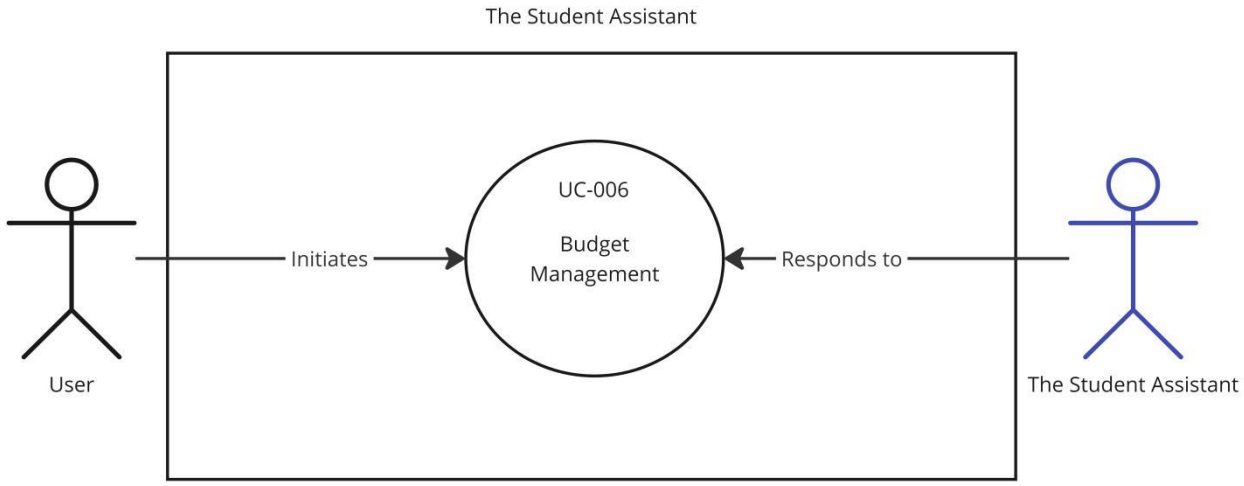

Figure 15 Use Case 6

Description:

This use case involves the management of budgets by users, allowing them to set, edit, and track their financial goals and limits within the app.

Use case Id: U C - 0 0 6		
Use case Id:	UC-006	
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Student Assistant Application***		
Feature:	*Financial Tracker*	
Pre-condition:	-The user has registered and logged into the Student Utility App. -The user has access to the Financial Tracker feature.	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	The user initiates the "Budget Management" use case. The user selects the option to set a new budget.	The system displays a form for entering budget details, including the budget name, amount, and associated categories. The user enters the budget information and submits it.
2.	The system stores the budget information in the user's financial records. The system provides tools for tracking expenses against the budget.	The user is provided with confirmation of the budget creation and access to tools for tracking expenses against the budget..
Alternate Scenarios:		
1a: *Action:* *1. The user enters incorrect or incomplete budget information.* *Software Reaction:* *1. The system prompts the user to correct the information and resubmit the budget creation.* ***2a:*** *Action:* *1. The user wishes to edit or delete an existing budget.* *Software Reaction:* *1. The user can select a budget from their records and edit or delete it.*		
Post Conditions		
Step#	***Description***	

1.	The user sets a new budget, and the system updates the financial records.
2.	The user can view and manage their budgets, including tracking expenses against them.
Use Case Cross referenced	None

5.7 View Timetable

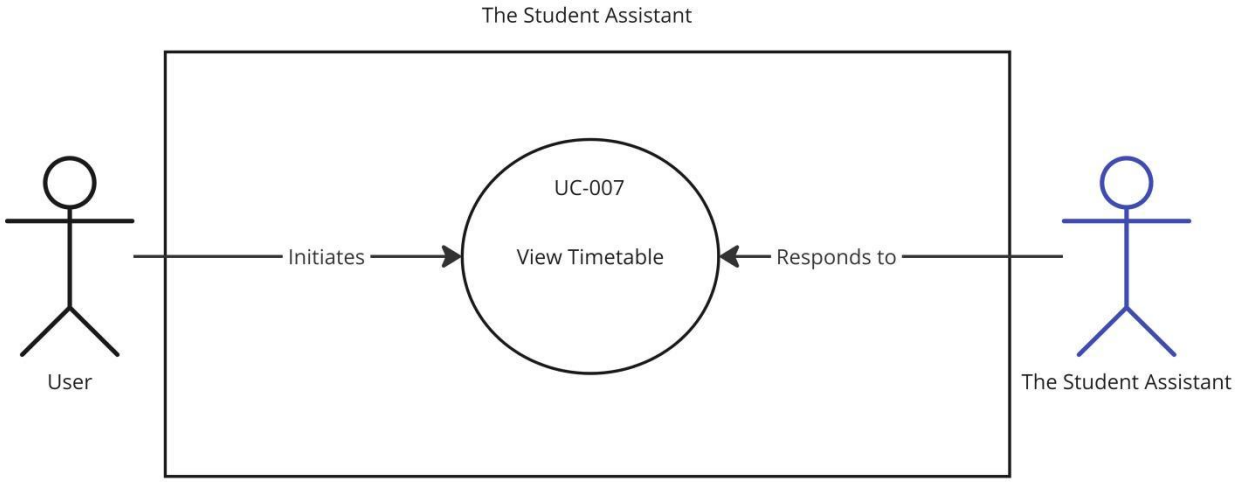

Figure 16 Use Case 7

Description:

This use case allows users to view their class schedules and timetable information within the Student Assistant App.

Use case Id: U C - 0 0 7	
Use case Id:	UC-007
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Student Assistant Application***	
Feature:	*Time Table Management*
Pre-condition:	-The user has registered and logged into the Student Utility App. -The user has access to the Time Table Management feature.

Scenarios		
Step#	**Action**	**Software Reaction**
1.	The user initiates the "View Time Table" use case. The user selects the option to view their time table.	The system retrieves the user's class schedules and timetable information. The system displays the user's class schedule with details such as class names, instructors, times, and locations.
Alternate Scenarios:		
1a: *Action:* *1. The user encounters a technical issue or receives an error while attempting to view the time table.* *Software Reaction:* *1. The system displays an error message and provides instructions to resolve the issue or contact support.*		
Post Conditions		
Step#	**Description**	
1.	The user successfully views their time table and class schedules.	
Use Case Cross referenced	None	

5.8 Add Class/Exam to Timetable

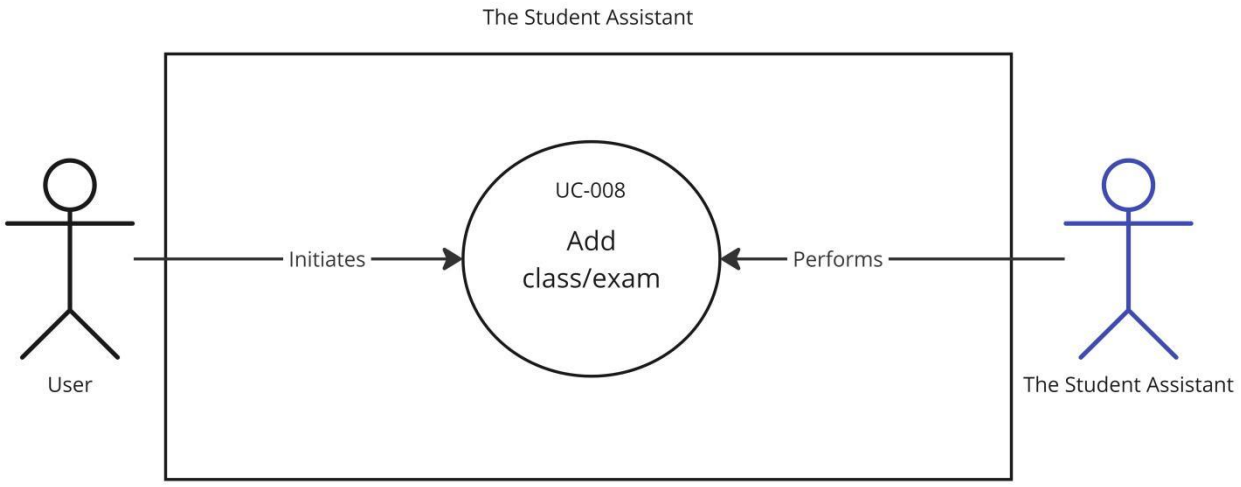

Figure 17 Use Case 8

Description:

This use case allows users to add a new class or exam to their timetable within the Student Assistant Application.

<table>
<tr><th colspan="3">Use case Id: U C - 0 0 8</th></tr>
<tr><td>Use case Id:</td><td colspan="2">UC-008.</td></tr>
<tr><td colspan="3">Actors:

- Primary Actor: User
- Secondary Actor: Student Assistant Application</td></tr>
<tr><td colspan="3">Feature: Time Table Management</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">-The user has registered and logged into the Student Assistant App.
-The user has access to the Time Table Management feature.
-The user has a class or exam that they want to add to their timetable.</td></tr>
<tr><td colspan="3">Scenarios</td></tr>
<tr><td>Step#</td><td>Action</td><td>Software Reaction</td></tr>
<tr><td>1.</td><td>The user initiates the "Add Class/Exam to Timetable" use case.
The user selects the option to add a new class or exam to their timetable.</td><td>The system displays a form for entering class/exam details, including the class/exam name, date, time, instructor, and location.
The user enters the class/exam information and submits it.</td></tr>
<tr><td>2.</td><td>The system stores the class/exam information in the user's timetable.
The system updates the user's timetable to include the new class/exam.</td><td>The user is provided with confirmation of the class/exam addition, and the timetable is updated accordingly.</td></tr>
<tr><td colspan="3">Alternate Scenarios:</td></tr>
<tr><td colspan="3">1a:
Action:
1. The user enters incorrect or incomplete class/exam information.

Software Reaction:
1. The system prompts the user to correct the information and resubmit the class/exam addition.

2a:
Action:
1. The user decides to cancel or discard the class/exam addition.

Software Reaction:
1. The system cancels the addition and returns the user to their previous state in the app.</td></tr>
<tr><td colspan="3">Post Conditions</td></tr>
<tr><td>Step#</td><td colspan="2">Description</td></tr>
<tr><td>1.</td><td colspan="2">The user successfully adds a new class/exam to their timetable, and the system</td></tr>
</table>

	updates the timetable accordingly.
Use Case Cross referenced	None

5.9 Edit Class/Exam in Timetable

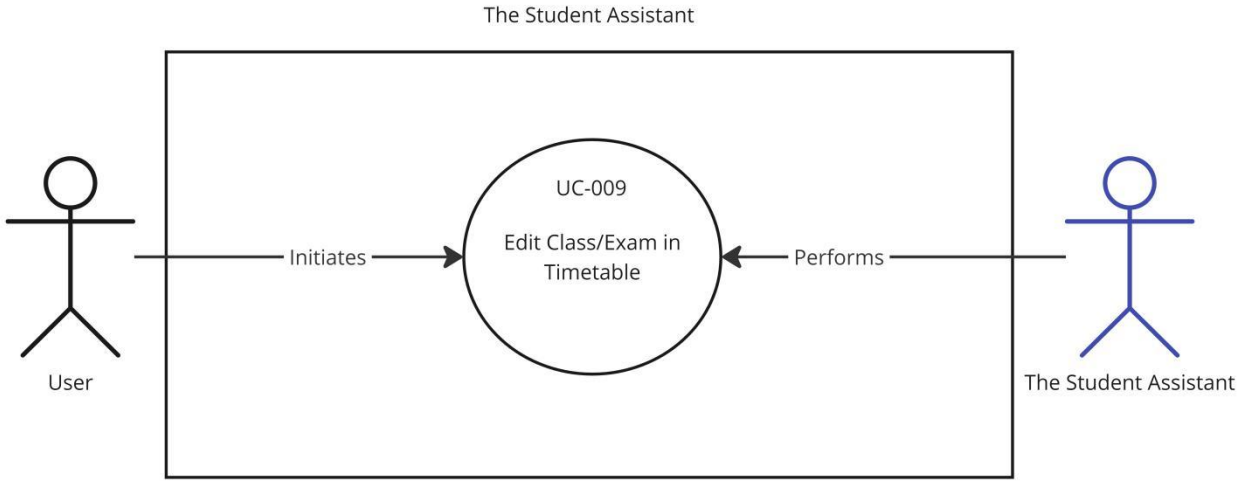

Figure 18 Use Case 9

Description:

This use case allows users to edit the details of a class or exam in their timetable within the Student Assistant Application.

Use case Id: U C - 0 0 9	
Use case Id:	UC-009
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Student Assistant Application***	
Feature:	*Time Table Management*
Pre-condition:	-The user has registered and logged into the Student Assistant App. -The user has access to the Time Table Management feature. -The user has at least one class or exam in their timetable.
Scenarios	

Step#	***Action***	***Software Reaction***
1.	The user initiates the "Edit Class/Exam in Timetable" use case. The user selects the class or exam they want to edit from their timetable.	The system displays the current details of the selected class/exam and provides options to edit specific fields such as class/exam name, date, time, instructor, and location.
2.	The user makes the desired edits to the class/exam details. The user submits the edited information.	The system updates the class/exam details in the user's timetable.
Alternate Scenarios:		
1a: *Action:* *1. The user decides to cancel or discard the edits.* *Software Reaction:* *1. The system cancels the edits and returns the user to their previous state in the app.*		
Post Conditions		
Step#	***Description***	
1.	The user successfully edits the class/exam details in their timetable, and the system updates the timetable accordingly.	
Use Case Cross referenced	None	

5.10 Delete Class/Exam in Timetable

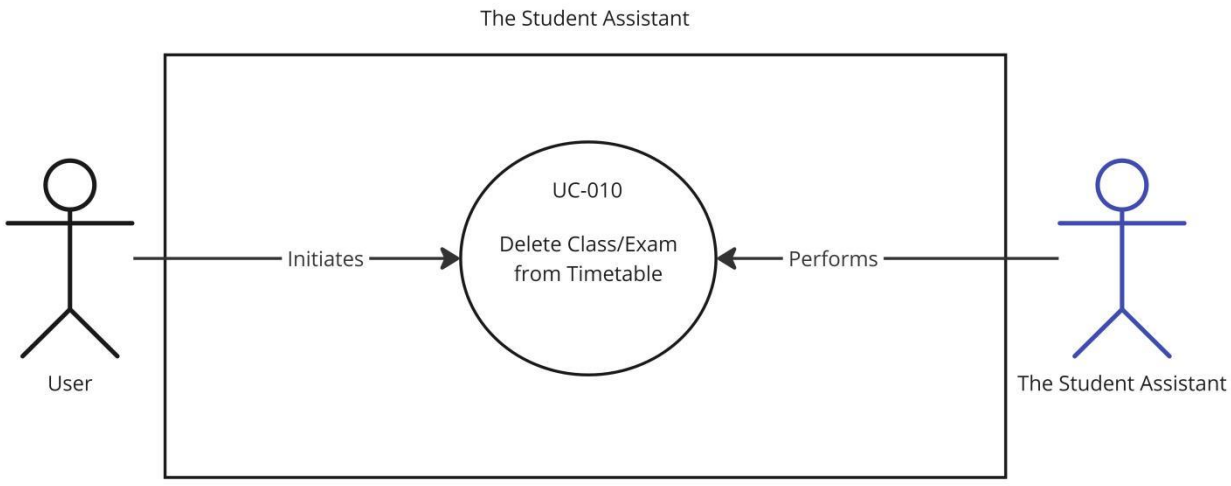

Figure 19 Use Case 10

Description:

This use case allows users to remove a class or exam from their timetable within the Student Assistant Application.

Use case Id: U C - 0 1 0		
Use case Id:	UC-010	
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Student Assistant Application***		
Feature:	*Time Table Management*	
Pre-condition:	-The user has registered and logged into the Student Utility App. -The user has access to the Time Table Management feature. -The user has at least one class or exam in their timetable..	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	The user initiates the "Delete Class/Exam from Timetable" use case. The user selects the class or exam they want to delete from their timetable.	The system prompts the user to confirm the deletion.
2.	The user confirms the deletion.	The system removes the selected class/exam from the user's timetable.
Alternate Scenarios:		
1a: *Action:* *1. The user decides not to delete the class/exam.* *Software Reaction:* *1. The system cancels the deletion and returns the user to their previous state in the app.* ***2a:*** *Action:* *1. User selects an existing filtering rule for editing.* *2. User modifies the filtering criteria or action.* *3. User deletes the selected filtering rule.* *Software Reaction:* *1. The system updates the rule with the new criteria or action.* *2. The system removes the deleted rule from the list.*		
Post Conditions		
Step#	***Description***	
1.	The user successfully deletes the selected class/exam from their timetable, and the	

	system updates the timetable accordingly.
Use Case Cross referenced	None

5.11 Manage User Profile

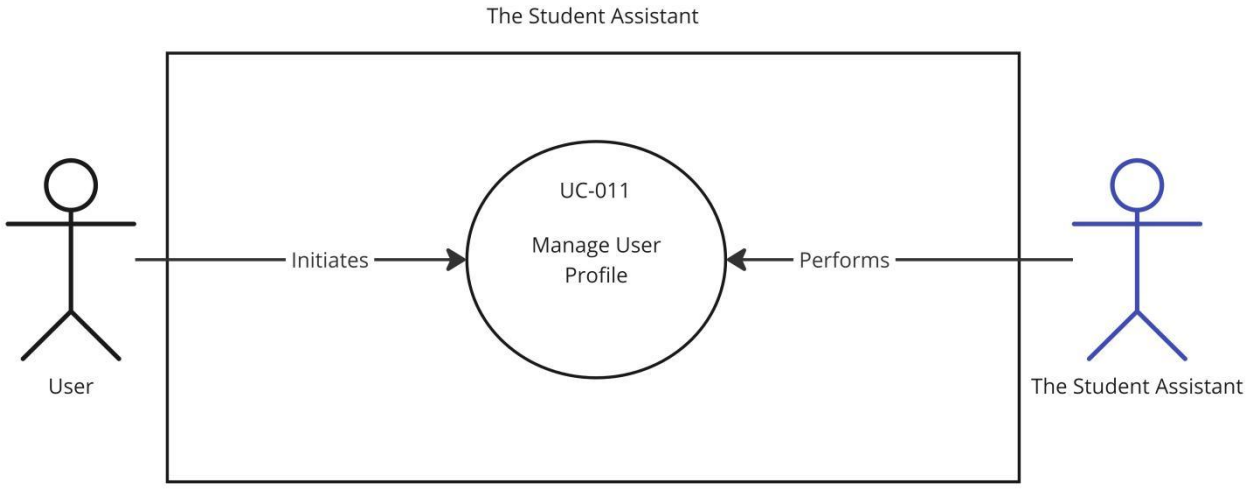

Figure 20 Use Case 11

Description:

This use case allows users to create, update, and manage their profiles within the Student Assistant Application.

Use case Id: U C - 0 1 1		
Use case Id:	UC-011	
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Student Assistant Application***		
Feature:	*User Profile Management*	
Pre-condition:	-The user has registered and logged into the Student Utility App. -The user has access to the User Profile Management feature.	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	The user initiates the "Manage User Profile" use case. The user can choose to create a new	The system displays the user profile form for creating or editing. If creating a new profile, the user fills in the

	profile or edit an existing profile.	required details. If editing an existing profile, the user can modify the existing information.
2.	The user saves the changes.	The system validates the information. If the information is valid, the system updates the user's profile. If there are validation errors, the system provides appropriate error messages.
Alternate Scenarios:		
1a: *Action:* *1. The user cancels the profile creation or editing.* *Software Reaction:* *1. The system cancels the operation and returns the user to their previous state in the app.*		
Post Conditions		
Step#	***Description***	
1.	The user successfully creates a new profile or updates an existing profile, and the system saves the changes.	
Use Case Cross referenced	None	

5.12 Manage Notifications

Description:

This use case allows users to manage notifications, including enabling, disabling, and customizing notification settings within the Student Assistant Application.

Use case Id: U C - 0 1 2		
Use case Id:	UC-012	
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Student Assistant Application***		
Feature:	***Notification Management***	
Pre-condition:	The user has registered and logged into the Application. The user has access to the Notification Management feature.	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	The user initiates the "Manage	The system displays the notification settings

	Notifications" use case.	screen, including options to enable or disable notifications. The user can further customize notification preferences, such as notification sounds, frequency, and notification categories.
2.	The user customizes their notification preferences.	The system saves the user's preferences and updates the notification settings accordingly.
Alternate Scenarios:		
1a: *Action:* *1. The user chooses to disable notifications entirely.* *Software Reaction:* *1. The system disables all notifications for the user.*		
Post Conditions		
Step#	***Description***	
1.	The user successfully customizes their notification preferences, and the system saves the changes.	
Use Case Cross referenced	None	

5.13 Quick Search

Description:

This use case allows users to manage notifications, including enabling, disabling, and customizing notification settings within the Student Assistant Application.

Use case Id: U C - 0 1 3		
Use case Id:	UC-013	
Actors: ***- Primary Actor: User*** ***- Secondary Actor: Student Assistant Application***		
Feature:	***Quick Search***	
Pre-condition:	The user has registered and logged into the Student Utility App. The user has access to the Quick Search feature.	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	The user initiates the "Quick Search" use case. The user enters a search query in the	The system processes the search query. The system displays a list of search results, including relevant information, resources, or

	search bar.	events matching the query.
2.	The user selects a search result.	The system provides additional details about the selected result or navigates the user to the relevant section of the app.
Alternate Scenarios:		
1a: *Action:* *1.The user enters an invalid or non-matching search query.* *Software Reaction:* *2. The system displays a message indicating that no results were found for the query.*		
Post Conditions		
Step#	***Description***	
1.	The user successfully finds and interacts with information, resources, or events using the Quick Search feature.	
Use Case Cross referenced	None	

6. Other Nonfunctional Requirements

6.1 Performance Requirements

❖ Response Time:

The Student Assistant Application must respond to user interactions within 1 second for the majority of actions, ensuring a seamless and responsive user experience.
The AI Assistant's response time for routine inquiries should not exceed 7 seconds to maintain an efficient interaction.

❖ Scalability:

The system should be able to accommodate a minimum of 4000 concurrent users without significant degradation in performance.
The application should handle data growth without performance degradation, supporting a minimum of 100,000 user accounts and 1,000,000 records in the database.

❖ Data Retrieval:

Data retrieval from external sources, such as the university email server, should occur within 10 seconds for routine queries.

- **Email Summarization:**

The email summarization process should be completed in real-time as emails are received, ensuring that users receive summarized content promptly.

- **AI Assistant's Processing Time:**

The AI Assistant should process complex user queries and provide responses within 7 seconds to maintain efficient interactions.

- **Database Queries:**

Database queries for user data and timetables should execute in less than 10 seconds.

- **Mobile Application Launch Time:**

The mobile application should launch and become operational within 7 seconds of user interaction.

6.2 Safety Requirements

- **Data Security:**

The Student Utility App must implement robust data security measures to protect sensitive user information, including personal and financial data, to prevent unauthorized access and data breaches.

- **User Privacy:**

The application should adhere to data privacy regulations and protect user privacy by ensuring that personal information is not shared with third parties without explicit consent.

- **Email Privacy:**

Users' email content and communications must be kept private and not used for any purpose other than the intended functionality of the app.

- **Timetable Accuracy:**

The system should ensure the accuracy of timetables, as incorrect scheduling or location information could lead to confusion and disruption for users.

- **AI Assistant Behavior:**

The AI Assistant should be programmed to provide information and assistance that adheres to ethical and safety standards. It must not engage in harmful or inappropriate conversations, provide sensitive information to unauthorized users, or engage in malicious activities.

- **Backup and Data Recovery:**

Regular automated backups of user data and application state should be maintained to prevent data loss. Additionally, a data recovery mechanism must be in place to restore lost data in case of an unexpected failure.

- **Legal and Regulatory Compliance:**

The application must comply with all applicable laws, regulations, and policies related to user safety, data protection, and privacy.

- **Certifications:**

The Student Utility App should adhere to safety and security certifications if required by relevant authorities or educational institutions.

- **Emergency Procedures:**

In case of system failures or issues affecting users, the application should provide clear and concise error messages and guidance on what actions the user should take.

- **User Account Protection:**

User accounts must be protected against unauthorized access, requiring strong passwords and the option for multi-factor authentication for added security.

❖ **Safeguards and Error Handling:**

The application should have safeguards in place to handle errors gracefully and prevent potential system failures or data loss.

6.3 Security Requirements

❖ **User Authentication:**

The Student Utility App should implement a secure user authentication system, requiring users to provide valid credentials, such as usernames and passwords, to access their accounts.

❖ **Data Encryption:**

All sensitive data transmitted between the app and the server, including personal and financial information, must be encrypted using industry-standard encryption protocols.

❖ **Secure Communication:**

The application should utilize secure communication protocols (e.g., HTTPS) to protect data during transit.

❖ **Data Privacy:**

The app should adhere to data privacy regulations, ensuring that user data is not shared with third parties without explicit consent.

❖ **User Authorization:**

Authorization mechanisms should be in place to control what actions users are allowed to perform within the app, preventing unauthorized access to sensitive functionality.

- **Password Policies:**

Users should be required to create strong passwords, and password policies (e.g., complexity requirements) should be enforced.

- **Secure APIs:**

Any third-party APIs or services (e.g., Gmail API) used by the application must be secure and adhere to best practices for data protection.

- **Security Testing:**

Conduct security assessments, and vulnerability scanning, to identify and address security weaknesses during testing phase.

- **Incident Response:**

Establish an incident response plan to address security incidents promptly, minimize potential damage, and notify affected users when necessary.

- **Compliance with Regulations:**

Ensure compliance with data protection regulations, industry-specific security standards, and any applicable laws and regulations related to data security and user privacy.

- **Certifications:**

If required by authorities or educational institutions, the Student Utility App should obtain and maintain security and privacy certifications.

- **Secure Development Practices:**

Ensure that secure development practices are followed during the software development life cycle to prevent common vulnerabilities.

6.4 Software Quality Attributes

❖ **Usability:**

The Student Utility App should provide a user-friendly and intuitive interface to ensure that users can easily navigate and utilize its features. Usability will be assessed through user feedback and testing to achieve a user satisfaction rating of at least 85%.

❖ **Reliability:**

The application should operate consistently and reliably. It should have a system uptime of at least 99.9%, ensuring minimal downtime for maintenance or unexpected failures.

❖ **Performance:**

The Student Utility App should be responsive and provide efficient performance, with a maximum response time of 1 second for critical user interactions, such as loading a class schedule or summarizing an email.

❖ **Availability:**

The system should be available 24/7, with planned maintenance windows communicated in advance. The target availability is 99.9% to ensure access at all times.

❖ **Scalability:**

The application should be scalable to accommodate an increasing number of users and data. It should be capable of handling a 20% growth in user base and data volume without significant performance degradation.

❖ **Maintainability:**

The software should be designed with modularity and maintainability in mind. Code changes and updates should be made with ease, and the development team should be able to release monthly updates or patches.

- **Security:**

The application should have a strong security architecture and robust authentication mechanisms to protect user data. Vulnerability assessments should be conducted at least twice a year to ensure the system's security posture.

- **Testability:**

The software should be designed with testability in mind. Unit testing and integration testing should be conducted for each feature, and a minimum code coverage of 80% should be maintained.

- **Adaptability:**

The application should be adaptable to changing user needs and feature requests. New features or enhancements should be integrated within a two-week development cycle.

- **Correctness:**

The software should be free from critical defects. The application should have less than 1 critical defect per 1,000 lines of code, as measured by static code analysis and user feedback.

- **Robustness:**

The app should be resilient to unexpected conditions or inputs, with error messages and graceful degradation in case of system failures.

- **Portability:**

The application will also be converted to support IOS Environment if the project goes on to be running with optimistic time.

6.5 Business Rules

❖ **User Authentication:**

Users must log in to access their profiles, ensuring secure access to personal information.

❖ **Class Scheduling:**

Classes must not overlap in a student's timetable, preventing scheduling conflicts.
Faculty can schedule and reschedule classes as necessary, with notifications sent to affected students.

❖ **Email Summarization:**

The system automatically identifies emails of interest based on predefined criteria and summarizes them.
Users can provide feedback to improve the email summarization algorithm.

❖ **Expense Tracking:**

Users can add and categorize expenses, helping them manage their finances.
Expenses must be categorized appropriately, and users can set monthly budgets.

❖ **Security and Data Privacy:**

User data must be protected in compliance with relevant data protection regulations.
User-specific data is accessible only to the respective user and authorized administrators.

❖ **Notifications:**

Users receive timely notifications about schedule updates, reminders, and other relevant information.
Notifications are delivered through the app and may include email notifications for important updates.

❖ **Cloud Integration:**

The application securely integrates with cloud services for data storage, ensuring data availability and backup.

❖ **User Feedback:**

Users can provide feedback, report issues, or suggest improvements within the application.

❖ **Timetable Update Approval:**

Users must approve any changes to their class schedules initiated by faculty members. Faculty members require approval from the relevant department or program officers for significant changes to class schedules.

❖ **Financial Data Accuracy:**

The system ensures accurate financial data and calculations for expense tracking and budget management.

❖ **Quick Search Limitations:**

The Quick Search feature provides a limited number of results, ensuring a manageable user experience.

7. Other Requirements

7.1 Database Requirements:

❖ The system shall use a secure and reliable database management system for data storage and retrieval.
❖ Data backups and routine maintenance must be performed to ensure data integrity and availability.
❖ The database design shall support efficient data retrieval and querying.

7.2 Internationalization and Localization:

- The application shall support multiple languages and regional settings to cater to a diverse user base.
- Users can select their preferred language and locale, affecting the application's text and date/time formats.

7.3 Legal and Compliance Requirements:

- The system shall comply with all relevant data protection and privacy laws, including GDPR and local regulations.
- Users must agree to terms and conditions and privacy policies during registration.

7.4 Performance Testing:

- The system shall undergo performance testing to ensure it can handle expected user loads efficiently.
- Response times and resource utilization must meet specified criteria under typical usage scenarios.

7.5 Documentation:

- The project shall include comprehensive documentation for developers, system administrators, and end-users.
- Documentation shall be regularly updated to reflect system changes and improvements.

7.6 Third-Party Services:

- The application may integrate with third-party services for features like email summarization and cloud storage.

- The integration shall adhere to the terms and conditions of these services.

Appendix A: Glossary

- **SRS:** Software Requirements Specification.
- **FYP:** Final Year Project.
- **AI:** Artificial Intelligence.
- **IBA:** Institute of Business Administration.
- **RAM:** Random Access Memory.
- **API:** Application Programming Interface.
- **GDPR:** General Data Protection Regulation.

End of Document