UTM
UNIVERSITI TEKNOLOGI MALAYSIA

FACULTY OF COMPUTING
UTM Johor Bahru

SECJ 2203: Software Engineering

Semester 01, 2024/2025

System Documentation (SD)

Serene System

Version 3.0

19-01-2025

Prepared by: Serenity Pvt. Ltd.

Presentation link :

https://drive.google.com/drive/folders/16hBZCoCXm6o-AIuedl-ucW1ZHdEEHbuf?usp=share_link

Revision Page

a. **Overview**

The current version of the SD includes a detailed description of the Digital Mental Health Literacy Hub, outlining the project's needs, proposed system, chosen methodology, and implementation plan. It emphasizes addressing mental health challenges faced by higher education students through innovative digital tools and resources.

b. **Target Audience**

The SD is targeted at:

- Higher education students
- Mental health professionals
- Administrative staff
- University stakeholders

c. **Project Team Members**

List the team members in a table by stating their roles and the status for each assigned task e.g. by sections for this SD version (complete, partially complete, incomplete). If the assigned tasks are not done and have been assigned to other team members, state accordingly.

Member Name	**Role**	**Task**	**Status**
Fares Hamid Abdo Mohamed	Developer	Developed project NABC and implemented features	Complete
Anjum Siddiqua Tanveer Siddiqui	Project Manager	Monitor group progress	Complete

Ivor Barrie Jaffery	Analyst	Conducted case study and researched existing systems	Complete
Humaira Sheyla Nurfayza	Designer	Designed system interface and features & references	Complete

d. Version Control History

Version	**Primary Author(s)**	**Description of Version**	**Date Completed**
1.0	Anjum Siddiqua Tanveer	Completed the SD	16//11/24
2.0	Anjum Siddiqua Tanveer	Added use case diagrams	12/12/24
3.0	Anjum Siddiqua Tanveer	Added missing use case diagrams, package diagrams and Test cases	19/01/25

Table of Contents

1. Introduction

1.1 Purpose

The purpose of the System Documentation (SD) is to serve as a comprehensive reference for the design, requirements and testing of the system. This document integrates the Software Requirements Specification (SRS), System Design Document (SDD) and System Testing Document (STD) to provide a guide that is united for all stakeholders, including developers, testers and project managers. The SD plan the scope and objectives of the system, specifying functional, non-functional requirements, system architecture and testing strategies. It also includes models such as use case descriptions, sequence diagrams, and activity diagrams to explain system processes and ensure consistency throughout the development lifecycle (Sommerville, 2016). This documentation is for maintaining the essential consistency among all stakeholders and supporting the future enhancements or maintenance. This SD is intended for a very wide range of audience, including technical teams which are responsible for system implementation, quality assurance teams verifying requirements with compliance, and the managers of the project overseeing system delivery. By committing to the guidelines outlined, the SD ensures a structured approach to system development, aligning with best practices in software engineering (Pressman, 2014).

1.2 Scope

The scope of this project Serene System is to provide a digital platform for higher education students to address their mental health challenges through their educational life. The system connects students with qualified experts and offers them the tools and resources to manage their mental health, the system acts as a central location for promoting mental health literacy.

The main users of the system consist of:

Students in higher education: Participate in peer-support forums, educational materials, and self-assessment tools.

Mental Health Professionals: Provide counseling and manage student cases through secured virtual sessions.

Administrative Staff and Faculty Members: Monitor engagement analytics and promote the initiative mental health.

The Serene System scope of the web-based platform includes the following:

Software Architecture and Compatibility:

The system will be designed to ensure compatibility with various devices and browsers, including desktops, tablets and smartphones.

Best Practices in Software Engineering:

Development will involve coding standards, testing procedures, and quality assurance practices for maintaining the high-performance and reliability.

Development Approach:

The system will be using the agile methodology, allowing repeated improvements based on stakeholders feedback.

Testing and Validation:

- The system will keep rigorous testing using both manual and automated methods to meet functional and nonfunctional requirements.
- Maintenance and Updates:
- This project will continue to satisfy the changes of user needs with ongoing support, which includes bug patches and system updates.
- The aim of this system is to offer an inclusive, user-friendly environment that equips stakeholders with the tools they need to improve mental health awareness, accessibility and support.

1.3 Definitions, Acronyms and Abbreviation

Term	Definition
SD	System Documentation. A comprehensive document outlining the requirements, design, and testing of the system.
SRS	Software Requirements Specification. A document that describes the functional and non-functional requirements of the system.
SDD	System Design Document. A document detailing the architectural and design components of the system.
STD	System Testing Document. A document describing the testing methods, strategies, and outcomes to ensure the system performs as expected.
Agile	A software development methodology focused on iterative progress, flexibility, and frequent user feedback.
Mental Health Literacy	The ability to recognize, manage, and seek help for mental health challenges effectively.
Higher Education Students	Primary users of the system who access tools and resources to improve their mental health.
Self-Assessment Tools	Interactive modules that allow users to evaluate their mental health status independently.
Virtual Counseling	Online one-on-one sessions with certified mental health professionals.
Gamification	The use of game-like elements to make mental health education engaging and interactive.

UI/UX	User Interface and User Experience. The design and usability aspects of the system to ensure it is user-friendly and accessible.
Dashboard	A visual interface providing administrative users with insights into system usage and student engagement.

1.4 References

purpose references

Sommerville, I. (2016).

- Book Name: *Software Engineering* (10th ed.).
- Publisher: Pearson.
- URL/Info: Pearson - Software Engineering

Pressman, R. S. (2014).

2. Book Name: *Software Engineering: A Practitioner's Approach* (8th ed.).
3. Publisher: McGraw-Hill Education.
4. URL/Info: McGraw-Hill Education - Software Engineering

scope references

Sommerville, I. (2016).

- **Book Name**: *Software Engineering* (10th ed.).
- **Publisher**: Pearson.
- **URL/Info**: Pearson - Software Engineering

Pressman, R. S. (2014).

- **Book Name**: *Software Engineering: A Practitioner's Approach* (8th ed.).
- **Publisher**: McGraw-Hill Education.
- **URL/Info**: McGraw-Hill Education - Software Engineering

Agile Alliance.

- **Resource**: Agile Methodology Concepts.
- **URL**: Agile Alliance - Agile 101

WCAG 2.1 (Web Content Accessibility Guidelines).

- **Standard for Accessibility Compliance**.
- **URL**: WCAG 2.1 Guidelines

IEEE Software Engineering Standards.

- **Standard for Software Engineering Practices (e.g., ISO/IEC/IEEE 29148)**.
- **URL**: ISO/IEC/IEEE 29148:2018

Performance and Other Requirements references

ISO/IEC/IEEE 29148:2018

- **Title**: *Systems and Software Engineering — Life Cycle Processes — Requirements Engineering*
- **Details**: This standard outlines best practices for defining functional and non-functional requirements, including performance and system attributes.
- **URL**: ISO/IEC/IEEE 29148:2018

ISO/IEC 25010:2011

- **Title**: *Systems and Software Quality Requirements and Evaluation (SQuaRE) System and Software Quality Models*
- **Details**: Defines quality attributes, including performance, usability, reliability, and security, for evaluating systems.
- **URL**: ISO/IEC 25010:2011

Sommerville, I. (2016)

- **Book Name**: *Software Engineering* (10th ed.).
- **Publisher**: Pearson.
- **Details**: Provides detailed guidance on specifying and analyzing non-functional requirements, such as performance and scalability.
- **URL**: Pearson - Software Engineering

Pressman, R. S. (2014)

- **Book Name**: *Software Engineering: A Practitioner's Approach* (8th ed.).
- **Publisher**: McGraw-Hill Education.
- **Details**: Discusses non-functional requirements, including performance benchmarks, scalability, and maintainability.
- **URL**: McGraw-Hill Education - Software Engineering

Nielsen, J. (1993)

- **Book Name**: *Usability Engineering*.
- **Details**: Provides a foundation for usability requirements, a key aspect of system performance.
- **URL**: Nielsen Norman Group - Usability Engineering

WCAG 2.1 (Web Content Accessibility Guidelines)

- **Details**: Specifies guidelines for accessibility, which is part of usability requirements in performance considerations.
- **URL**: WCAG 2.1 Guidelines

Design Constraints References

ISO/IEC/IEEE 29148:2018

- **Title**: *Systems and Software Engineering — Life Cycle Processes — Requirements Engineering*

- **Details**: Provides guidelines for documenting constraints, including environmental, hardware, security, and compatibility constraints.
- **URL**: ISO/IEC/IEEE 29148:2018

ISO/IEC 25010:2011

- **Title**: *Systems and Software Quality Requirements and Evaluation (SQuaRE) — System and Software Quality Models*
- **Details**: Specifies quality attributes that impact design constraints, such as compatibility, portability, and maintainability.
- **URL**: ISO/IEC 25010:2011

GDPR (General Data Protection Regulation)

- **Details**: Provides legal and regulatory requirements for data protection and privacy, which influence security constraints.
- **URL**: GDPR Official Site

Malaysia Personal Data Protection Act (PDPA), 2010

- **Details**: Sets standards for protecting personal data in Malaysia, relevant to security and legal constraints.
- **URL**: PDPA 2010 - Overview

Nielsen, J. (1993)

- **Book Name**: *Usability Engineering*.
- **Details**: Provides insights into constraints for user interface and system design related to usability.
- **URL**: Nielsen Norman Group - Usability Engineering

WCAG 2.1 (Web Content Accessibility Guidelines)

- **Details**: Defines accessibility requirements, a critical component of design constraints for user interfaces.

- **URL**: WCAG 2.1 Guidelines

Pressman, R. S. (2014)

- **Book Name**: *Software Engineering: A Practitioner's Approach* (8th ed.).
- **Publisher**: McGraw-Hill Education.
- **Details**: Discusses system constraints, including hardware, software, and external factors affecting design.
- **URL**: McGraw-Hill Education - Software Engineering

Diagrams

User story diagrams in UC03,UC09 made by ("plantuml").

- **PlantUML (https://plantuml.com/)**

1.5 Overview

The System Documentation (SD) provides a detailed reference for everyone involved in the development, including testing and maintaining of the Serene System. It combines important documents such as the Software Requirements Specification (SRS), System Design Document (SDD), and System Testing Document (STD). It provides many people including project managers, developers, testers, and other team members with a single resource. The SD keeps all stakeholders informed by ensuring the best practices are followed at every stage of the development process.

The document starts with an introduction that explains its purpose, scope, and the target audience. It talks about the main goals of the Serene System and describes its users such as students, mental health professionals, faculty members, and administrative staff along with the plans to help with mental health issues through features.

The SD gives an overview of the system that explains how it works and the structure behind it. It describes different parts of the system, such as the user interface (UI/UX), dashboard,

and backend. It also outlines the system's main functions and the non-functional requirements. Diagrams also provided including use cases, sequence, and activity diagrams to show how the system works and how different parts interact. The testing part talks about the method used to make sure the system works correctly both in manual and automated.

The SD also covers how the system will be maintained and updated. This includes plans for fixes, updates, and improvements based on feedback. In conclusion, the document emphasizes the importance of following guidelines to ensure the system is successfully implemented. By providing a clear and organized approach, the SD ensures that everyone involved is on the same page and helps the Serene System effectively support mental health in higher education.

2. Specific Requirements

2.1 User Roles

Our project “Serene System” is a Digital Mental Health Literacy Hub which is focused to cater the mental health needs of students studying in higher educational institutes. Even though it is mainly focused on catering to students, there are a lot of user groups involved in this system.

Higher Education Students:

Higher education students are the primary target users of this system since it is developed with the students' mental health in mind.

Roles:

Register and login to the system as a Student.

Access Self-Assessment tools

Use AI powered tools.

Access Interactive Mental Health Modules

Schedule Virtual Sessions

Mental Health Professionals:

Mental health professionals form the basis of the system. They make sure that students can have support that they need anytime, anywhere.

Roles:

Register and login to the system as a Mental health professional.

Verify Professional Certification.

Schedule Virtual Session.

Provide Instant Feedback.

Use AI powered tools.

Administrative Bodies:

Admins overlook the system in general. They provide and maintain the Analytics Dashboard and also identify any risk behavior pattern. They ensure compliance with data protection regulations and support in decision making.

Roles:

Register and login to the system as an Admin

Access Analytics Dashboard

Faculty Members:

The faculty members are like the caretakers of students because they look for high risk students, undergo training to identify and provide support for students.

Roles:

Register and login to the system as a Faculty member.

Use AI powered tools.

Access Interactive Mental Health Modules.

Promote Mental Health Awareness.

2.1.1 User Role 1: Mental Health Professionals

User Need

Mental Health Professionals need a way to access the system.

User Stories

UC01 : *As a* Mental Health Professional I want to verify my professional certificate so that I can ensure the students that I am reliable and trustable and to be able to access the system.

2.1.2 User Role 2 : Higher Education Students

User Need

Higher education students need a way to schedule a virtual meeting with a mental health professional from their own place.

User Stories

UC03 : As a higher education student I should be able to schedule a virtual session with a certified mental health professional so that I don't need to go away from my place or have to visit a clinic.

2.1.3 User Role 3 : Administrative Body

User Need

Admins need a way to get to know the engagement metrics and identify high risk students at a glance.

User Stories

UC04 : A*s an* Admin I want to access the Analytics dashboard so that I can help track overall student engagement, identify high-risk patterns.

2.1.4 User Role 4 : Higher Education Students

User Need

Higher education students need a way to use chatbots that are specially designed for mental health consultation.

User Stories

UC05 : As a higher education student I want to chat with AI powered chatbots regarding my mental health so that I get precise replies as it is focused and developed mainly for mental health concerns.

2.1.5 User Role 5 : Higher Education Students

User Need

Higher education students need a way to educate themselves about mental health.

User Stories

UC08 : As a higher education student I want to educate myself about mental health through interactive learning modules so that I can self-identify or identify any mental health concerns in me or my peers to support myself and them.

2.1.6 User Role 6 : Higher Education Students

User Need

Higher education students need a way to access their health mental scores from their own place.

User Stories

UC09 : As a higher education student I want to access my mental health score so that I can identify any mental health issues that might affect me in a greater way if went unnoticed.

2.1.7 User Role 7 : Faculty Members

User Need

Faculty Members need a way to support my students and their well-being.

User Stories

UC10 : *As a* Faculty member I want to promote mental health awareness among my students so that I can serve as a potential supporter and advocate for student well-being.

2.1.8 User Role 8: All users

User Need

All the users need a secure way to access the system.

User Stories

UC08 : As a user I want to securely access the system so that I can get the benefits of the system but also ensure only verified people use the system making sure it is friendly but not malicious.

2.1.9 User Role 9 : Higher Education Students

User Need

Higher education students need a way to engage in open discussions about mental health concerns while staying anonymous.

User Stories

UC11 : *As a H*igher education student I want to engage in open discussions about mental health concerns while staying anonymous so that I can share my feelings with someone who has similar problems as me to empathize with me.

2.2 System Features

The Serene System is a hub management system that acts as a platform mainly for students, especially university students to access related functions that help monitor their mental health as illustrated in Figure 2.1 below. It is a web-based system that can be accessed no matter the device, whether it be on mobile or on a computer. The detailed description of each use case is tabulated in Table 2.1.

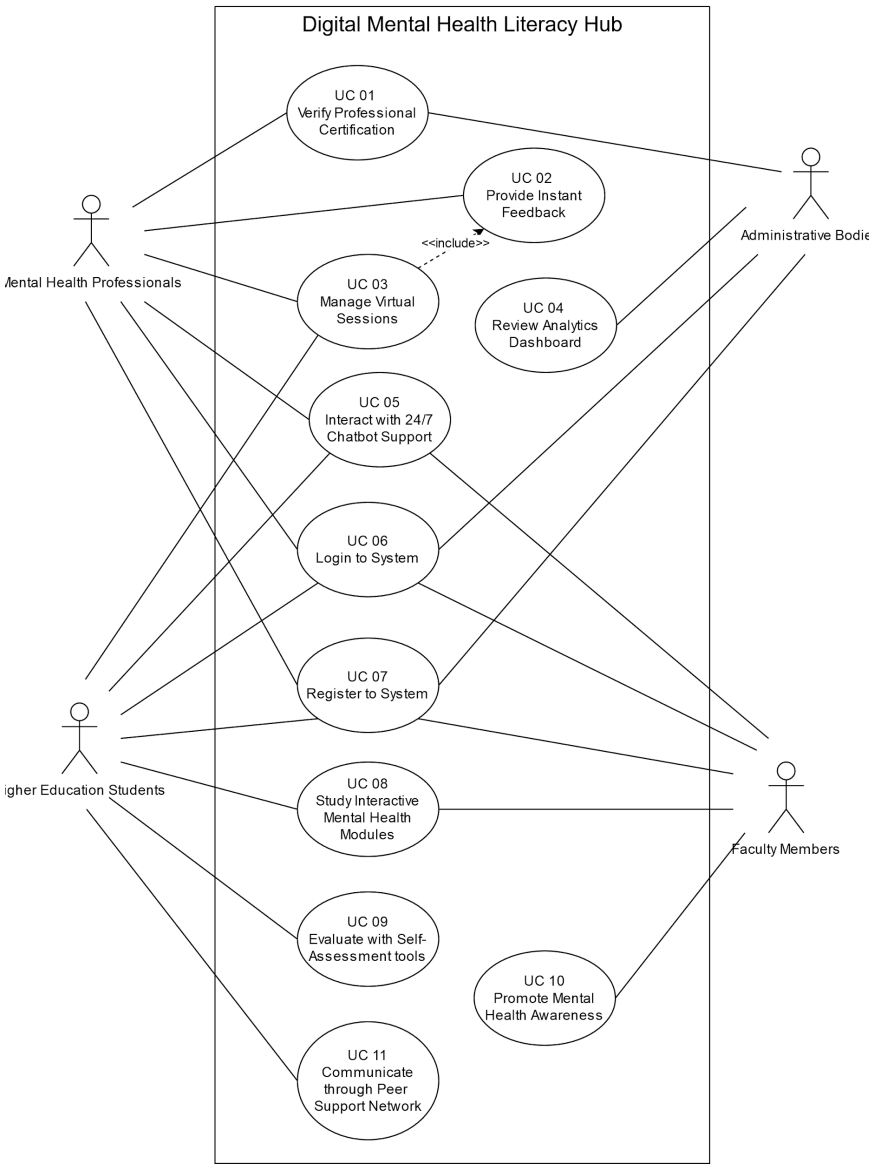


Figure 2.1: Use Case Diagram for Serene System

Use Case	Function	Description
UC 01	Verify Professional	Allows Mental Health Professionals to verify their certifications with administrative bodies before gaining

	Certification	access to their dedicated interface of the system to help students on the platform.
UC 02	Provide Instant Feedback	Allows Mental Health Professionals to be able to provide instant feedback to students after certain virtual sessions.
UC 03	Manage Virtual Sessions	Students can manage virtual sessions with verified Mental Health Professionals if they require a more verbal approach to their mental problems.
UC 04	Review Analytics Dashboard	Administrative bodies can review an analytics dashboard to take a glance at how the platform is currently doing in terms of technicality and also identify high risk patterns among students to send help from the faculty members.
UC 05	Interact with 24/7 Chatbot Support	This use case allows users to interact with chatbot to help either navigate through the platform or suggest actions to take for first-time users of the system
UC 06	Login To System	This use case allows users to login to the system once registered.
UC 07	Register to System	This use case allows first-time users to register to the system with their credentials.
UC 08	Study Interactive Mental Health Modules	Allows access to the interactive mental health modules to gain insight or learn a thing or two about mental health problems.
UC 09	Evaluate with Self Assessment Tools	Allows students to evaluate and make self assessment on their progress to overcome their current mental health problems.
UC 10	Promote Mental Health Awareness	This use case allows faculty members to gain access to tools to promote mental health awareness to the users and to social media.
UC 11	Communicate through Peer Support Network	This use case allows students to connect and communicate with other peers who also seek help regarding their mental health problems and chat either anonymously or not.

Table 2.1: Detailed Description for Each Use Case

Class Diagram

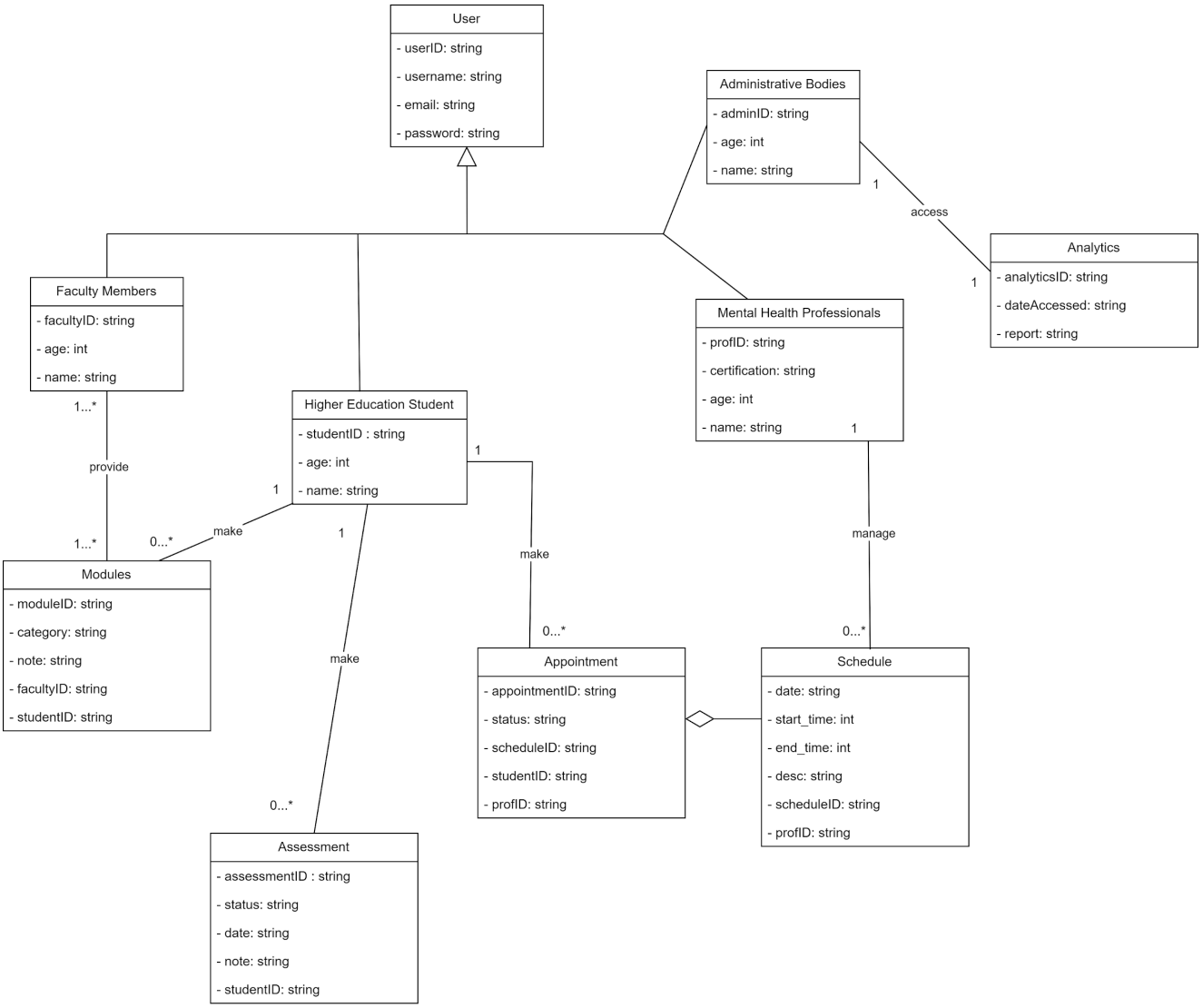


Figure 2.2: Class Diagram for Serene System

State Diagram

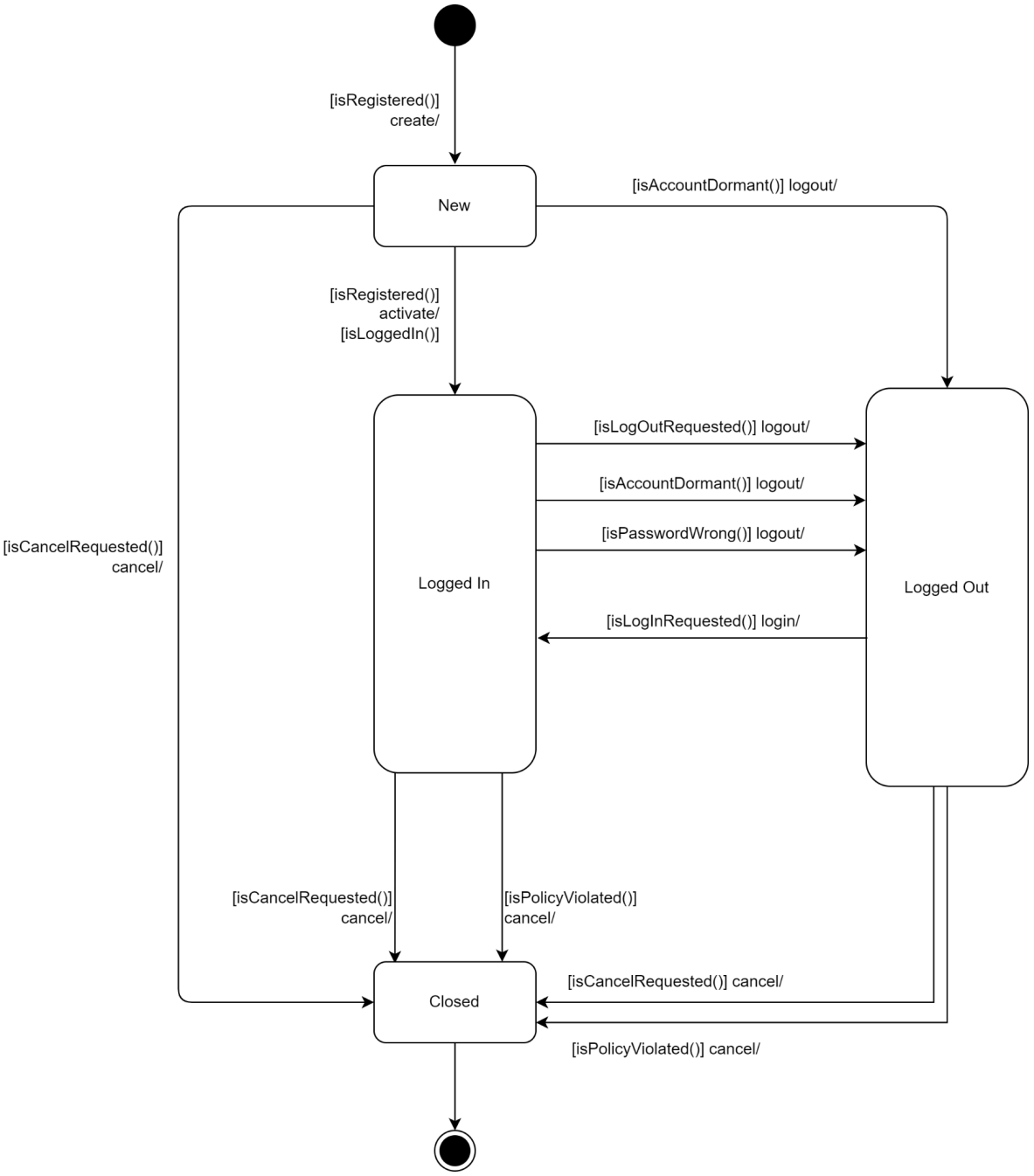


Figure 2.3: State Diagram for Register and Login Process of User for Serene System

2.3 Launch Phase

Sprint	Features	ID	User Story	Status	Assignee
Sprint 1	Requirement Analysis	1	-	Done	All
Sprint 2	Dashboard and Login	2	UC04 <Access Analytics Dashboard>	Done	Anjum
		3	UC01<Verify Professional Certification>	Done	Anjum
		4	UC 06 <Login To System>	Done	Fares
		5	UC 07 <Register to System>	Done	Sheyla
Sprint 3	Self assessment tools and forum	6	UC 09 < Access Self Assessment Tools >	Progress	Fares
		7	UC 11 <Access Peer Support Network >	Progress	Ivor
Sprint 4	Content page and Gamification	8	UC 08 <Access Interactive Mental Health Modules>	To Do	Ivor
		9	UC 10 < Promote Mental Health Awareness >	To Do	Sheyla
Sprint 5	Chatbot and Virtual Session	10	UC 02 <Provide Instant Feedback>	To Do	Fares
		11	UC 03 <Schedule Virtual Sessions>	Done	Fares
		12	UC 05 : <Interact with 24/7 Chatbot Support>	To Do	Anjum

Table 2.2 Launch Phase

2.4 User Story Details

2.4.1 **UC01: User Story <Verify Professional Certification>**

User Story ID	US01
User Story Name	Verify Professional Certification
User Story Description	*As a* Mental Health Professional I want to verify my professional certificate so that I can ensure the students that I am reliable and trustable and to be able to access the system.
Acceptance Criteria(s)	**Precondition**: The mental health professional should register as a user and login to the system to apply for verification. **Postcondition**: The system should send a notification upon successful verification of the mental health professional. **Other Conditions**: The mental health professional should hold a valid professional certificate
Normal Flow(s)- NF	1. The mental health professional registers and logs into the system. 2. The user navigates to the "Professional Certification Verification" section. 3. The user uploads their professional certificate for verification 3.1 If the user cancels the operation go to AF2 3.2 If an error occurs while uploading documents go to EF1 3.3 Upon successful upload show “Upload Successful”. 4. The system validates the document and forwards it to the authority for verification. 4.1 If the document to deemed invalid go to AF1

	5. Upon successful verification, the system updates the user's status to "Verification Successful" and sends a confirmation notification.
Alternative Flow(s) - AF	**AF1. Invalid Certificate** 1. The certificate is deemed invalid during verification. 2. The system updates the user's status to "Verification Failed" and notifies the user with the reason. **AF2. Request Cancellation** 1. The user cancels the verification request before submission. 2. The system discards the request without saving it.
Exception Flow(s) - EF	**EF1. Error occurs while uploading document** 1. Show “Error Uploading” message and ask the user to upload the document again

Table 2.3 User Story Description for <Verify Professional Certification>

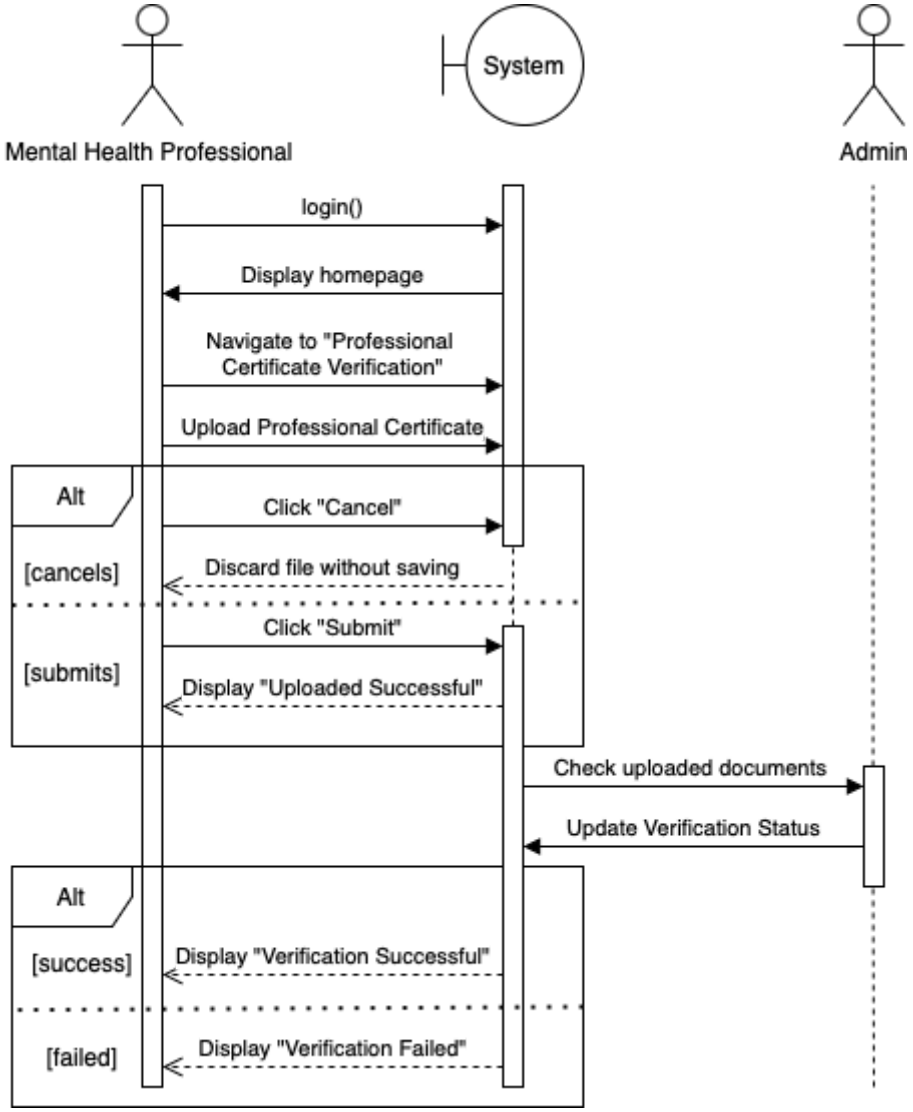


Figure 2.4 Sequence Diagram for <Verify Professional Certification>

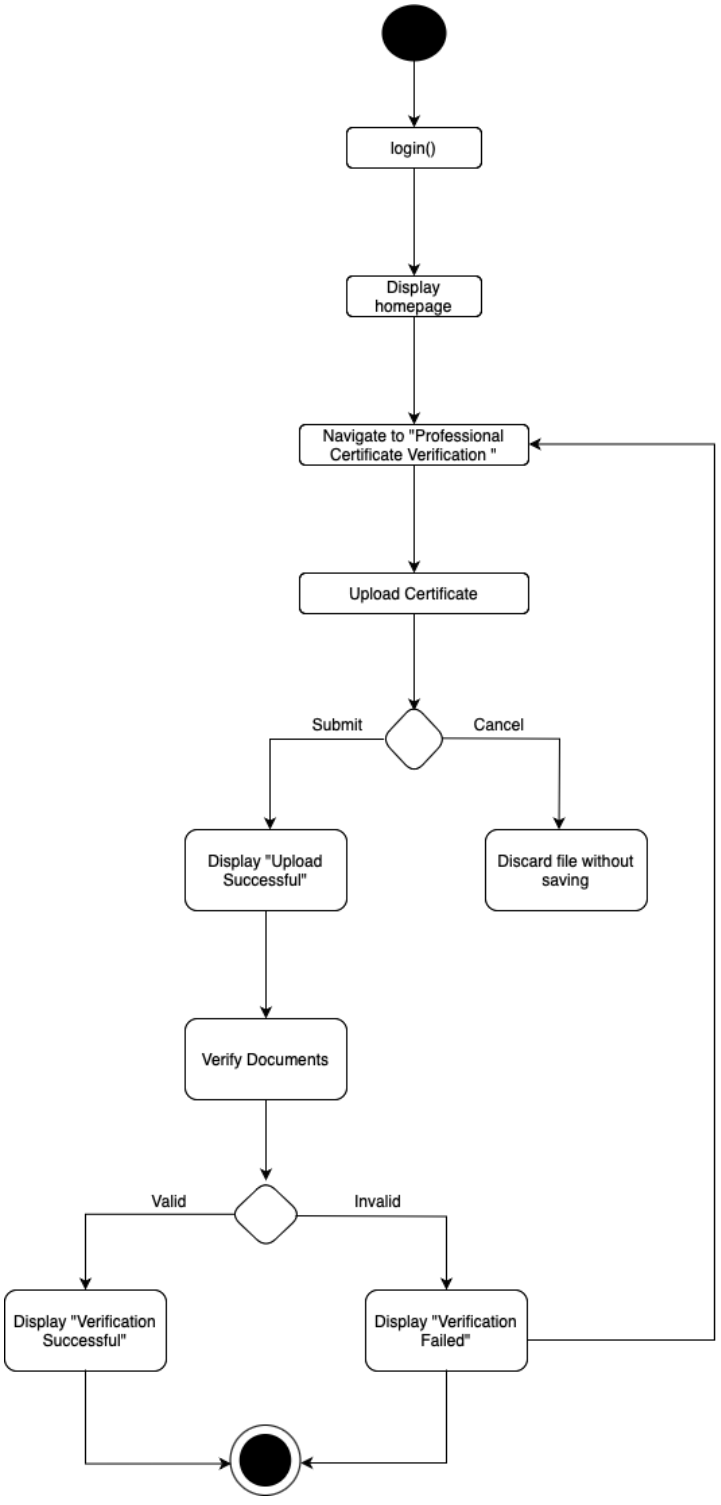


Figure 2.5 ActivityDiagram for <Verify Professional Certification>

2.4.2 **UC02: User Story <Provide Instant Feedback>**

User Story ID	US002
User Story Name	Providing Instant Feedback
User Story Description	As a **Mental Health Professional**, I want to provide instant feedback to students based on the outcomes of a completed virtual session so that I can guide them effectively in real time.
Acceptance Criteria(s)	1. The feedback should be stored in the system. 2. Students should be notified immediately. 3. Feedback can only be provided for completed sessions.
Normal Flow(s) (NF)	1. The Mental Health Professional logs in. 2. The professional navigates to the feedback section. 3. The professional fills out the feedback form. 4. The professional submits the form. 5. The system stores the feedback and displays a success message.
Alternative Flow(s) (AF)	- AF1: If the form submission fails due to connectivity issues, the user is prompted to retry once the issue is resolved..
Exception Flow(s) (EF)	- EF1: If the system is offline, save feedback locally and synchronize it with the server when the connection is restored.

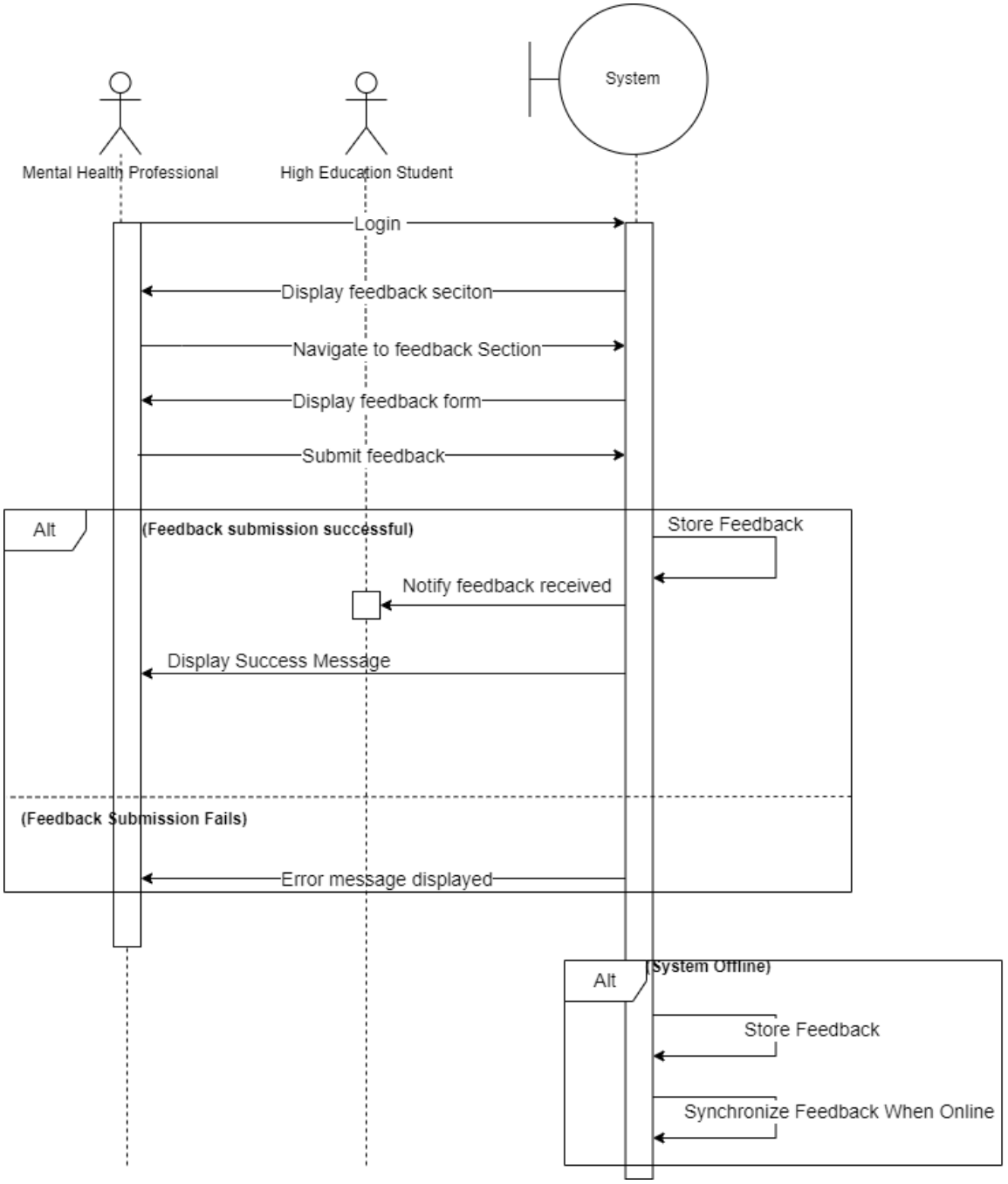


Figure 2.6 Sequence Diagram for <Provide Instant Feedback>

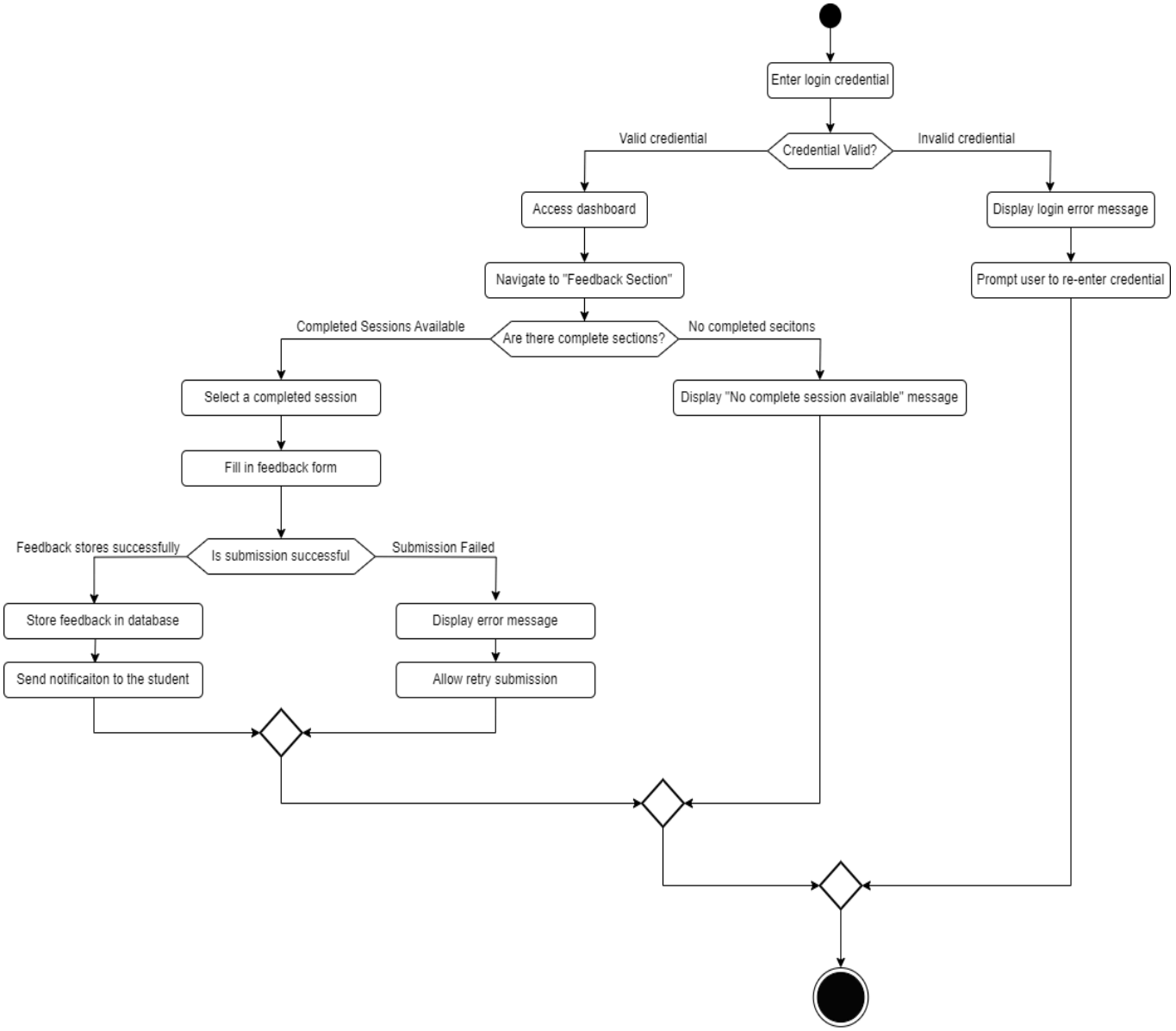


Figure 2.7 Active Diagram for <Provide Instant Feedback>

2.4.3 **UC03: User Story <Schedule Virtual Counselling Session>**

User Story ID	US03
User Story Name	Scheduling Virtual Counseling Session
User Story Description	As a **student**, I want to schedule a virtual counseling session so that I can receive professional help.
Acceptance Criteria(s)	**Postcondition**: The booking is confirmed, and reminders are sent. **Precondition**: The student is logged into the system. **Other Conditions**: Available slots for counselors exist.
Normal Flow(s)-NF	1. The student logs in. 2. The student selects “Virtual Counseling” from the menu. 3. The system displays available counselors and slots. 4. The student selects the counsellor and slot then clicks “Submit” 5. If the booking was successful display “Booking successful” message
Exception Flow(s) - EF	If the booking failed display “Booking not successful”

Table 2.4 User Story Description for <Schedule Virtual Counselling Session>

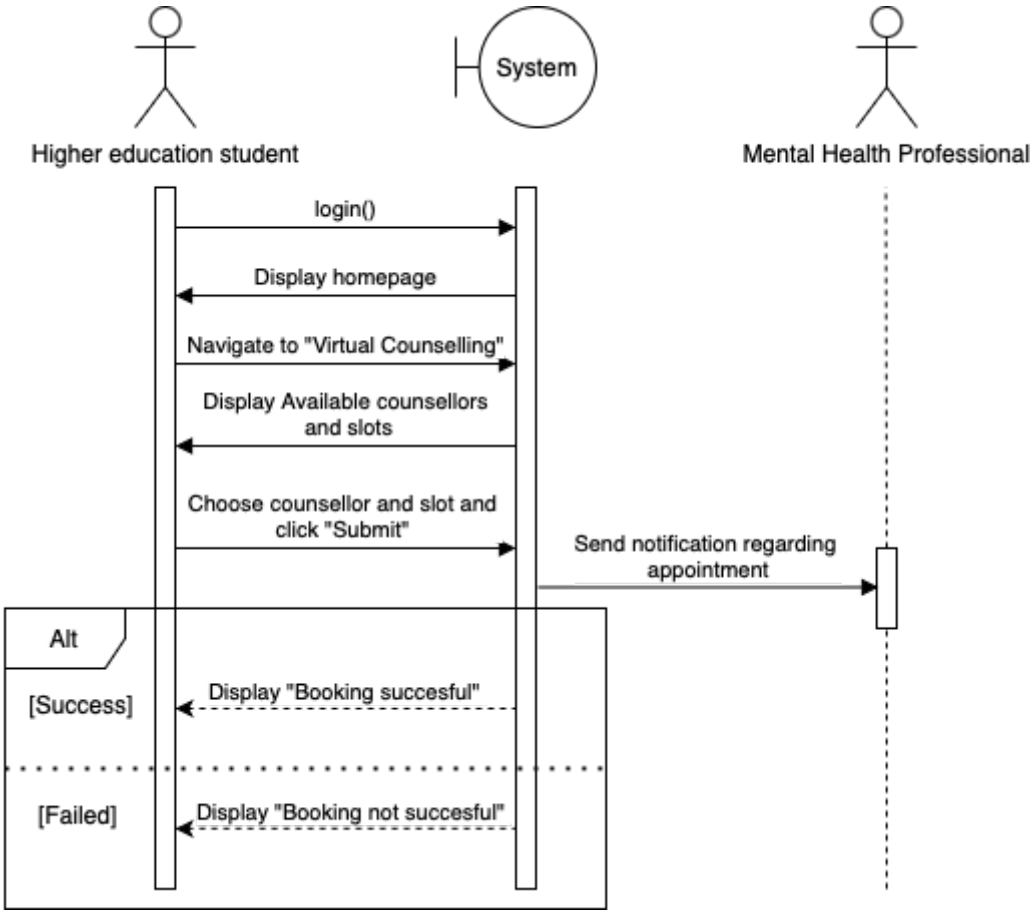


Figure 2.6 Sequence Diagram for <Schedule Virtual Counselling Session>

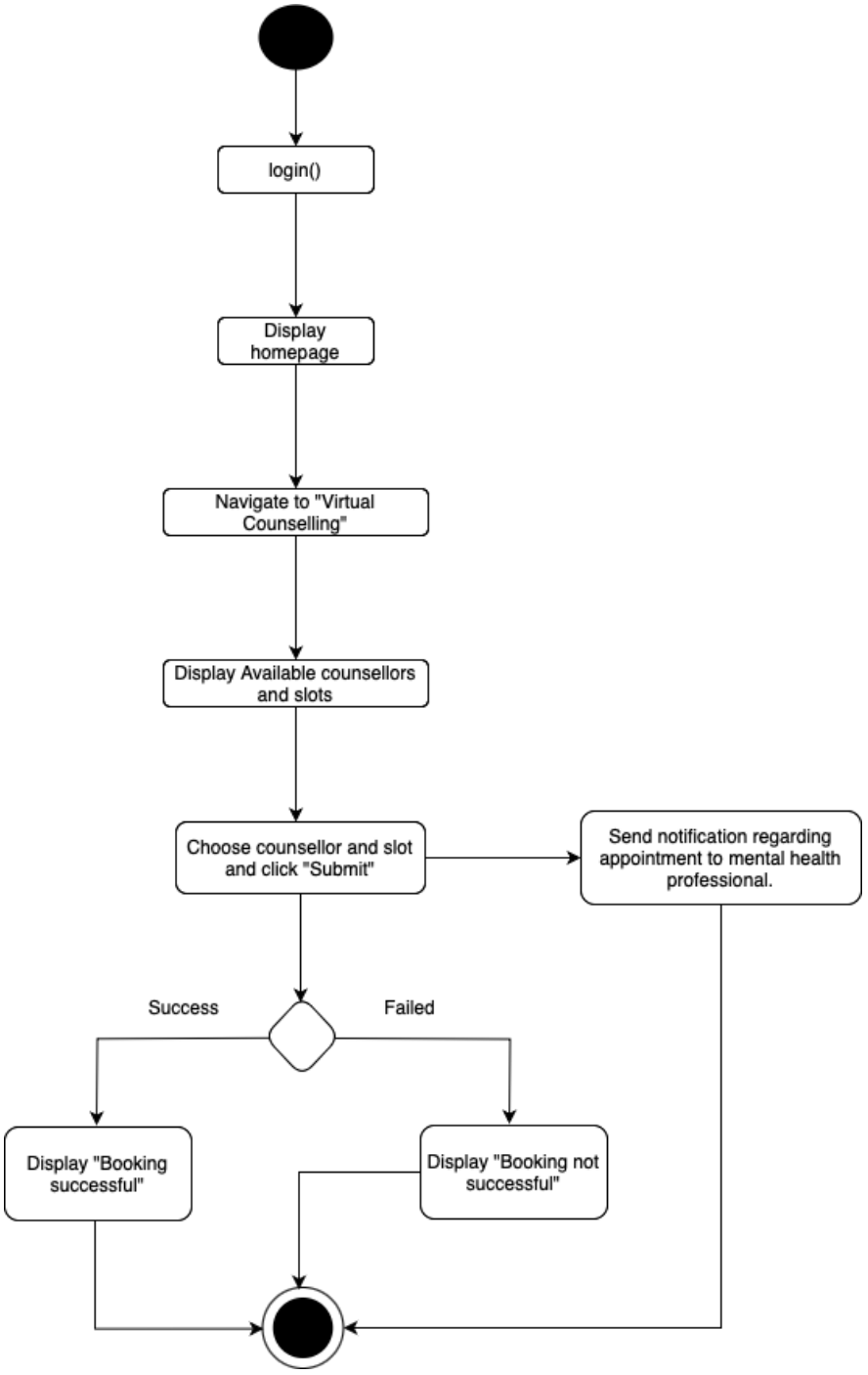


Figure 2.7 Activity Diagram for <Schedule Virtual Counselling Session>

2.4.4 **UC04: User Story <Access Analytics Dashboard>**

User Story ID	US04
User Story Name	Access Analytics Dashboard
User Story Description	*As an* Admin I want to access the Analytics dashboard so that I can help track overall student engagement, identify high-risk patterns.
Acceptance Criteria(s)	**Precondition**: The Admin should be logged to the system. **Postcondition**: The system should send a notification to concerned faculty members regarding high-risk students and also recommend methods to increase overall student engagement.
Normal Flow(s)- NF	1. The Admin logs into the system. 2. The Admin navigates to the "Analytics Dashboard" section. 3. The system displays an overview of the student engagement metrics and high-risk patterns. 4. The system sends a notification to the concerned faculty members about high-risk students and displays “Notification sent successfully”.
Exception Flow(s) - EF	If some error occurs while sending notification to faculty member show the “Notification not sent” message and ask the admin to retry again.

Table 2.4 User Story Description for <Access Analytics Dashboard>

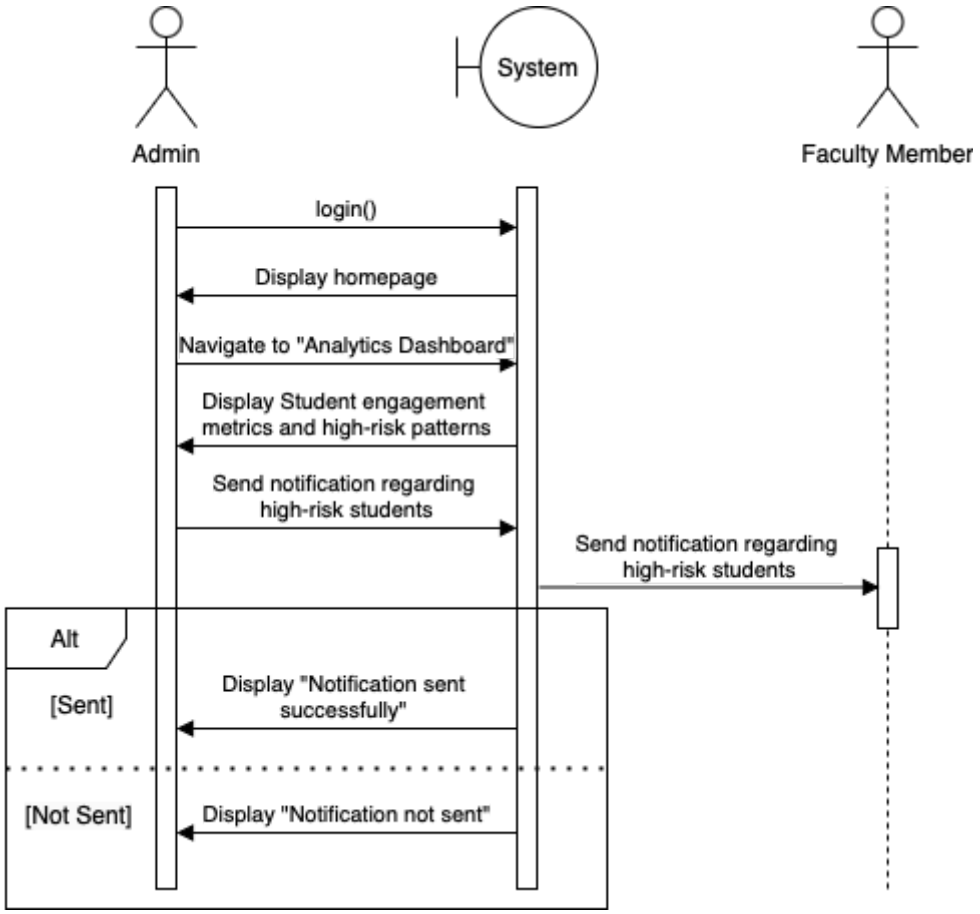


Figure 2.8 Sequence Diagram for <Access Analytics Dashboard>

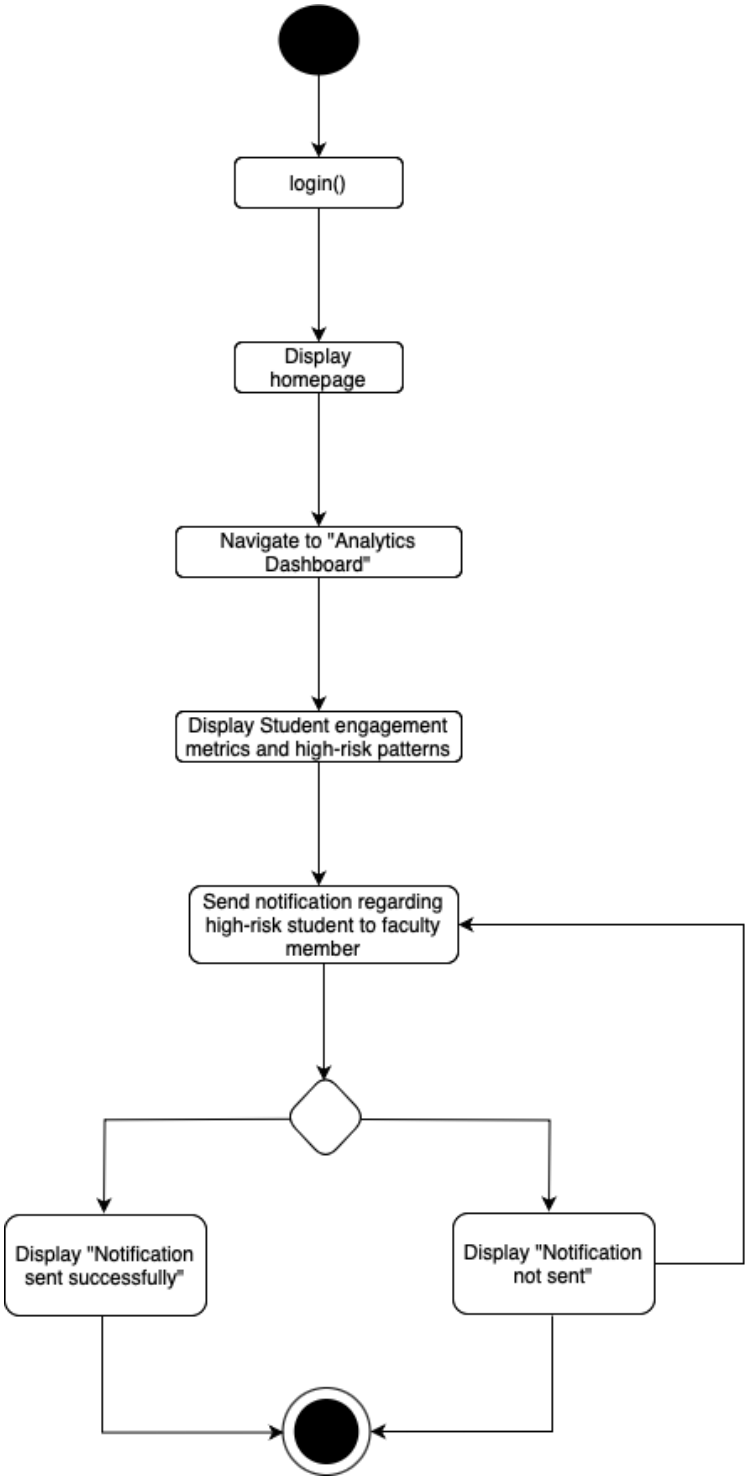


Figure 2.9 Activity Diagram for <Access Analytics Dashboard>

2.4.5 **UC05: User Story <Interact with 24/7 Chatbot Support>**

User Story ID	*US05*
User Story Name	*Interact with 24/7 Chatbot Support*
User Story Description	As a higher education student I want to interact with 24/7 chatbot support regarding my mental health so that I get precise replies as it is focused and developed mainly for mental health concerns.
Acceptance Criteria(s)	**Postcondition:** The system successfully analyzes and provides precise and relevant mental health based on student's input. **Precondition:** The students logged into the system **Other Condition:** The chatbot support will be able to store secure data from student's input and if the chatbot cannot handle a query, it must suggest a mental health professional.
Normal Flow(s)- NF	1. The student logs in. 2. The student navigates to the 'Chatbot Support' section. 3. The system displays the chatbot interface. 4. The student can type anything for input related to mental health. 5. The chatbot processes the input and generates a precise response. 6. The student reviews the response. 7. The chatbot provides follow-up recommendations. 8. The chatbot stores data to the system.
Alternative Flow(s) - AF	If the chatbot cannot handle the query, the chatbot will inform the students and suggest contacting mental health professionals along with providing the contact person or referral links.
Exception Flow(s) - EF	If the chatbot is error, The system will notify and display an option for reloading it.

Table 2.5 User Story Description for <Interact with 24/7 Chatbot Support>

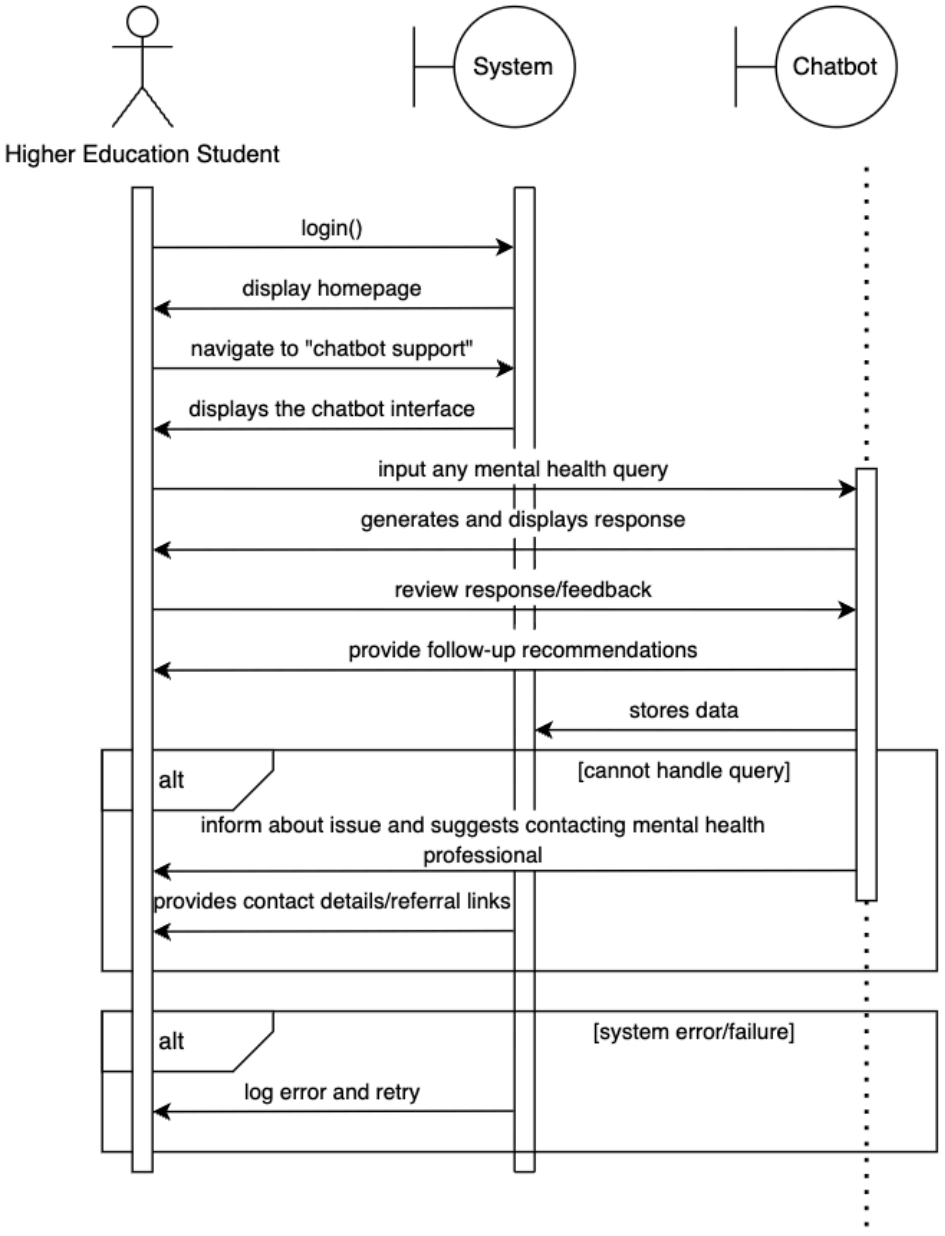


Figure 2.10 Sequence Diagram for <Interact with 24/7 Chatbot Support>

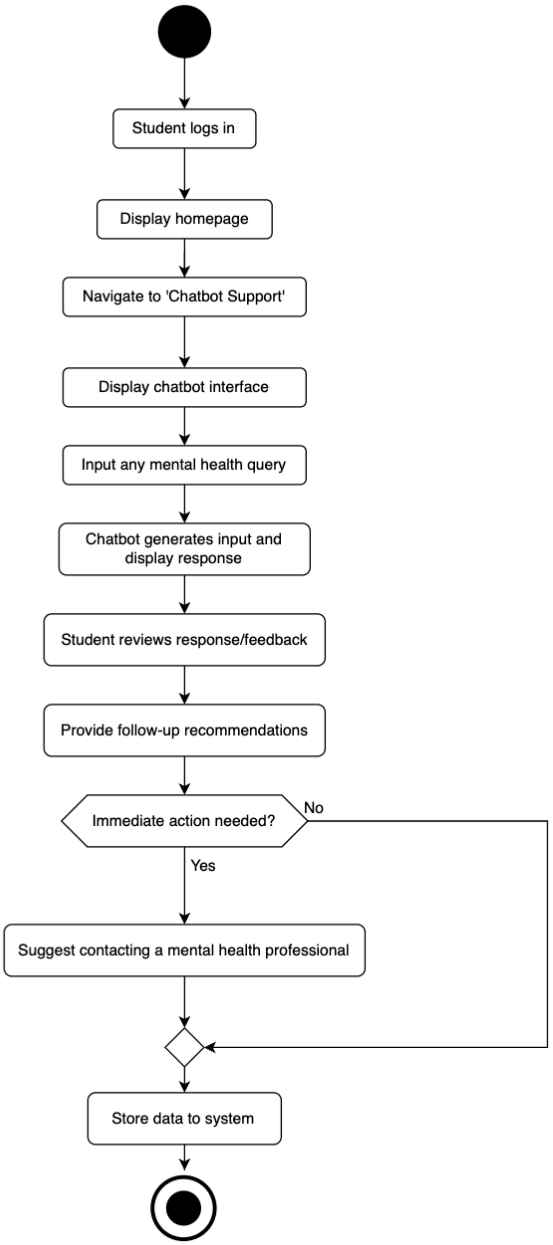


Figure 2.11 Activity Diagram for <Interact with 24/7 Chatbot Support>

2.4.6 **UC06: User Story <Login to the System>**

User Story ID	US006
User Story Name	Login to the System
User Story Description	As a user, I want to securely log in to the system so that I can access its features based on my role.
Acceptance Criteria(s)	- Postcondition: User is logged in and directed to their dashboard. - Precondition: The user has valid login credentials. - Other Conditions: Failed login attempts trigger appropriate error messages.
Normal Flow(s) (NF)	1. The user opens the system login page. 2. The user enters valid credentials (username and password). 3. The user submits the login form. 4. The system verifies the credentials. 5. The user is granted access to their role-specific dashboard.
Alternative Flow(s) (AF)	- AF1: If the user forgets their password, they can use the "Forgot Password" feature to reset it.
Exception Flow(s) (EF)	- EF1: If the system is offline, notify the user and prevent login.

Table 2.6 User Story Description for <Login to the System>

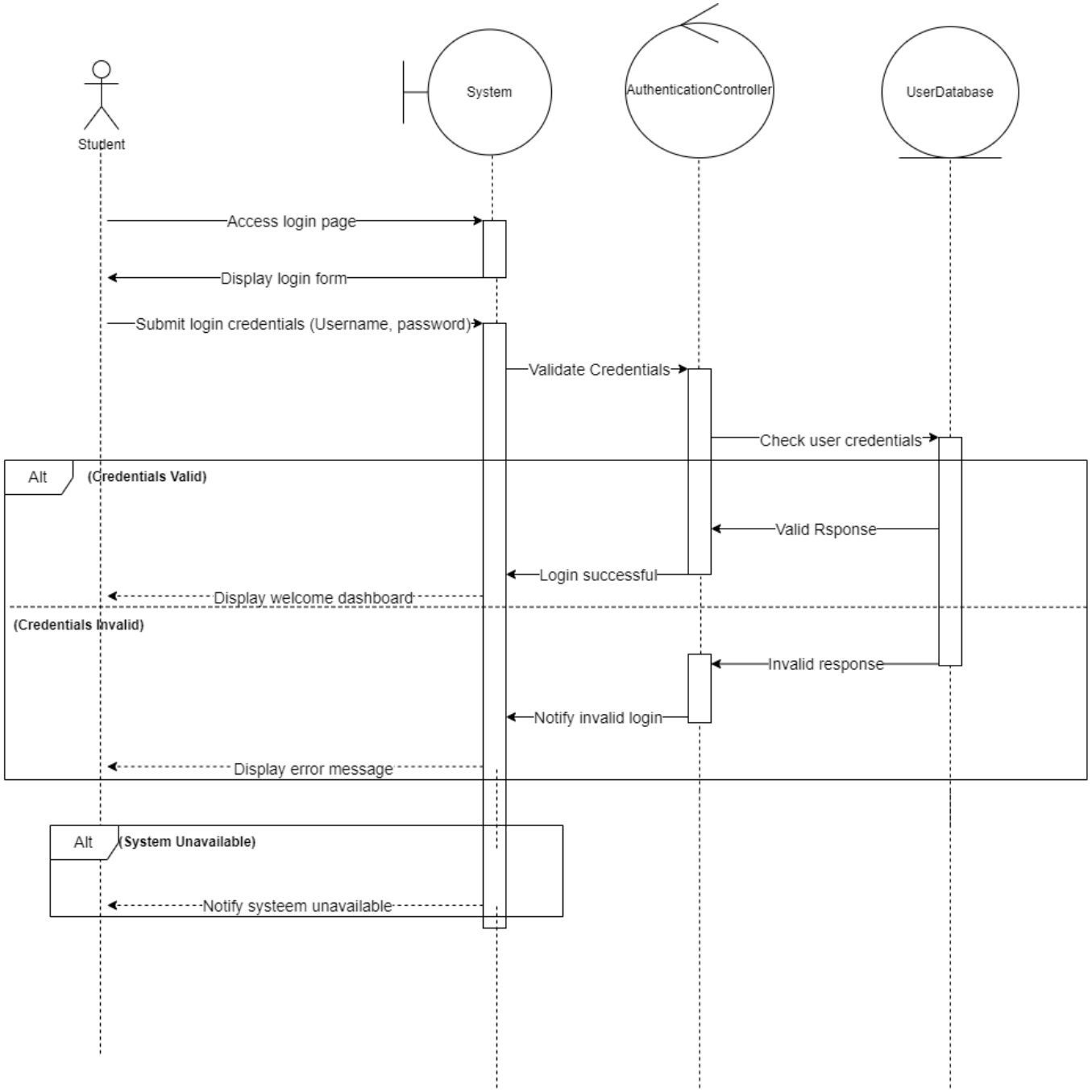


Figure 2.12 Sequence Diagram for <Login to the system>

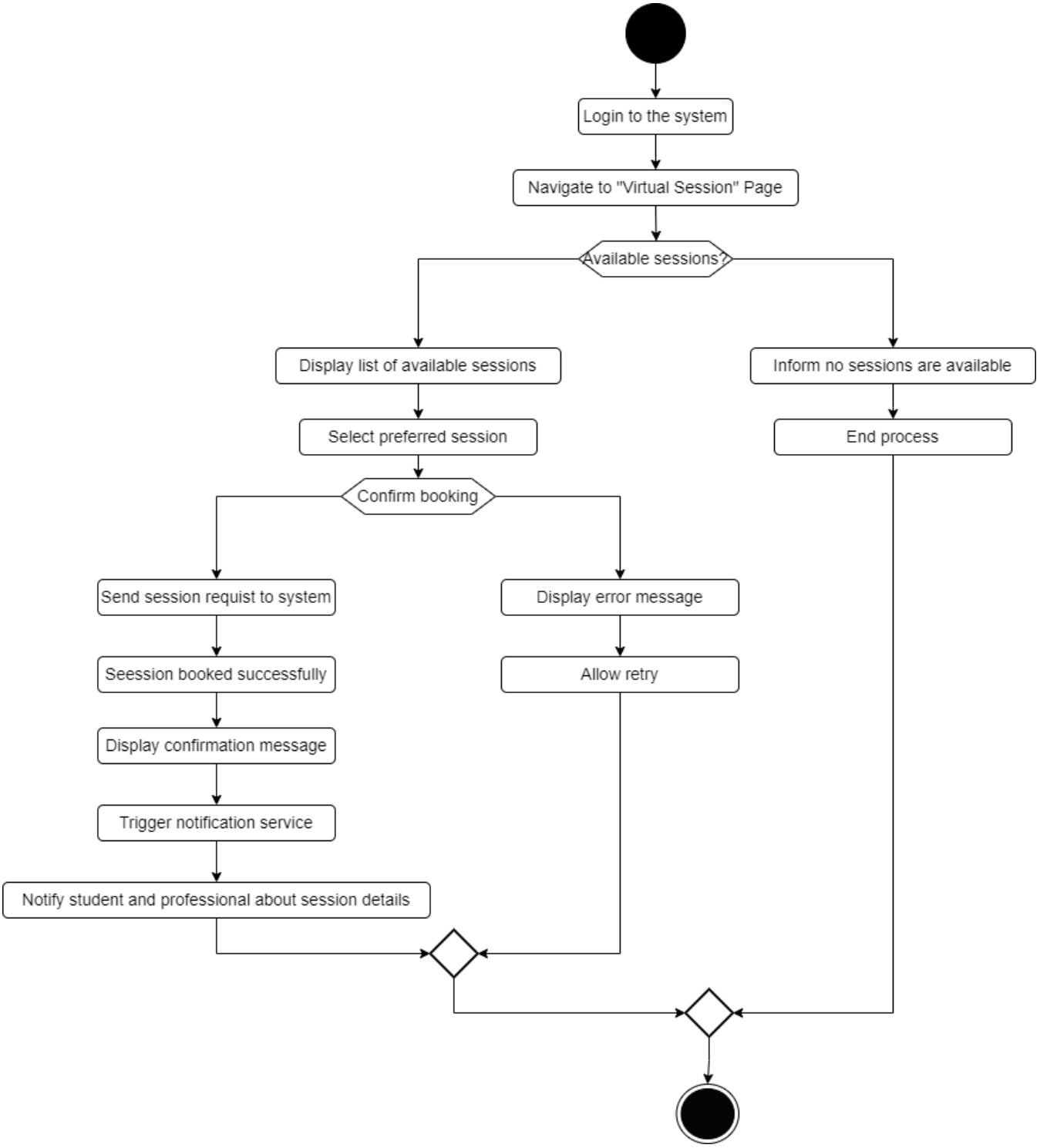


Figure 2.13 Active Diagram for <Login to the system>

2.4.7 **UC07: User Story <Register To The System>**

User Story ID	**US007**
User Story Name	Register to the System
User Story Description	As a **new user**, I want to register for the system so that I can access its features based on my role.
Acceptance Criteria(s)	- **Postcondition**: User is successfully registered and redirected to the login page. - **Precondition**: The user provides all required information. - **Other Conditions**: If registration fails, appropriate error messages are displayed.
Normal Flow(s) (NF)	1. The user navigates to the registration page. 2. The user fills out the registration form with valid information. 3. The user submits the form. 4. The system verifies the input and creates the user account. 5. The system displays a success message and redirects the user to the login page.

Alternative Flow(s) (AF)	- **AF1:** If the username or email is already registered, notify the user and allow them to choose a different one.
Exception Flow(s) (EF)	- **EF1:** If the system encounters a server issue, inform the user and retry the registration later.

Table 2.6 User Story Description for <Register to the System>

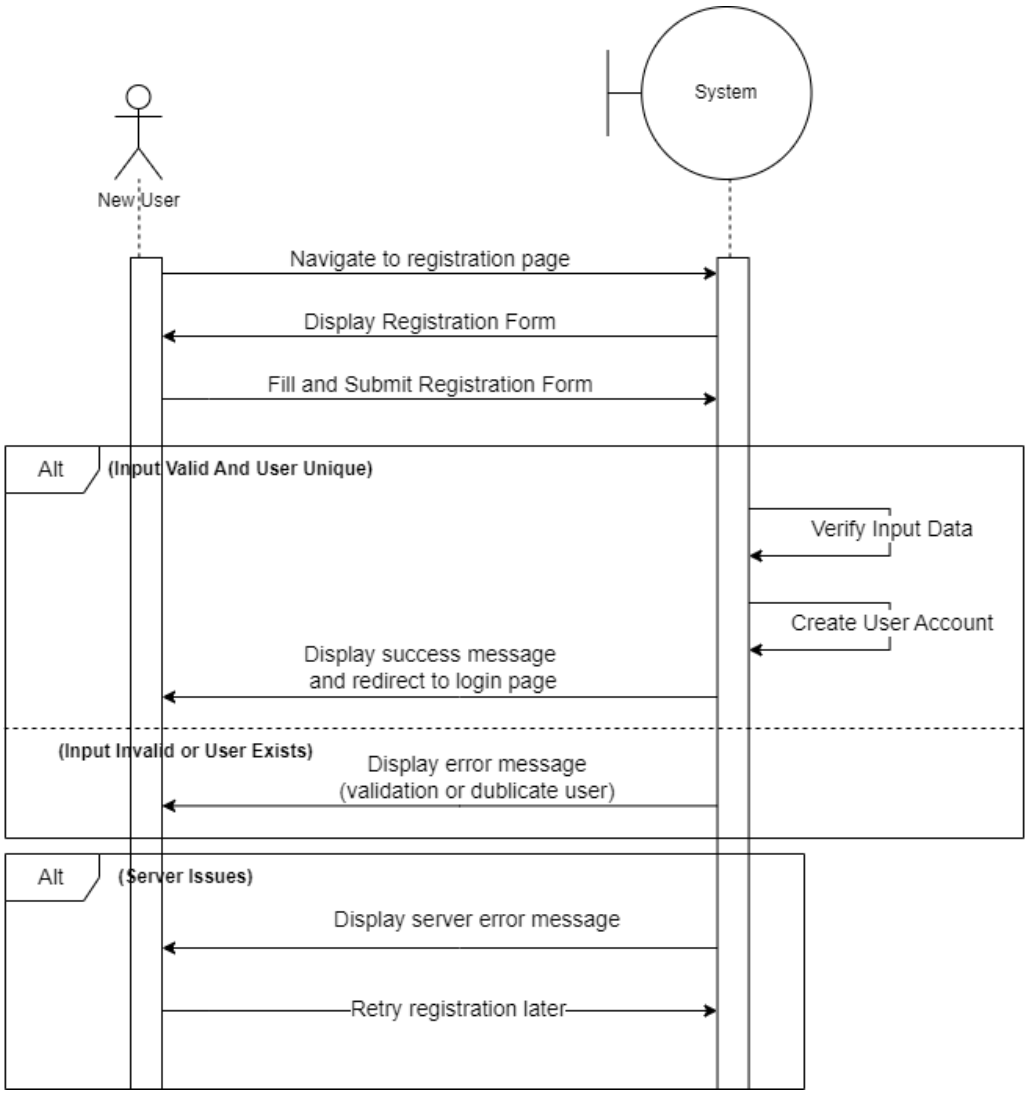


Figure 2.10 Sequence Diagram for <Register to the System>

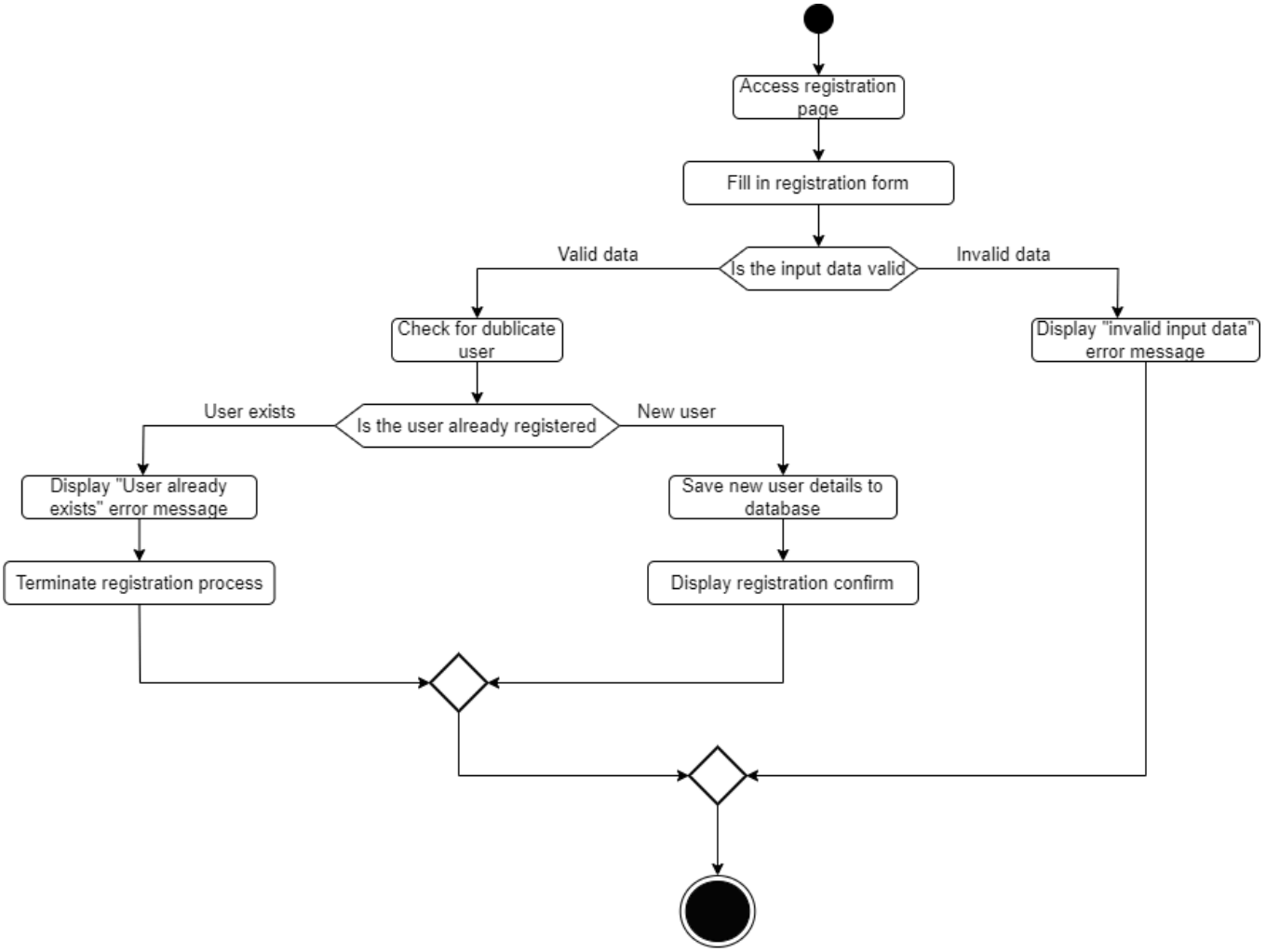


Figure 2.10 Active Diagram for <Register to the System>

2.4.8 **UC08: User Story <Access Interactive Mental Health Modules>**

User Story ID	US08
User Story Name	Study Interactive Mental Health Modules

User Story Description	As a student, I want to study Interactive Mental Health Modules so that I can get more information and learn about mental health especially when I am facing one myself.
Acceptance Criteria(s)	**Post Conditions:** Gain information about mental health problems and learn the steps needed to be done to overcome it. **Precondition:** The student is logged into the system **Other Conditions:** Suitable modules are posted by the faculty members.
Normal Flow(s)- NF	NF: 1. The student logs in. 2. The student selects "Interactive Mental Health Modules" from the menu. 3. The system displays all available modules. 4. If there are no modules available 4.1 The student can notify the faculty members about the lack of modules available on the system. 4.2 The use case terminates. 5. The student selects the module they want. 6. The system sends notification to students once they have finished the module.
Alternative Flow(s) - AF	Alternate Flow: At any point during the module, the student may leave the module and access it at the same progress they left it.
Exception Flow(s) - EF	Exception Flow When the system fails to display a module, log the error and retry.

Table 2.6 User Story Description for <Access Interactive Mental Health Modules>

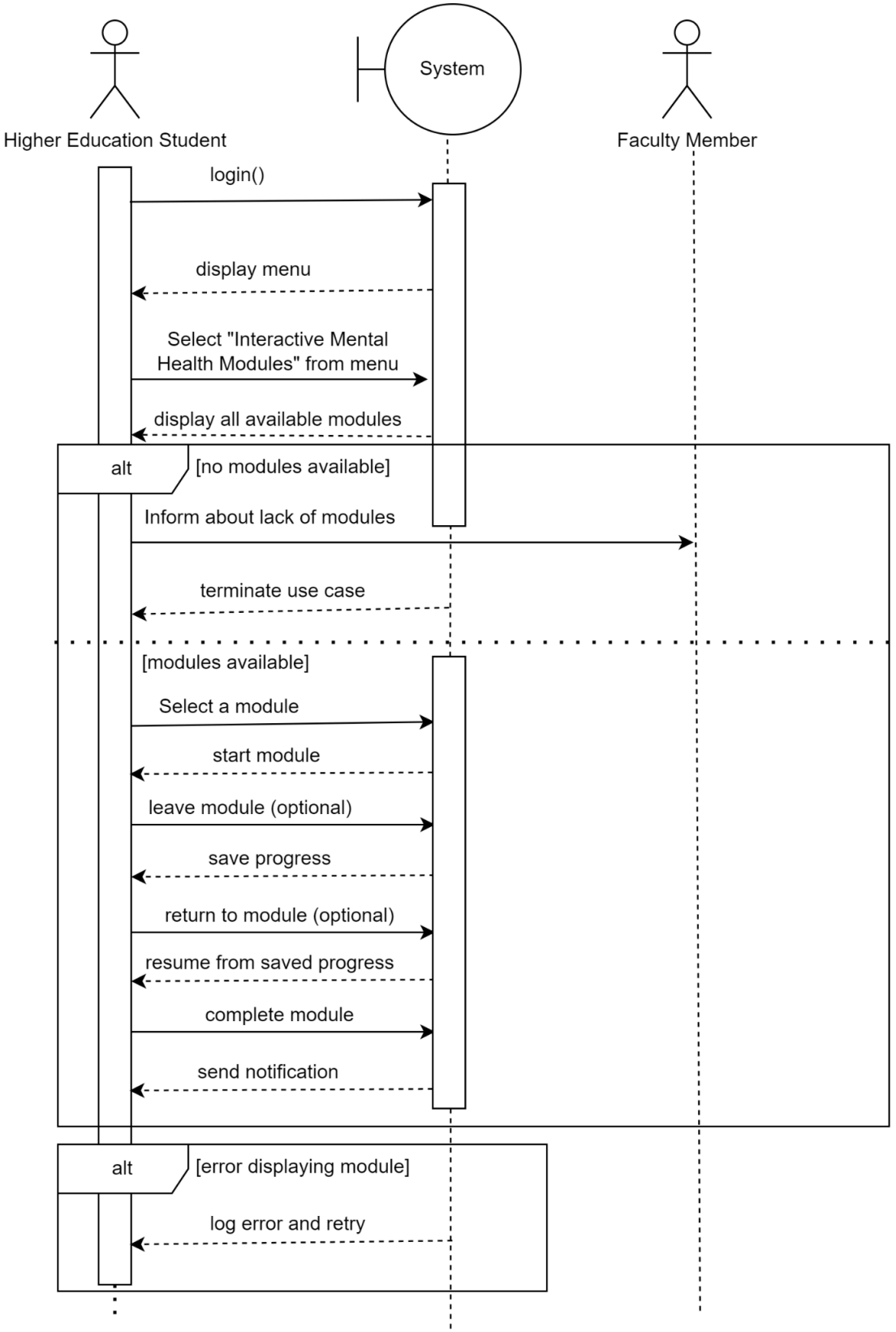

Higher Education Student
System
Faculty Member
login()
display menu
Select "Interactive Mental Health Modules" from menu
display all available modules
alt
[no modules available]
Inform about lack of modules
terminate use case
[modules available]
Select a module
start module
leave module (optional)
save progress
return to module (optional)
resume from saved progress
complete module
send notification
alt
[error displaying module]
log error and retry

Figure 2.12 Sequence Diagram for <Access Interactive Mental Health Modules>

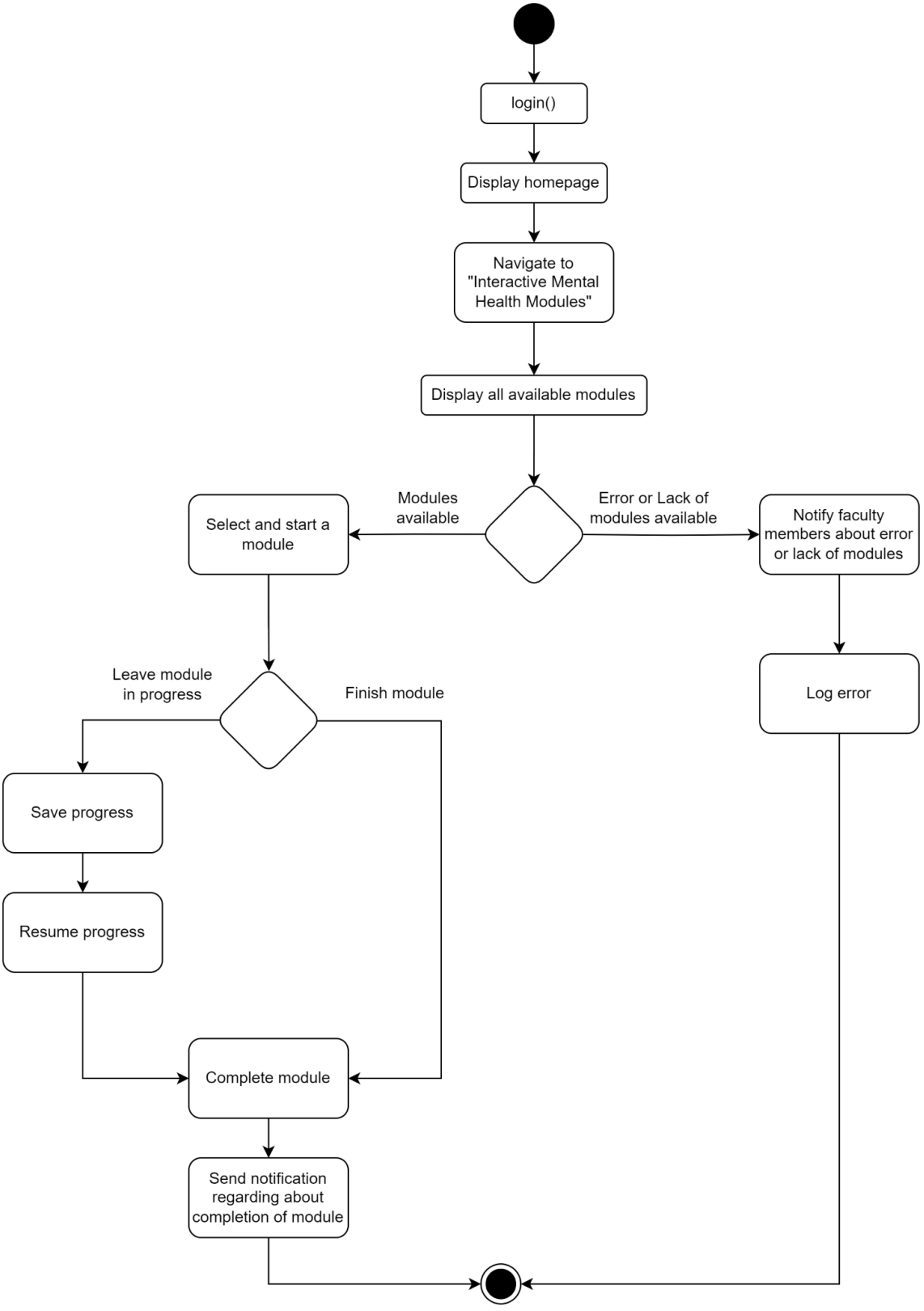

login()
Display homepage
Navigate to "Interactive Mental Health Modules"
Display all available modules
Modules available
Error or Lack of modules available
Select and start a module
Notify faculty members about error or lack of modules
Leave module in progress
Finish module
Log error
Save progress
Resume progress
Complete module
Send notification regarding about completion of module

Figure 2.13 Activity Diagram for <Access Interactive Mental Health Modules>

2.4.9 **UC09: User Story <Accessing Self-Assessment Tools>**

User Story ID	*US09*
User Story Name	*Accessing Self-Assessment Tools*
User Story Description	As a higher education student I want to access my mental health score so that I can identify any mental health issues that might affect me in a greater way if went unnoticed.
Acceptance Criteria(s)	**Postcondition**: Results and recommendations are displayed. **Precondition**: The student is logged into the system. **Other Conditions**: Self-assessment tools are available.
Normal Flow(s)-NF	1. The student logs in. 2. The student navigates to "Self-Assessment" 3. The system displays available assessments. 4. The student selects and completes an assessment. 5. The system provides results and recommendations.
Exception Flow(s) - EF	If there is a connection interruption during the assessment go back to the last saved assessment progress.

Table 2.6 User Story Description for <Accessing Self-Assessment Tools>

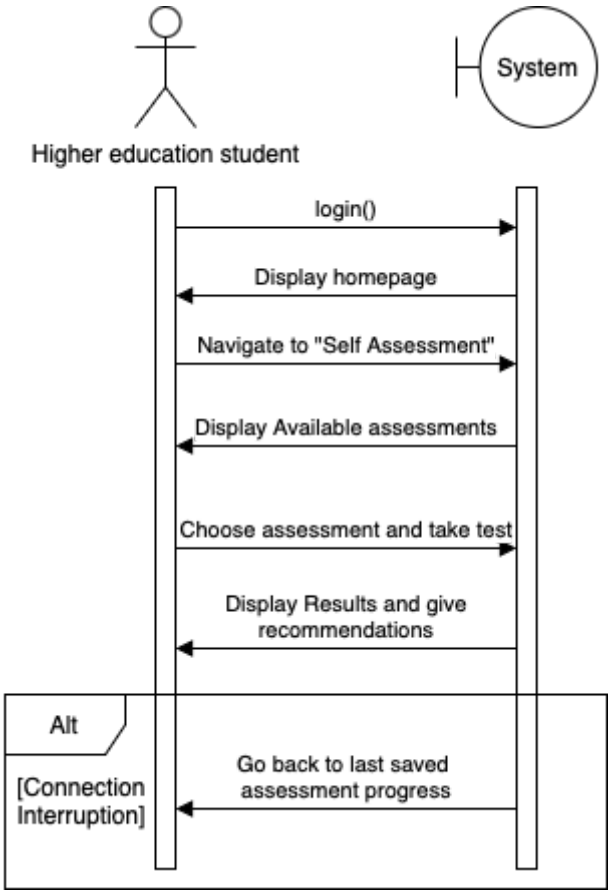


Figure 2.14 Sequence Diagram for <Accessing Self-Assessment Tools>

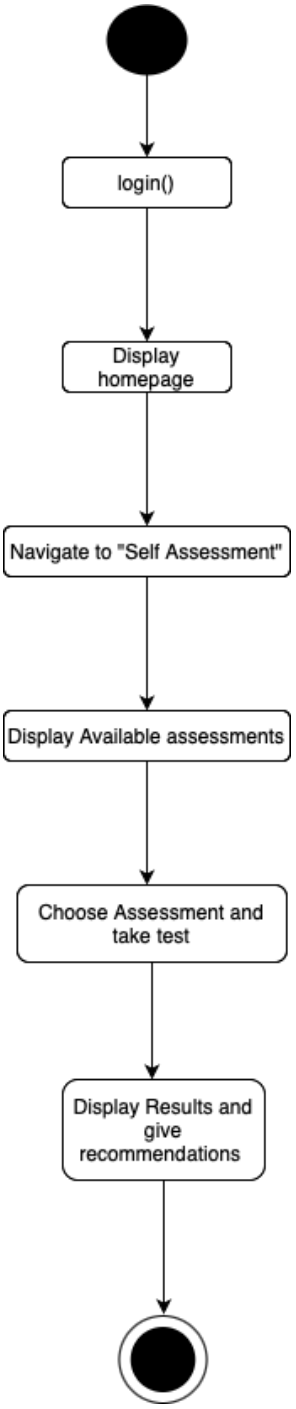


Figure 2.15 Activity Diagram for <Accessing Self-Assessment Tools>

2.4.10 **UC10: User Story <Promote Mental Health Awareness>**

User Story ID	*US10*
User Story Name	*Promote Mental Health Awareness*
User Story Description	As a Faculty member I want to promote mental health awareness among my students so that I can serve as a potential supporter and advocate for student well-being.
Acceptance Criteria(s)	**Postcondition:** Mental health awareness resources, events, or campaigns are successfully shared. **Precondition:** The student is logged into the system. **Other Condition:** Access to interactive and up-to-date materials are suitable for different target audiences.
Normal Flow(s)- NF	1. The students logs in 2. The students navigate to 'Promote Mental Health Awareness' tools. 3. The system displays options for promoting mental health awareness. 4. The students select options that are preferred. 5. The system processes their request by submitting it to Faculty Members for approval. 6. Faculty members approve the request. 7. The system notifies the student of a successful request.
Alternative Flow(s) - AF	The system allows students to upload materials for any promotion.
Exception Flow(s) - EF	If the system fails to publish the content, the system will notify the students about the error and provide an option to retry.

Table 2.7 User Story Description for <Promoting Mental Health Awareness >

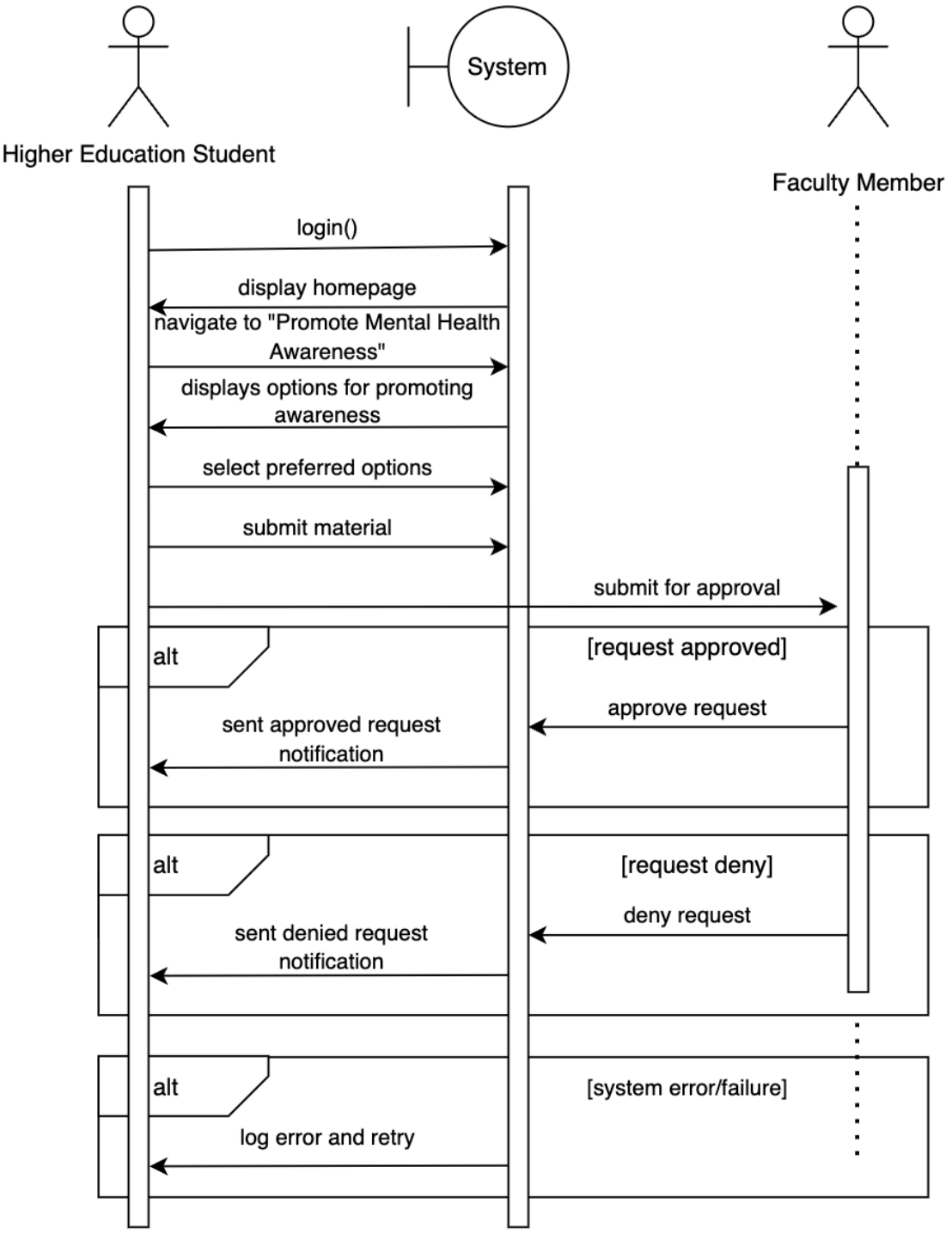


Figure 2.16 Sequence Diagram for <Promoting Mental Health Awareness>

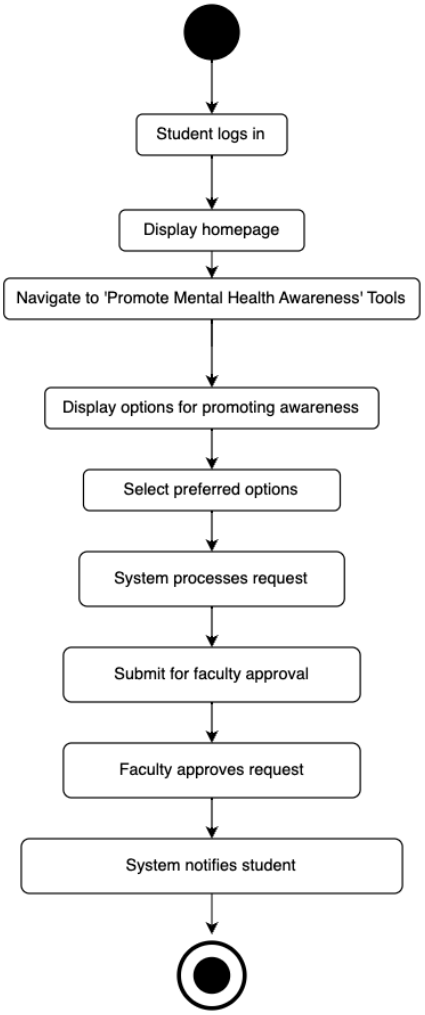


Figure 2.17 Activity Diagram for <Promoting Mental Health Awareness>

2.5 Performance and Other Requirements

Performance Requirements

- **Response Time**: When a user logs in, navigates, or submits, the system should react in less than 2 seconds.
- **Concurrent Users**: It is going to support up to **500 concurrent users** without performance decrease.

- **Availability**: The system will make sure there is 99.9% work time, to not interrupt user access.
- **Data Processing**: The system will generate the time of analytics queries and reports, which shouldn't be more than three seconds.

Usability Requirements

- **Accessibility**: The platform must comply with **WCAG 2.1** standards to ensure accessibility for users with disabilities.
- **User Interface**: For administrators, professionals, and students same, the interface should be simple to use and intuitive.
- **Cross-Platform Compatibility**: Ensure functionality across devices (desktop, tablet, mobile) and modern web browsers.

Reliability Requirements

- **Data Consistency**: Ensure all input data (assessments, bookings) are reliably stored and retrieved.
- **Fault Tolerance**: The system should recover from crashes or failures within 5 minutes without data loss.
- **Backup Mechanisms**: Daily automated backups should be performed to prevent data loss.

Security Requirements

- **Data Encryption**: All sensitive data must be encrypted during storage and transmission using **AES-256 encryption**.
- **Authentication**: Implement **two-factor authentication** for system access.
- **Access Control**: To guarantee that users only access authorized functionalities, use role-based access control, or RBAC.

Scalability Requirements

- **Future Growth**: Expand to support up to 1,000 users at once.
- **Modular Design**:New features should be able to be added to the architecture without requiring a significant redesign.

Environmental Requirements

- **Operating Environment**: The system should run on a **cloud-based infrastructure**, supporting all modern operating systems (Windows, Linux, macOS).
- For upcoming expansions, facilitate integration with third-party APIs (external mental health resources).

Hardware

- **The rams and storage:** Minimum server requirements: 4-core CPU, 16GB RAM, 500GB SSD for fast and quick storage.

Maintainability and Support

- **Code Standards**: Make sure all inline documentation is thorough and adhere to clean coding standards.
- **Bug Fixes and Updates:**Automated tools should help with monitoring and error handling, and make sure patches and updates can be applied easily without causing system congestion.

2.6 Design Constraints

Environmental Constraints

- The system must operate in a cloud-based environment with servers designed for 15°C to 32°C and 50%-60% humidity.

Hardware Constraints

- The system must run on servers with at least 4-core processors, 16GB RAM, and 500GB SSD storage.
- Devices used by end users must have at least 4GB of RAM and be compatible with a current web browser.

Security Constraints

- AES-256 encryption is required for sensitive data.
- The system must comply with PDPA and GDPR regulations.
- Administrative access requires two-factor authentication (2FA).

Compatibility Constraints

- The system must support Chrome (v90+), Firefox (v85+), Safari (v14+), and Edge (v90+) across Windows, macOS, Android and iOS.
- It needs to integrate with third-party tools like Google Calendar and Zoom.
    - Performance Constraints
- Support up to **500 concurrent users**, scalable to 1,000 in the future.
- Make sure that user actions respond in two seconds and report generation takes five seconds.

3 System Architectural Design

3.1 Architecture Style and Rationale

The MVC architecture is really good for the application we have in mind, Serene System, which provides functionalities like accessing mental health resources, managing user interactions, and setting up virtual sessions. The management will be made by the model related data or operations, like testing results, user profiles, and session information. Administrators, mental health professionals, and students all will have an easy and simple experience, to the View component's management of the user interface.

User inputs will be through interpretation, also the updates of the model, and execution of the pertinent view, the controller will serve as a mediator. Among its many important advantages is the ease with which MVC enables modular development. to preserve the system's flexibility and upkeep. For example, modifications to the user interface (View) can be created without impacting the fundamental (Model) logic .

Splitting apart concerns also makes the process of debugging easier while enabling multiple team members to work on a lot of components at the exact time. MVC also increases the system's scalability, which makes it easier to accompany the new features like advanced analytics or AI-driven recommendations without affecting functionality that already exists. The architecture also facilitates flexible deployment on web and mobile platforms by keeping a clear separation between the components.To sum up, the MVC architecture provides a solid basis for the Serene System's development, ensuring flexibility, scalability, and maintainability while meeting the project's functional and non-functional goals

4 Detailed Description of Component

4.1 Complete Package Diagram

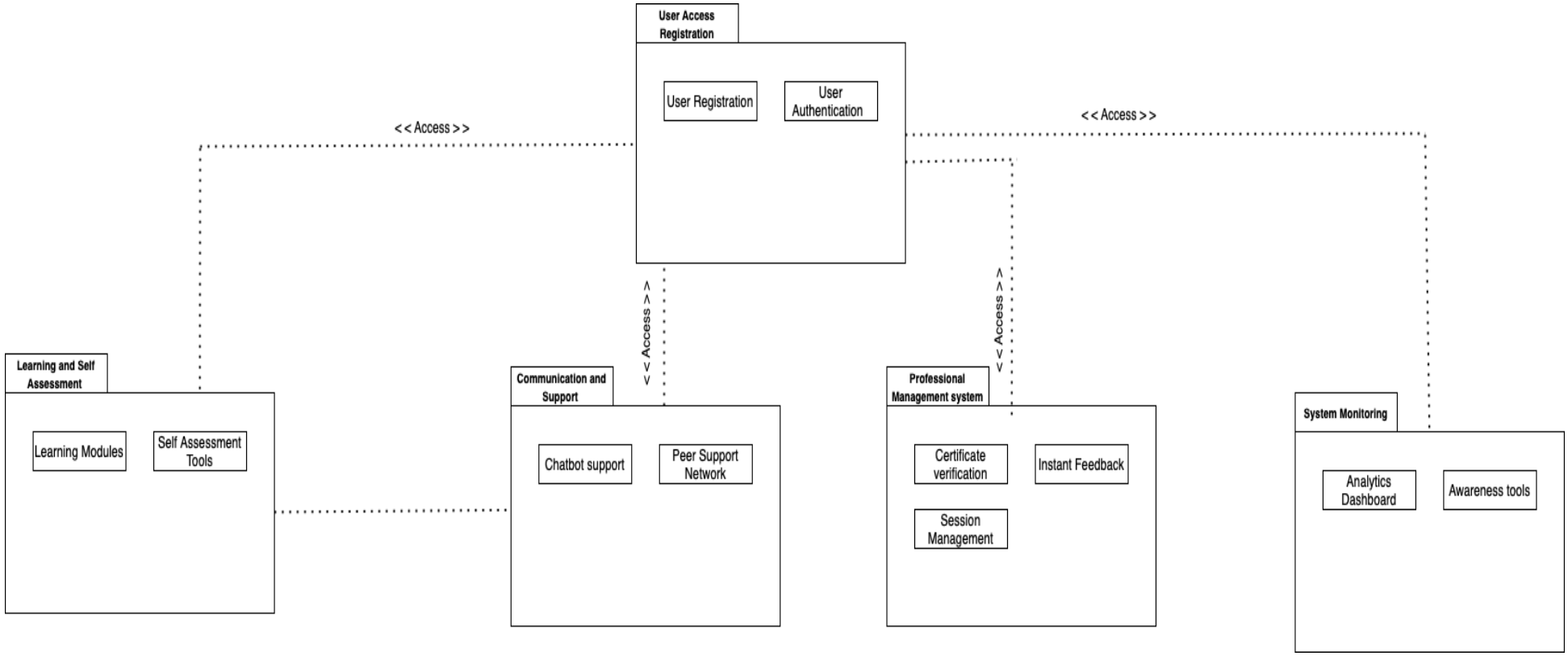


Figure 4.1: Package Diagram for <Name of the System>* Refer to appendix

4.2 Detailed Description

4.2.1 P001 <User Access Registration>Subsystem

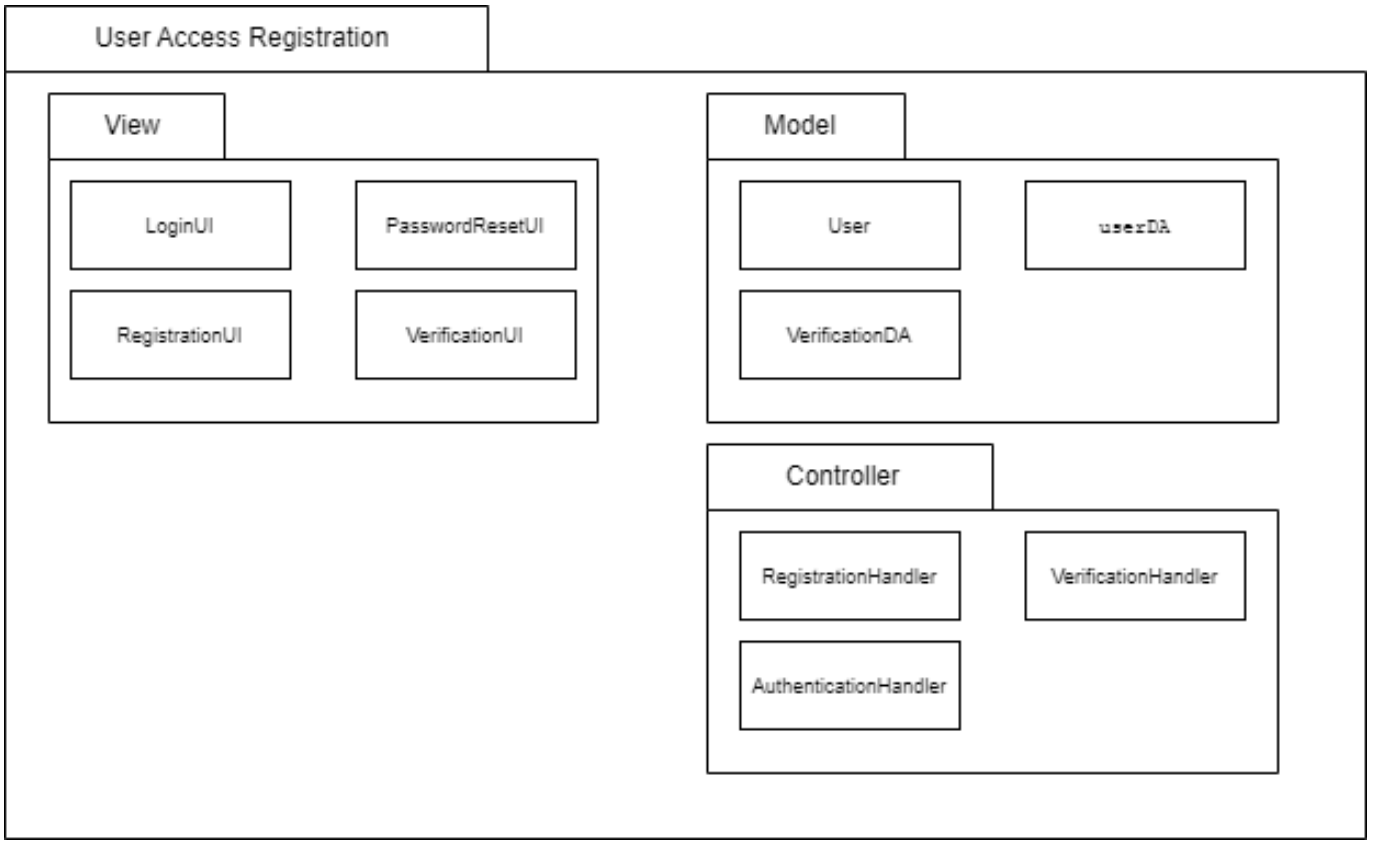


Figure 4.2 Package Diagram for<User Access Registration>Subsystem

4.2.1.1 Class Diagram

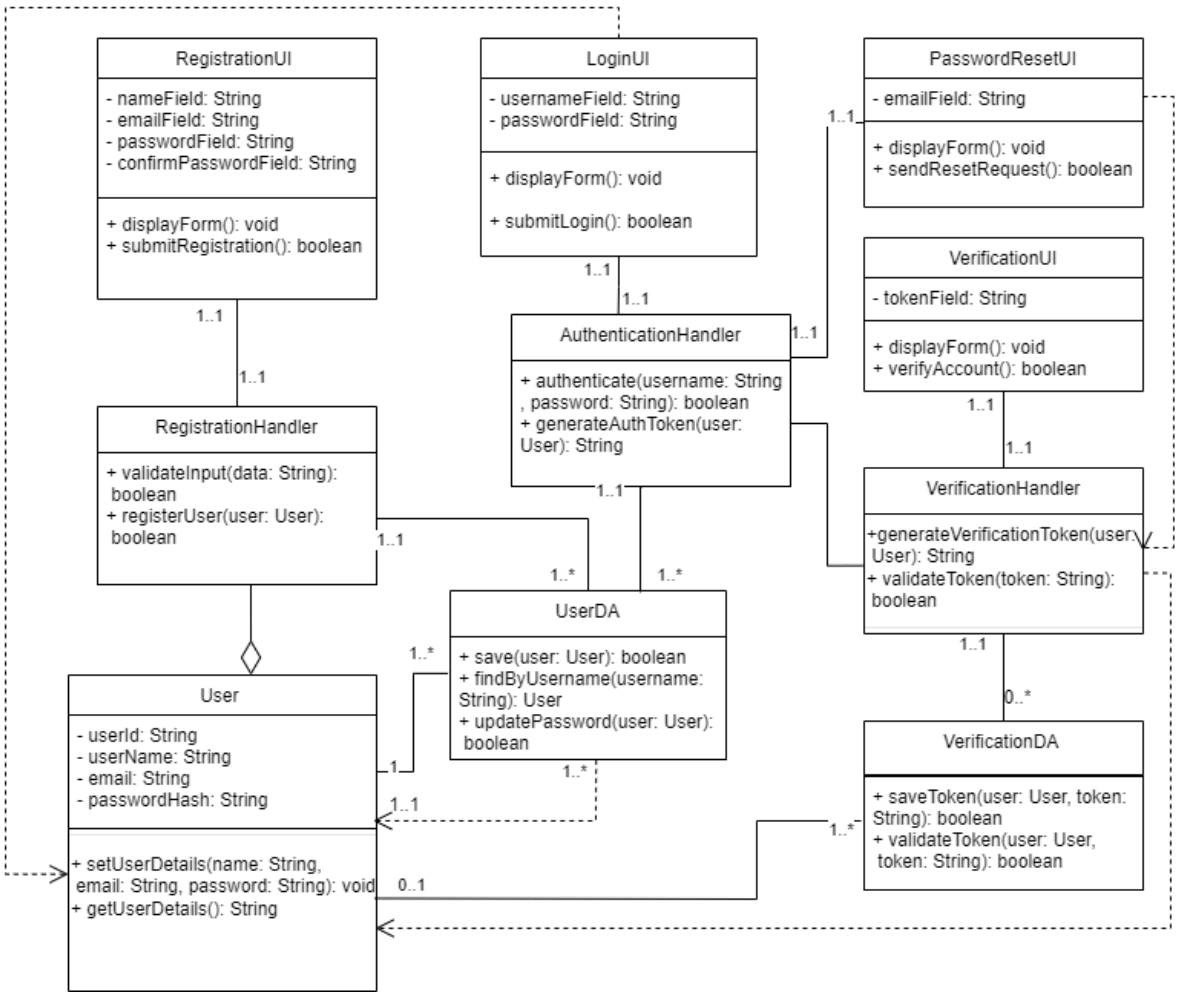


Figure 4.3: Class Diagram for<User Access Registration> Subsystem

Entity Name	LoginUI
Method Name	displayForm(): void
Input	None
Output	None
Algorithm	1. Start. 2. Render the login form on the screen. 3. Initialize input fields for username and password. 4. End.

Entity Name	LoginUI
Method Name	submitLogin(username: String, password: String): boolean
Input	username, password
Output	true if login is successful, false otherwise
Algorithm	1. Start. 2. Accept input values for username and password. 3. Validate inputs against system rules. 4. Return success or failure. 5. End.

Entity Name	RegistrationUI
Method Name	displayForm(): void
Input	None
Output	None
Algorithm	1. Start. 2. Render the registration form. 3. Initialize fields for name, email, password, and confirm password. 4. End.

Entity Name	RegistrationUI
Method Name	submitRegistration(name: String, email: String, password: String, confirmPassword: String): boolean
Input	name, email, password, confirmPassword
Output	true if registration is successful, false otherwise
Algorithm	1. Start. 2. Accept values for all fields. 3. Check that password matches confirmPassword. 4. Validate fields according to rules. 5. Submit data for storage. 6. Return success or failure. 7. End.

Entity Name	PasswordResetUI
Method Name	displayForm(): void
Input	None
Output	None
Algorithm	1. Start. 2. Render the password reset form. 3. Provide a field to input email. 4. End.

Entity Name	PasswordResetUI
Method Name	sendResetRequest(email: String): boolean
Input	email
Output	true if reset email sent successfully, false otherwise
Algorithm	1. Start. 2. Accept the email address. 3. Validate the email exists in the system. 4. If valid, generate a reset link and send it. 5. Return success or failure. 6. End.

Entity Name	VerificationUI
Method Name	displayForm(): void
Input	None
Output	None
Algorithm	1. Start. 2. Render the verification form. 3. Provide input fields for the verification token. 4. End.

Entity Name	VerificationUI
Method Name	verifyAccount(token: String): boolean
Input	token
Output	true if verification is successful, false otherwise
Algorithm	1. Start. 2. Accept the token as input. 3. Validate the token against stored system tokens. 4. If valid, mark the account as verified. 5. Return success or failure. 6. End.

Entity Name	User
Method Name	setUserDetails(name: String, email: String, password: String): void
Input	name, email, password
Output	None
Algorithm	1. Start. 2. Assign values to the User attributes: name, email, password. 3. End.

Entity Name	User
Method Name	getUserDetails(): String
Input	None
Output	Returns user details as a String
Algorithm	1. Start. 2. Concatenate name, email, and password into a single string. 3. Return the concatenated string. 4. End.

4.2.2 Sequence Diagrams

a) SD001: Sequence Diagram for Registration Process

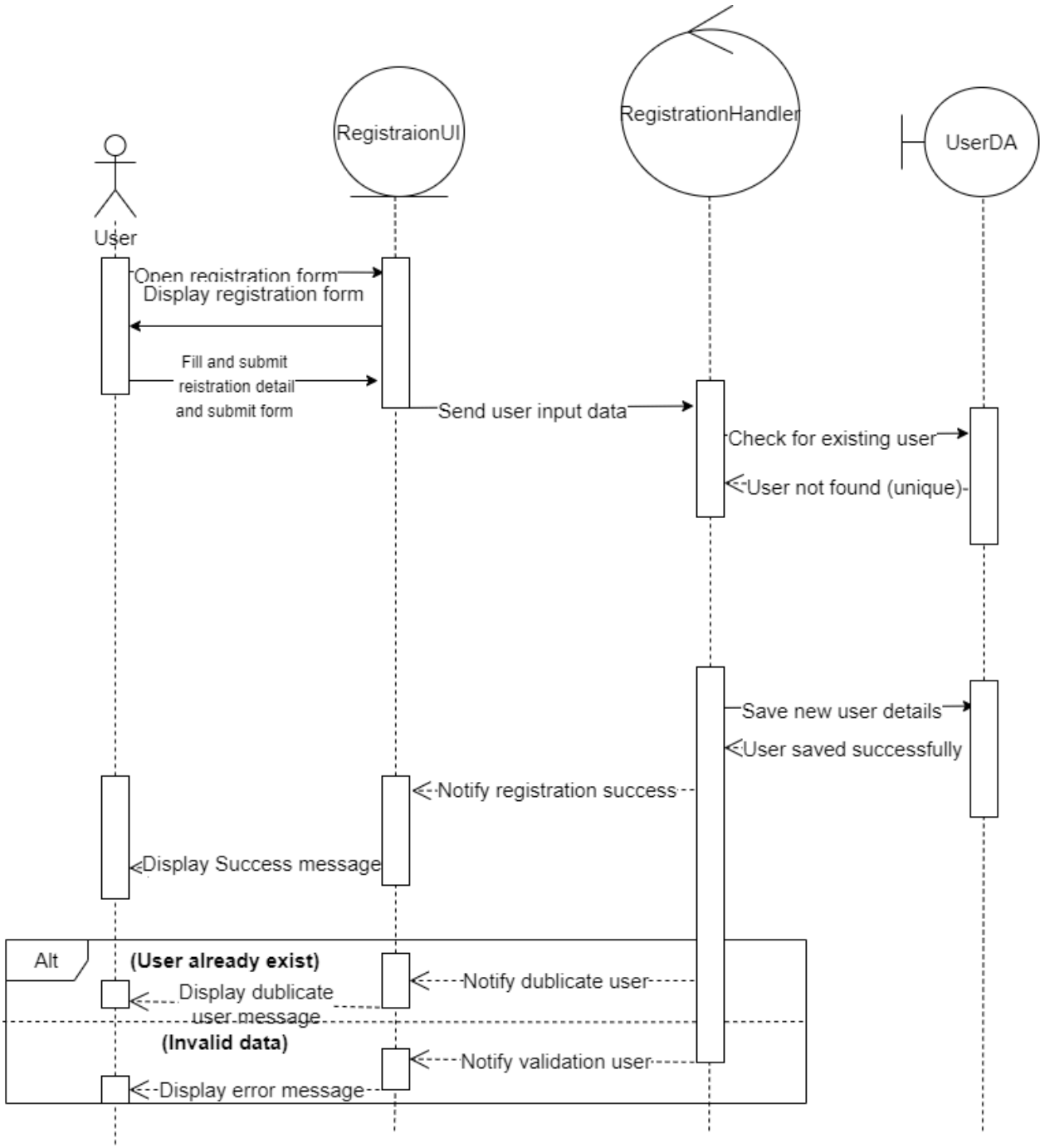


Figure 4.4 Sequence Diagram for <Registration Process>

b) SD002: Sequence Diagram for Verification process

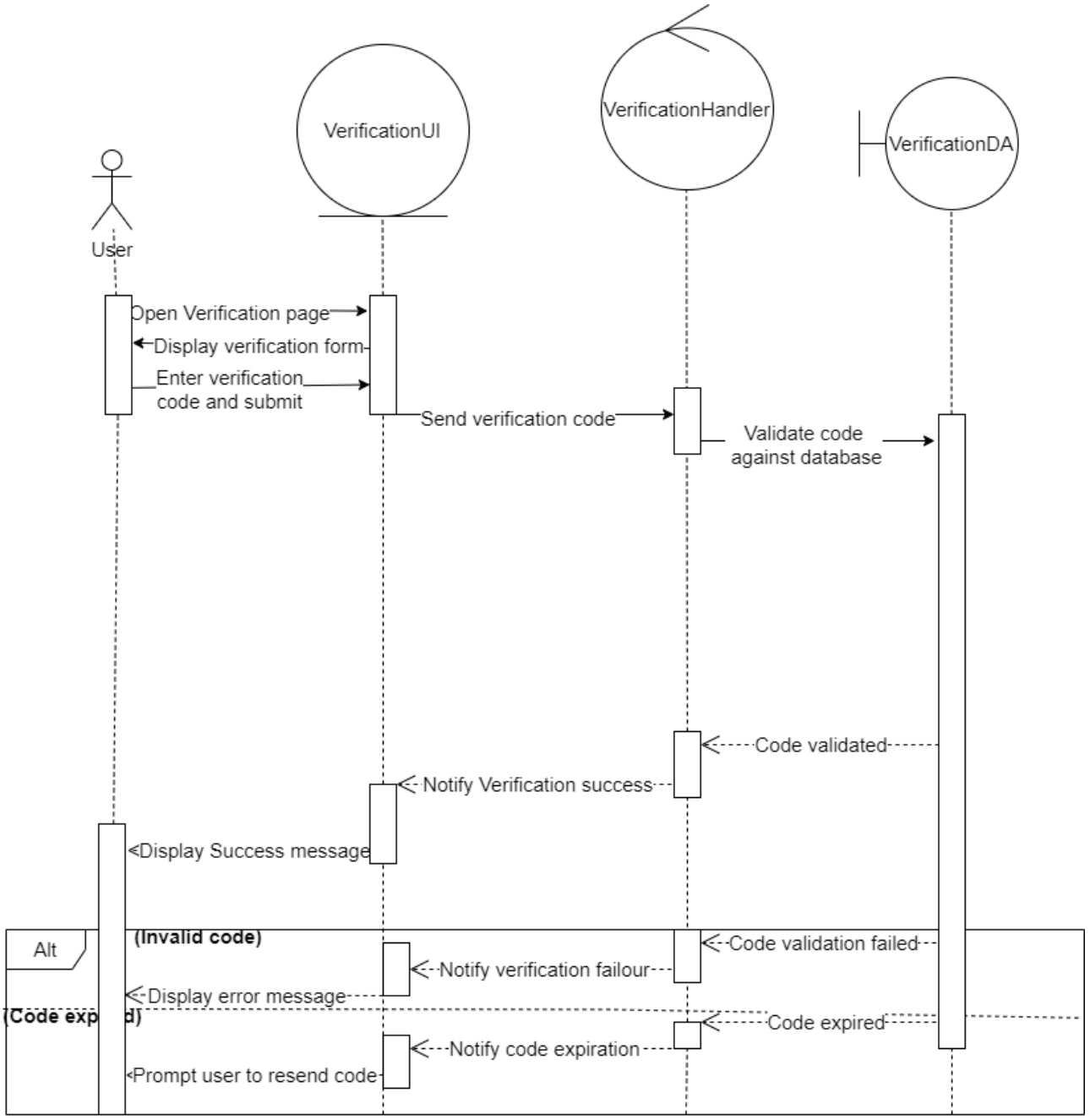


Figure 4.5 Sequence Diagram for <Verification process>

c) SD003:Sequence Diagram for Password Reset Process

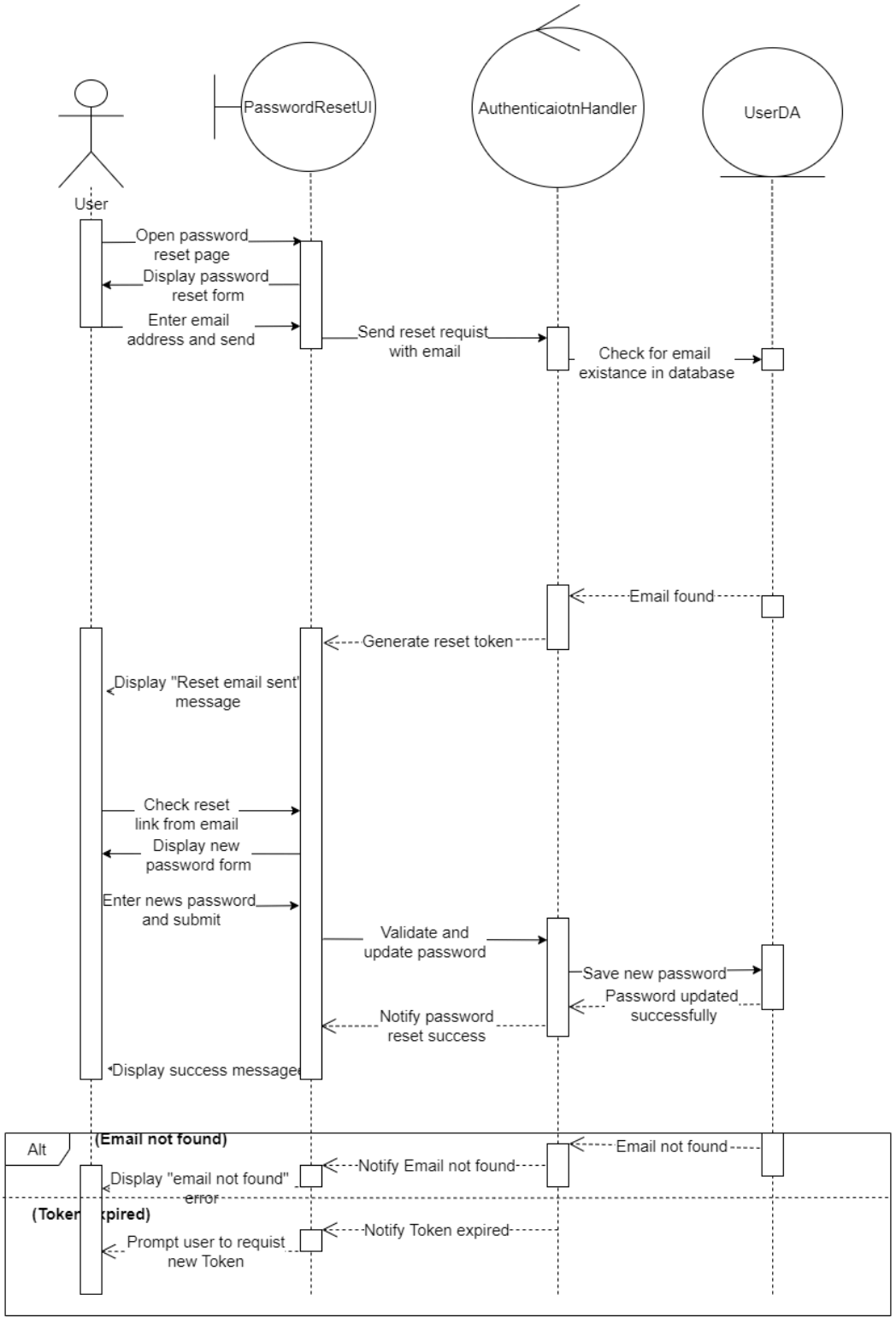


Figure 4.6 Sequence Diagram for <Password Reset Process>

4.2.2 **P002: <Learning and Self-Assessment> Subsystem**

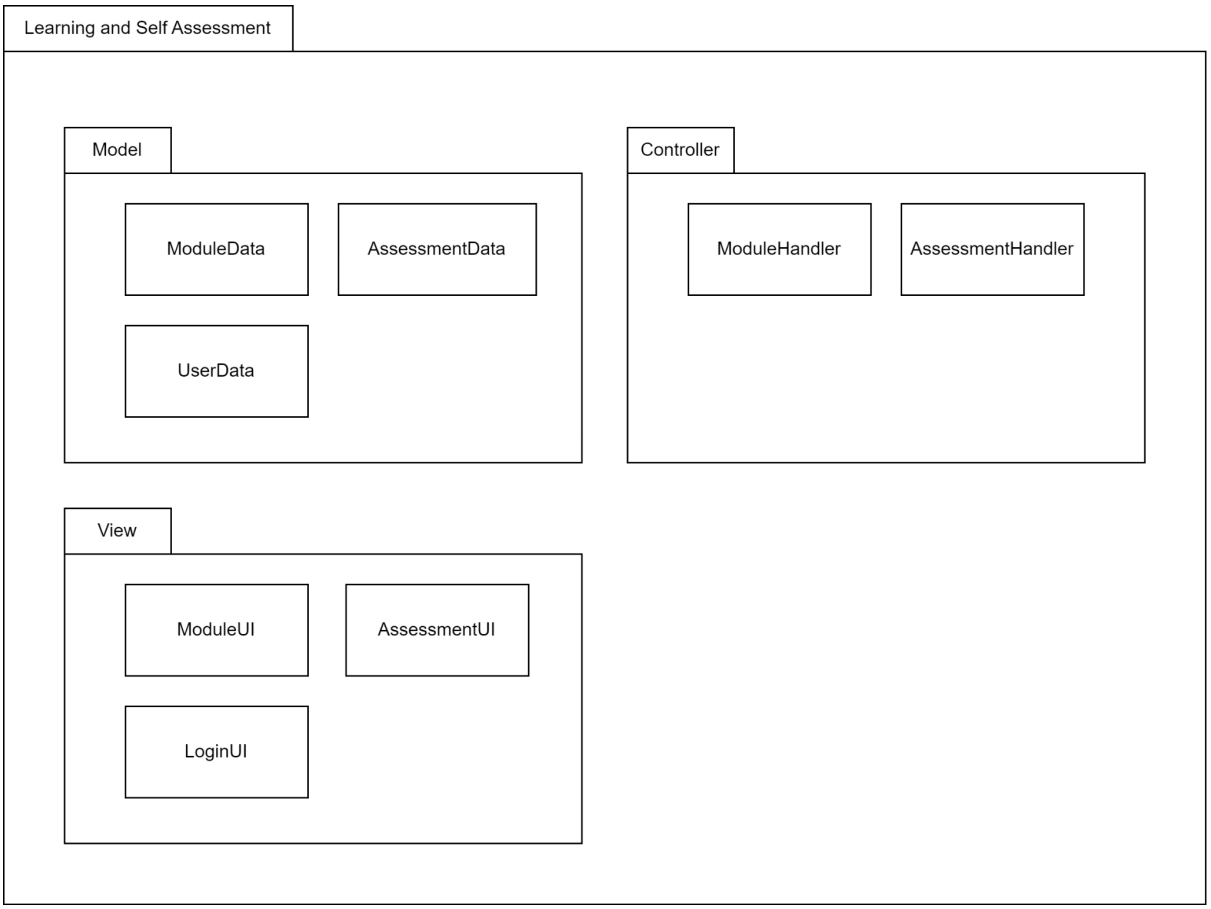


Figure 4.7: Package Diagram for <Learning and Self Assessment>

4.2.2.1 Class Diagram

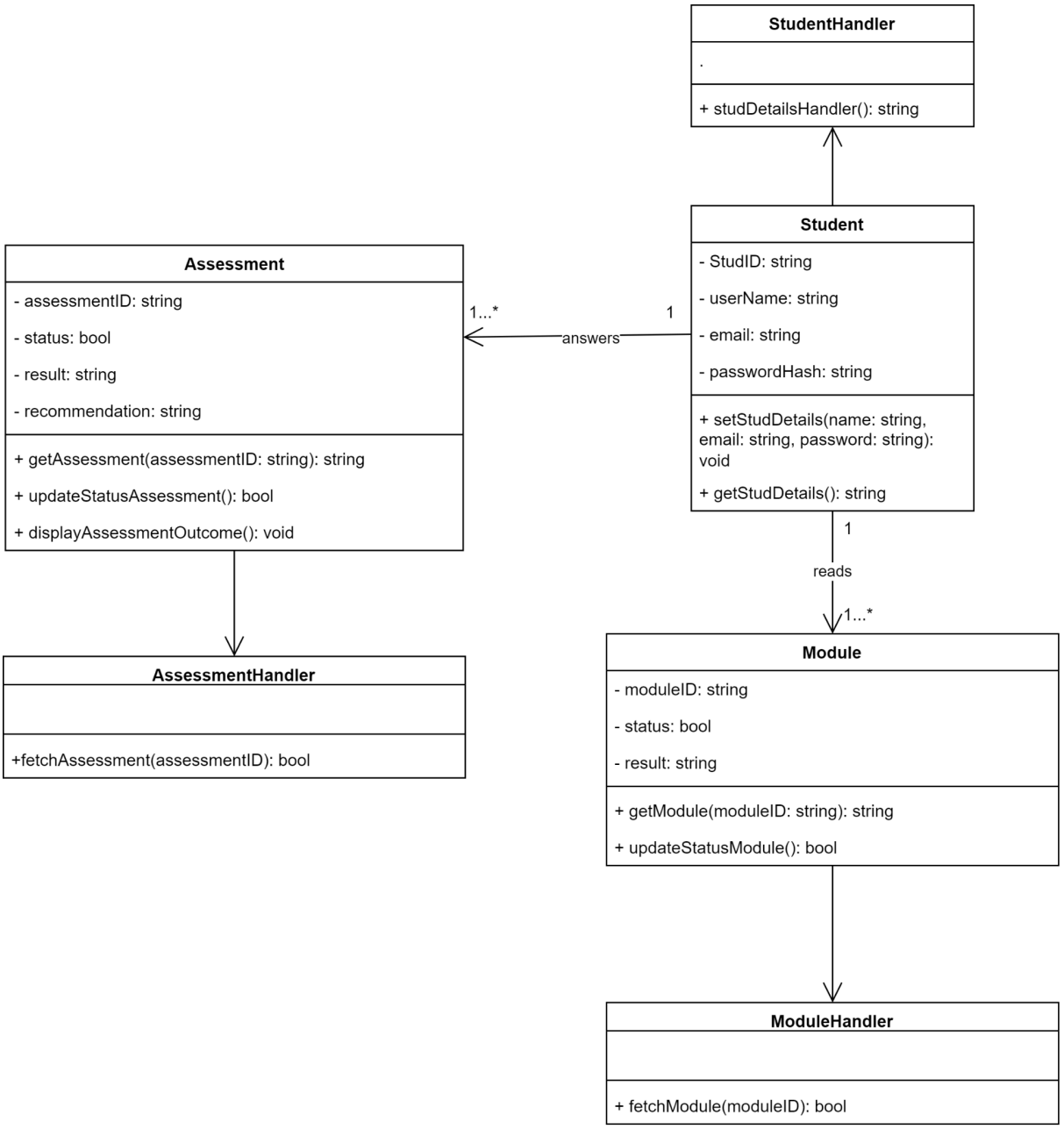


Figure 4.8: Class Diagram for <Learning and Self Assessment>

Entity Name	Assessment
Method Name	getAssessment()
Input	assessmentID
Output	string
Algorithm	1. Start. 2. Fetch the assessment that the user selected. 3. Returns the assessment or shows failure. 4. End.

Entity Name	Assessment
Method Name	updateStatusAssessment()
Input	None
Output	bool
Algorithm	1. Start. 2. Access the status of the current assessment. 3. If assessment is done, return true. 4. If the assessment was left halfway through, return false. 5. End

Entity Name	Assessment
Method Name	displayAssessmentOutcome()
Input	None
Output	Grade, result, and recommendation of assessment.
Algorithm	1. Start. 2. Check the status of the current assessment. 3. If true, display grade, result and recommendation of the assessment to the user. 4. If false, display an error to the user and prompt them to finish the assessment first. 5. End.

Entity Name	AssessmentHandler

Method Name	fetchAssessment()
Input	assessmentID
Output	bool
Algorithm	1. Start. 2. Fetch assessment requested by the user. 3. Returns true if the assessment is retrieved, false if not yet. 4. End.

Entity Name	Module
Method Name	getModule()
Input	moduleID
Output	string
Algorithm	1. Start. 2. Fetch the module that the user selected. 3. Returns the module or shows failure. 4. End.

Entity Name	Module
Method Name	updateStatusModule()
Input	None
Output	bool
Algorithm	1. Start. 2. Access the status of the current module. 3. If the module has been accessed, returns true. 4. Else, return false. 5. End.

Entity Name	ModuleHandler
Method Name	fetchModule()
Input	moduleID

Output	bool
Algorithm	1. Start. 2. Fetch module request by the user. 3. Returns true if the module is retrieved, false if not yet. 4. End.

4.2.2.2 Sequence Diagrams

a) SD004: Sequence diagram for Self-assessment

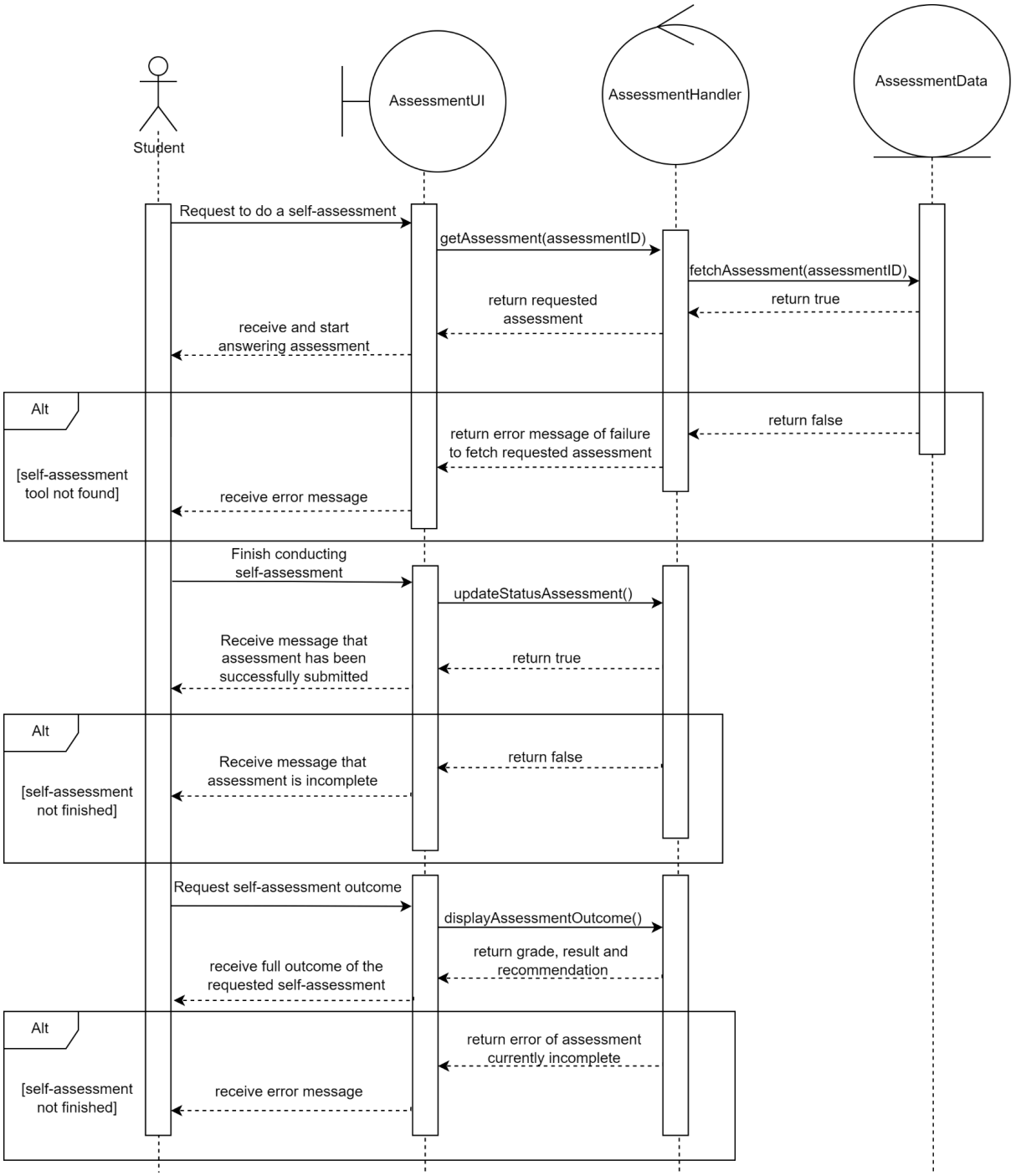


Figure 4.9 Sequence Diagram for <Self-Assessment>

b) SD005: Sequence diagram for Module

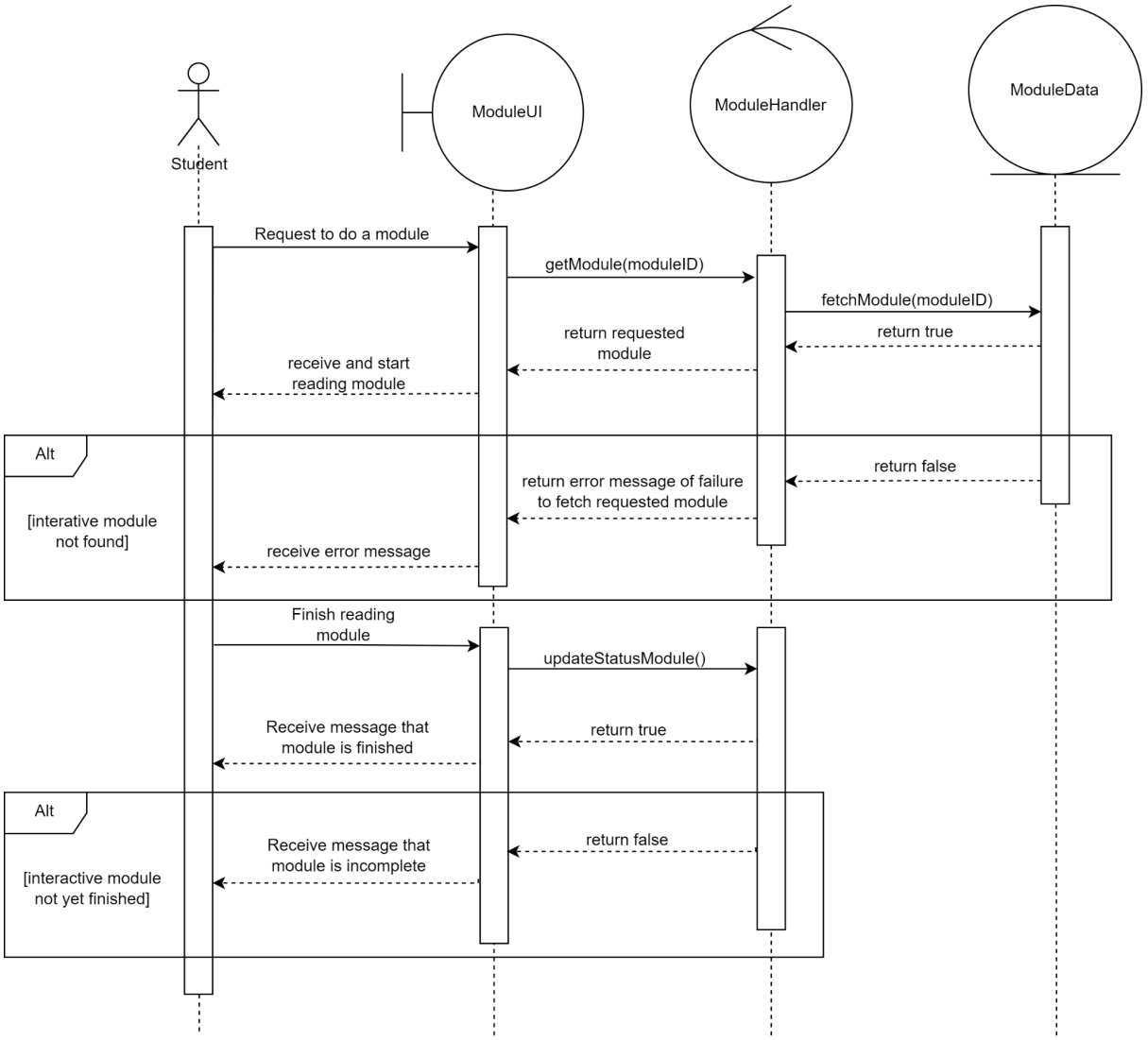


Figure 4.10 Sequence Diagram for <Interactive Modules>

4.2.3 **P003: <Communication and Support> Subsystem**

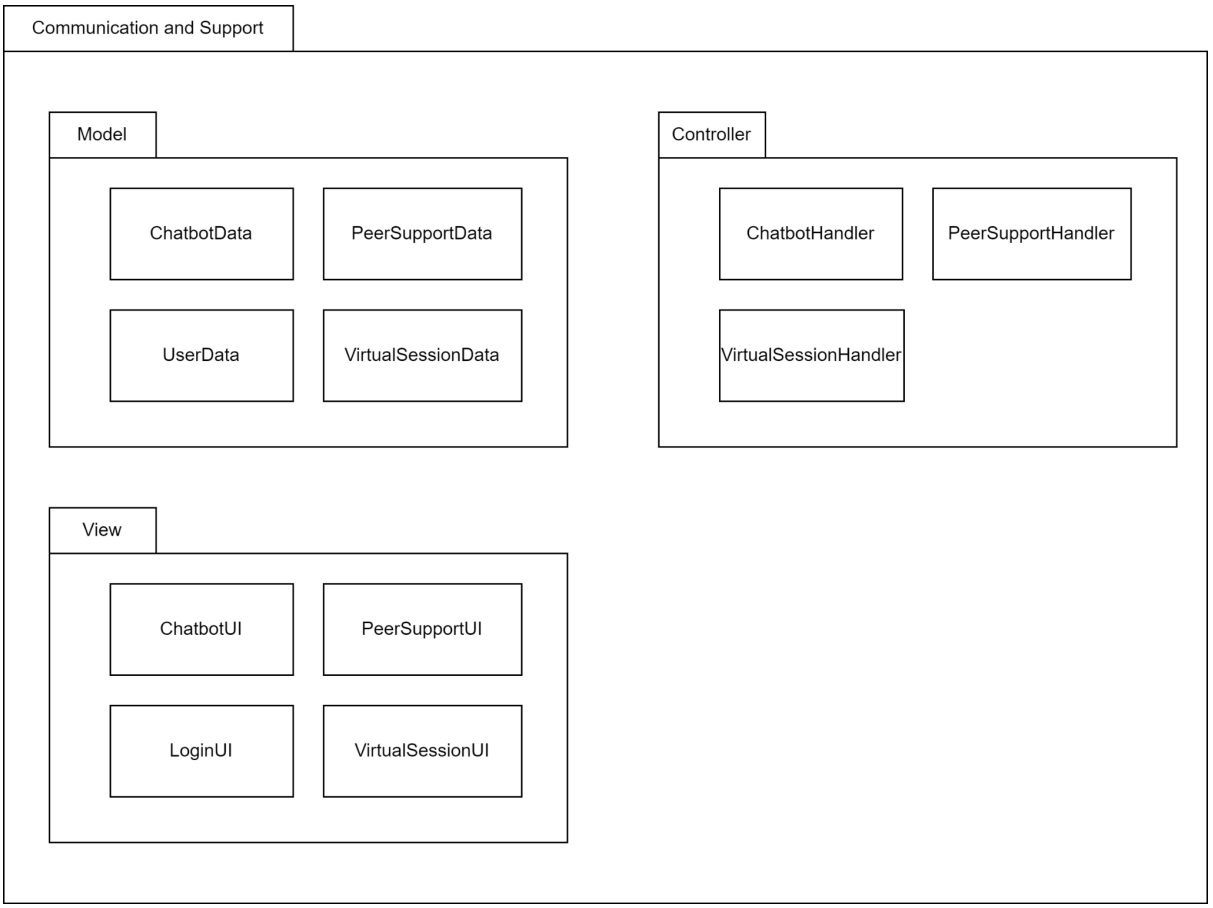


Figure 4.11: Package Diagram for <Communication and Support>

4.2.3.1 Class Diagram

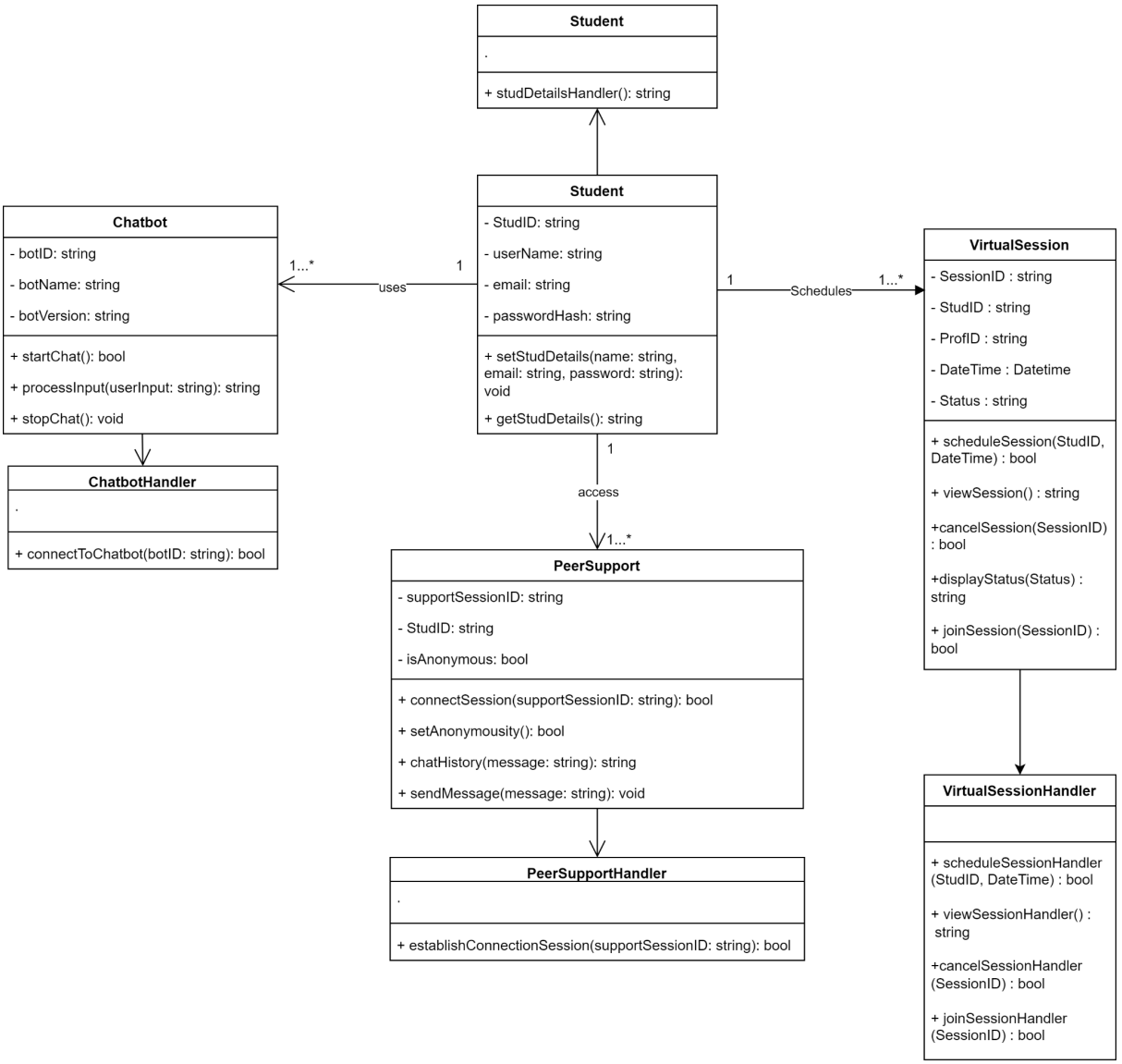


Figure 4.12: Class Diagram for <Communication and Support>

Entity Name	Chatbot
Method Name	startChat()
Input	None
Output	bool
Algorithm	1. Start. 2. Establish connection to the chatbot 3. If established, return true. 4. Else, return false. 5. End

Entity Name	Chatbot
Method Name	processInput()
Input	string
Output	string
Algorithm	1. Start. 2. Read user input and put it into memory. 3. Process the inputs and generate the reply. 4. Display the reply back to the user. 5. End

Entity Name	Chatbot
Method Name	stopChat()
Input	None
Output	None
Algorithm	1. Start. 2. Read user input that they are done with the chat. 3. End connection to chatbot. 4. End

Entity Name	ChatbotHandler
Method Name	connectToChatbot()

Input	botID: string
Output	bool
Algorithm	1. Start. 2. Establish a connection to the chatbot. 3. If established, return true. 4. Else, return false. 5. End

Entity Name	PeerSupport
Method Name	connectSession()
Input	supportSessionID
Output	bool.
Algorithm	1. Start. 2. Check the connection to the the peer support network 3. If connected, return true. 4. Else, return false. 5. End

Entity Name	PeerSupport
Method Name	setAnonymousity()
Input	None
Output	bool
Algorithm	1. Start. 2. Prompt the user if they want to be anonymous. 3. If yes, return true. 4. Else, return false. 5. End

Entity Name	PeerSupport
Method Name	sendMessage()
Input	string

Output	None
Algorithm	1. Start. 2. Read user input. 3. Sends the message to the network 4. End

Entity Name	PeerSupport
Method Name	chatHistory()
Input	string
Output	string
Algorithm	1. Start. 2. Access the chat of current peer support network session 3. Display all messages in the chat for the past 24 hours and incoming messages. 4. End

Entity Name	PeerSupportHandler
Method Name	establishConnectionSession()
Input	supportSessionID: string
Output	bool
Algorithm	1. Start. 2. Establish a connection to the selected peer support network session. 3. If established, return true. 4. Else, return false. 5. End.

Entity Name	VirtualSessionData
Method Name	viewSession()
Input	none
Output	string
Algorithm	1. Start. 2. Fetch Scheduled Virtual session details under the professional. 3. Return details. 4. End.

Entity Name	VirtualSessionData
Method Name	cancelSession()
Input	int
Output	bool
Algorithm	1. Start. 2. Enter session Id to cancel a scheduled session 3. Return success or failure 4. End.

Entity Name	VirtualSessionData
Method Name	joinSession()
Input	int
Output	bool
Algorithm	1. Start. 2. Enter session Id to join a scheduled session 3. Return success or failure of joining 4. End.

Entity Name	VirtualSessionData

Method Name	scheduleSession()
Input	string, DateTime
Output	bool
Algorithm	1. Start. 2. Schedule a session with StudID and DateTime. 3. Return success or failure 4. End.

Entity Name	VirtualSessionData
Method Name	displayStatus()
Input	string
Output	string
Algorithm	1. Start. 2. Call scheduleSessionHandler().VirtualSessionHandler 3. Return true if successful. 4. End.

Entity Name	VirtualSessionHandler
Method Name	scheduleSessionHandler()
Input	string, DateTime
Output	bool
Algorithm	1. Start. 2. Schedule a session with StudID and DateTime. 3. Return true if successful. 4. End.

Entity Name	VirtualSessionHandler
Method Name	cancelSessionHandler()

Input	SessionId
Output	bool
Algorithm	1. Start. 2. Call cancelSession().VirtualSessionData 3. Return true if successful. 4. End.

Entity Name	VirtualSessionHandler
Method Name	joinSessionHandler()
Input	SessionId
Output	bool
Algorithm	1. Start. 2. Validate sessionId 3. Mark user as joined in the session. 4. Return true. 5. End.

4.2.3.2 **Sequence Diagram**

a) SD006: Sequence diagram for Chatbot

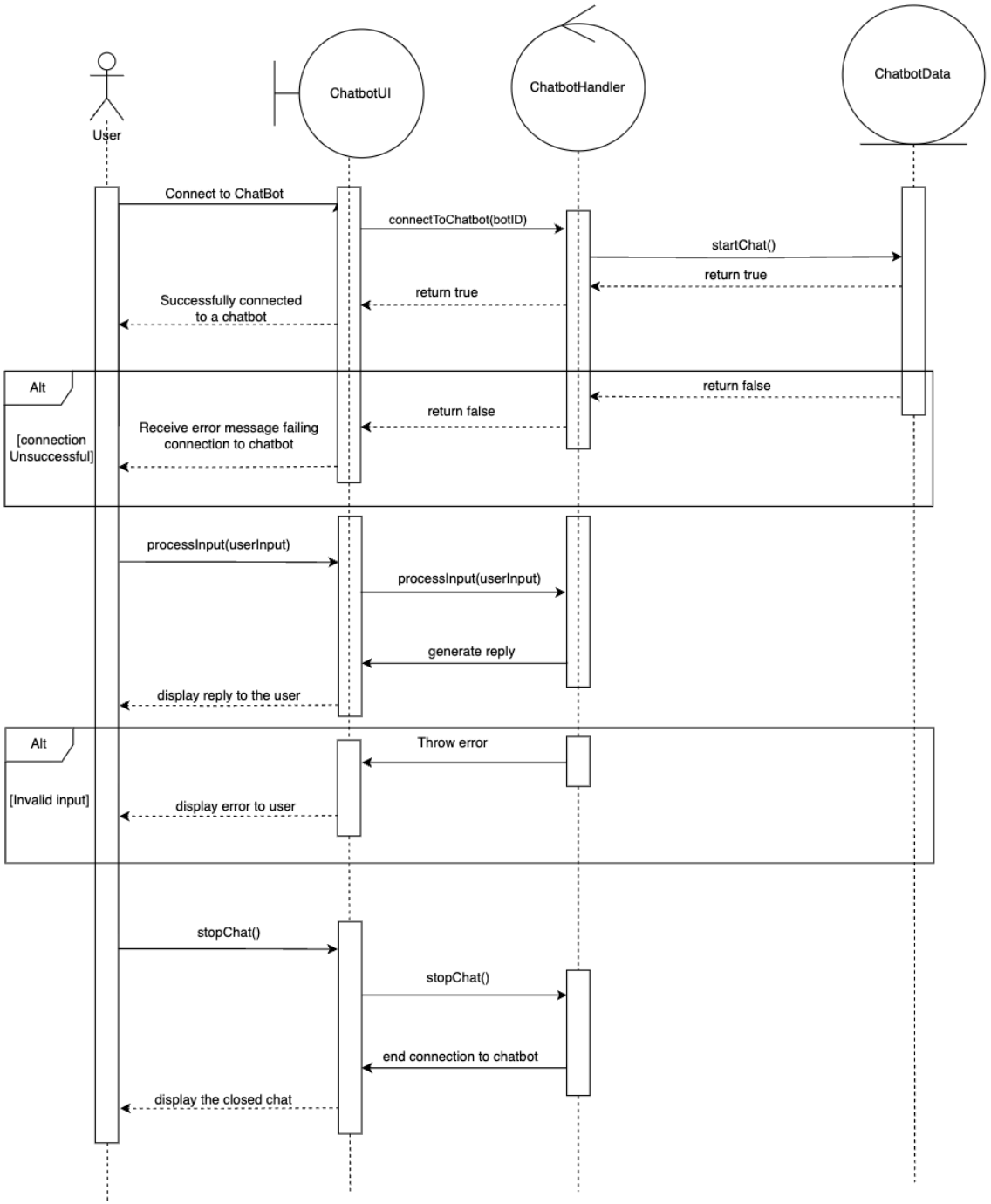


Figure 4.13 Sequence Diagram for <Chatbot>

b) SD007: Sequence diagram for Peer Support

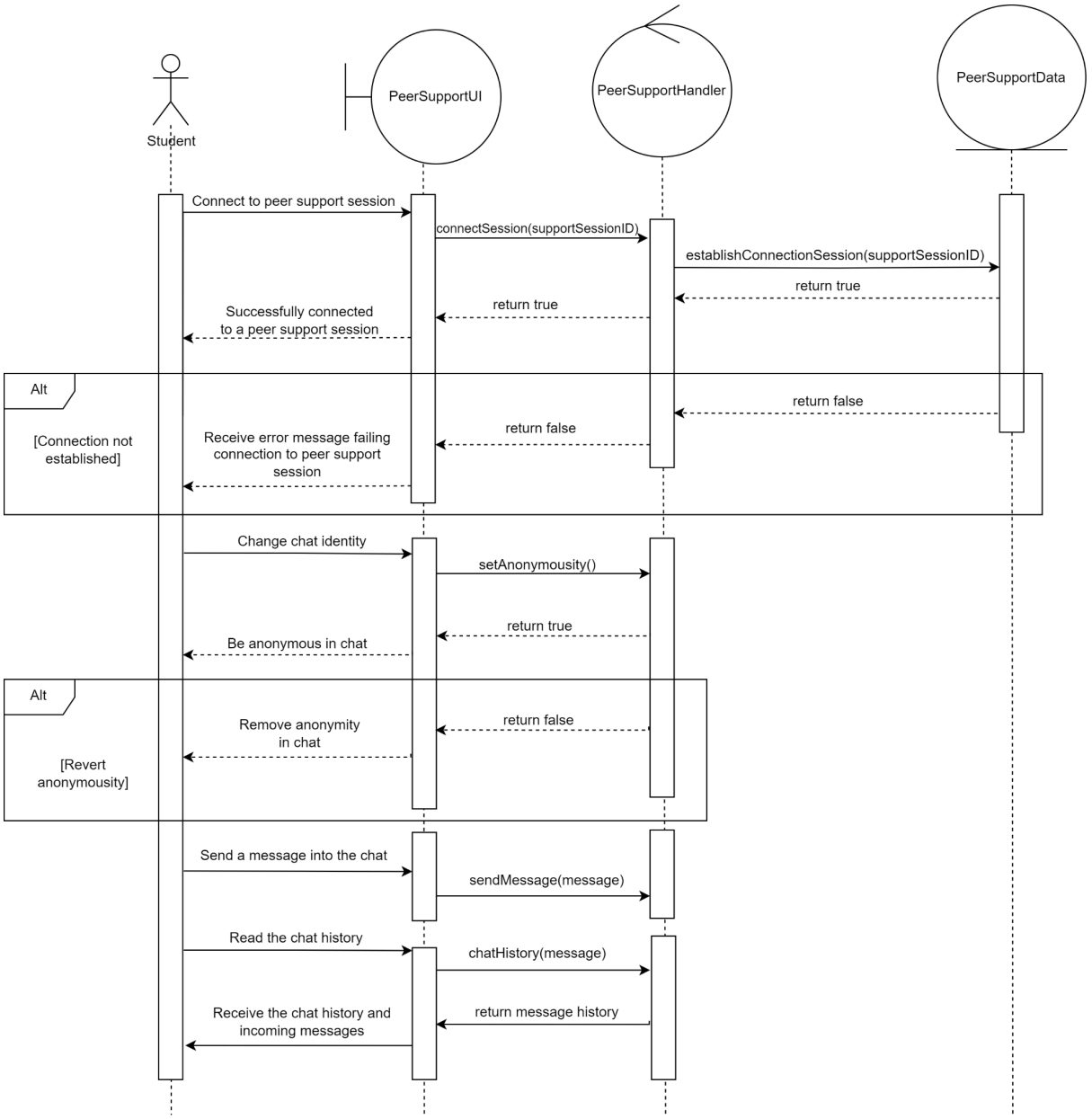


Figure 4.14 Sequence Diagram for <Peer Support>

c) SD008: Sequence diagram for Scheduling Virtual Session

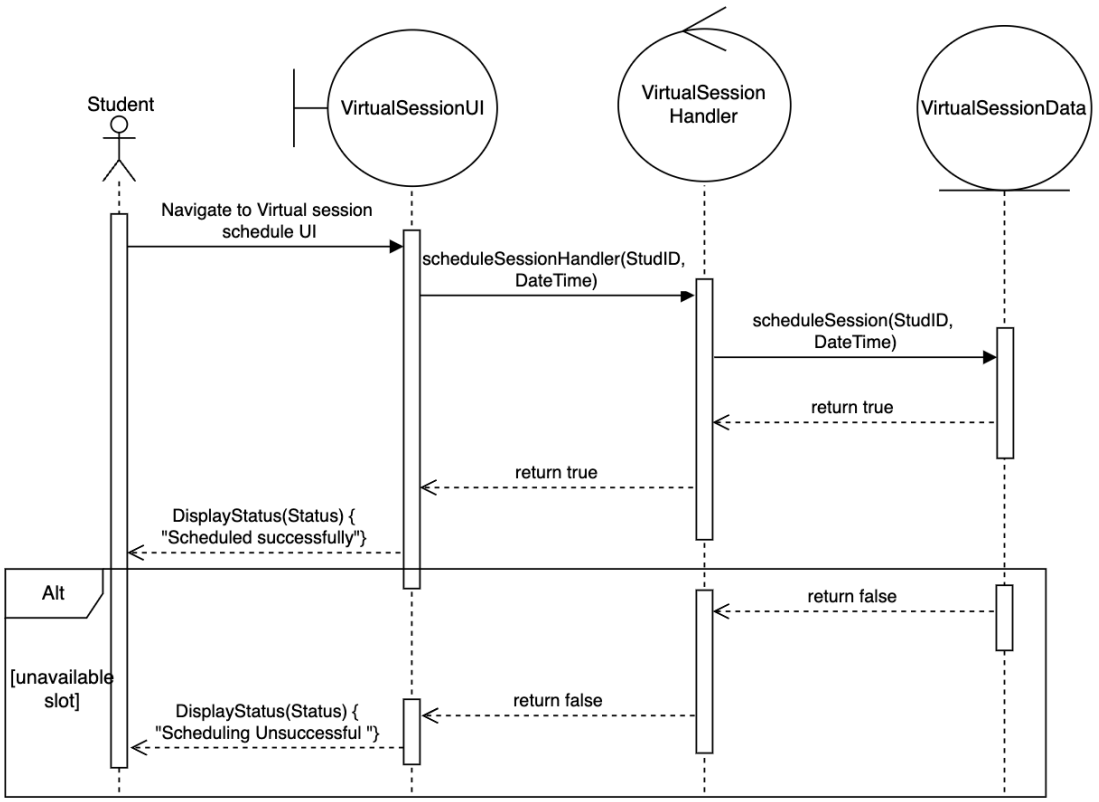


Figure 4.15 Sequence Diagram for <Scheduling Virtual Session>

4.2.4 **P004: <Professional Management System> Subsystem (ANJUM)**

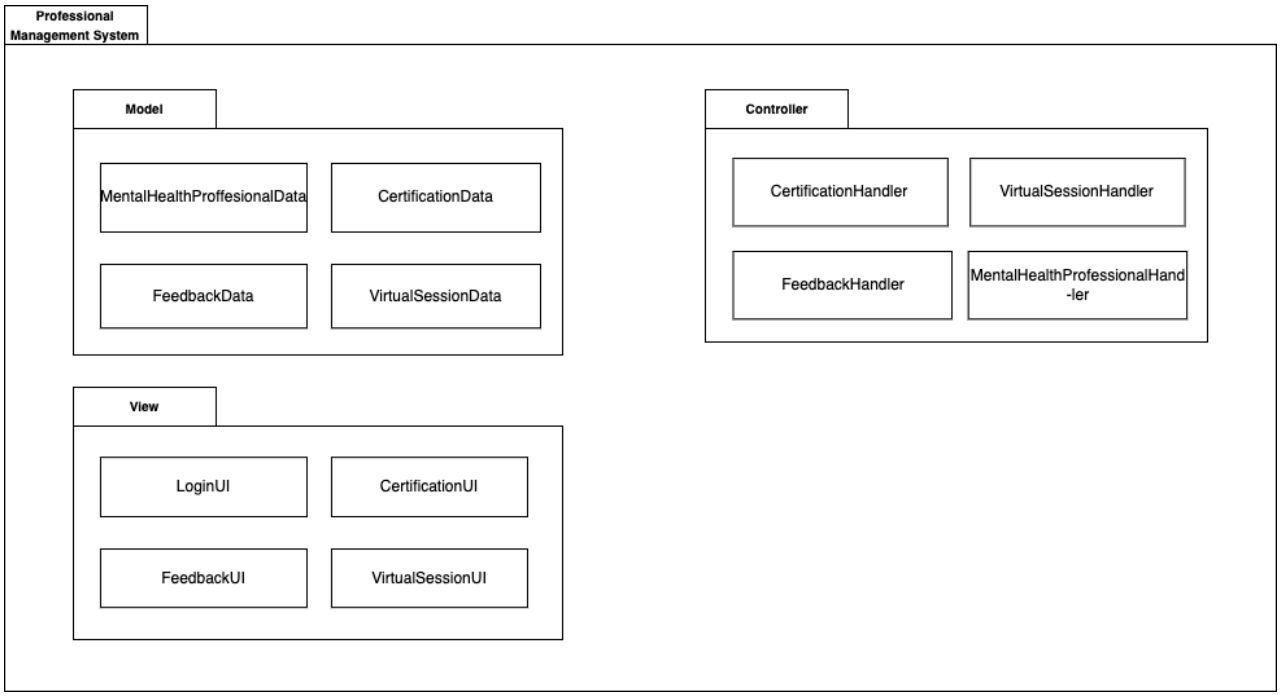


Figure 4.16 Package Diagram for <Professional Management System>

4.2.4.1 **Class Diagram**

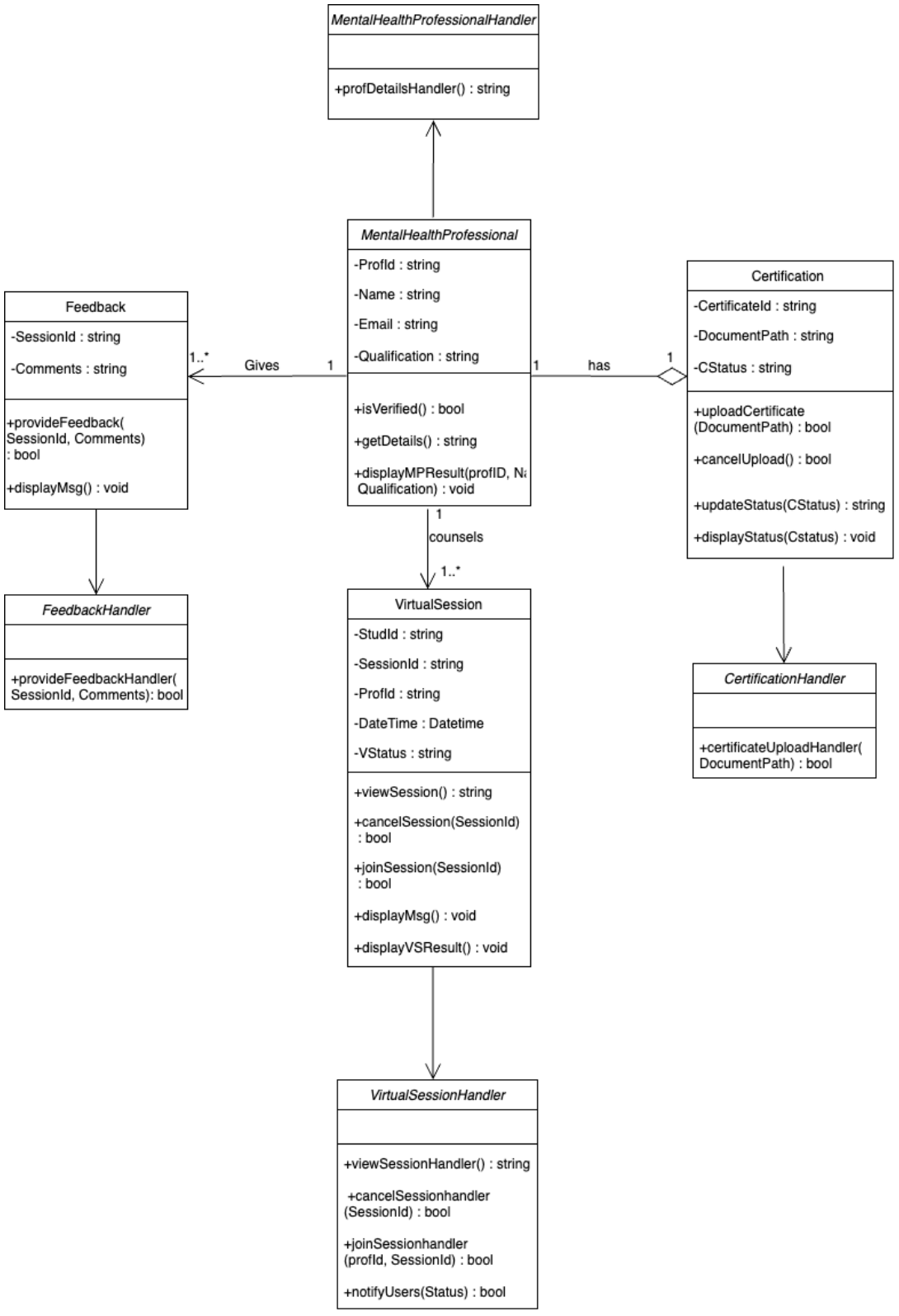


Figure 4.17 Class Diagram for <Professional Management System>

Entity Name	MentalHealthProfessionalData

Method Name	isVerified()
Input	None
Output	bool
Algorithm	1. Start. 2. Check if verification status is true. 3. Return result. 4. End.

Entity Name	MentalHealthProfessionalData
Method Name	getDetails()
Input	None
Output	string
Algorithm	1. Start. 2. Fetch professional details from attributes. 3. Return details. 4. End.

Entity Name	CertificateData
Method Name	uploadCertificate()
Input	DocumentPath
Output	bool
Algorithm	1. Start. 2. Fetch professional certificate from the specified document path 3. Upload the document 4. Return success or failure 5. End

Entity Name	CertificationHandler
Method Name	certificateUploadHandler()
Input	DocumentPath

Output	bool
Algorithm	1. Start. 2. Call uploadCertificate().Certification 3. Return true if successful. 4. End.

Entity Name	FeedbackData
Method Name	provideFeedback()
Input	int, string
Output	bool
Algorithm	1. Start. 2. Enter Session Id to give feedback 3. Give the feedback and submit 4. Return success or failure 5. End.

Entity Name	FeedbackData
Method Name	provideFeedbackHandler()
Input	int, string
Output	bool
Algorithm	1. Start. 2. Call provideFeedback().FeedbackData 3. Return true if successful. 4. End.

Entity Name	VirtualSessionData
Method Name	viewSession()
Input	none
Output	string

Algorithm	1. Start. 2. Fetch Scheduled Virtual session details under the professional. 3. Return details. 4. End.

Entity Name	VirtualSessionData
Method Name	cancelSession()
Input	int
Output	bool
Algorithm	1. Start. 2. Enter session Id to cancel a scheduled session 3. Return success or failure 4. End.

Entity Name	VirtualSessionData
Method Name	joinSession()
Input	int
Output	bool
Algorithm	1. Start. 2. Enter session Id to join a scheduled session 3. Return success or failure of joining 4. End.

Entity Name	ManageVirtualSessionHandler
Method Name	cancelSessionHandler()
Input	SessionId
Output	bool

Algorithm	1. Start. 2. Call cancelSession().VirtualSessionData 3. Return true if successful. 4. End.

Entity Name	ManageVirtualSessionHandler
Method Name	joinSessionHandler()
Input	SessionId
Output	bool
Algorithm	1. Start. 2. Validate sessionId 3. Mark user as joined in the session. 4. Return true. 5. End.

Entity Name	ManageVirtualSessionHandler
Method Name	notifyUser()
Input	ProfId, StudId , status
Output	bool
Algorithm	1. Start. 2. Send the status of a virtual Session (scheduled, canceled) to the specified professional and student respective of their Id. 3. Return true if successful. 4. End

4.2.4.2 **Sequence Diagram**

a) SD009: Sequence diagram for Viewing Mental Health Professional Details

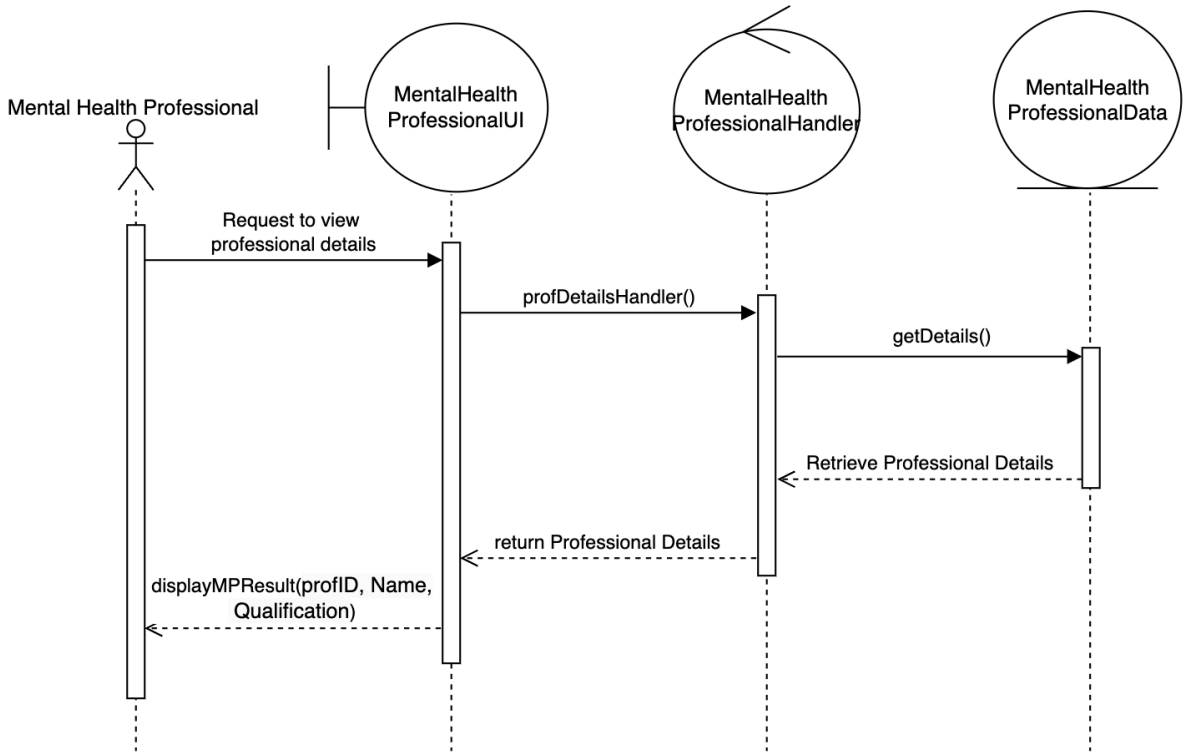


Figure 4.18 Sequence Diagram for <Viewing Mental Health Professional Details>

b) SD010: Sequence diagram for Verify Professional Certificate

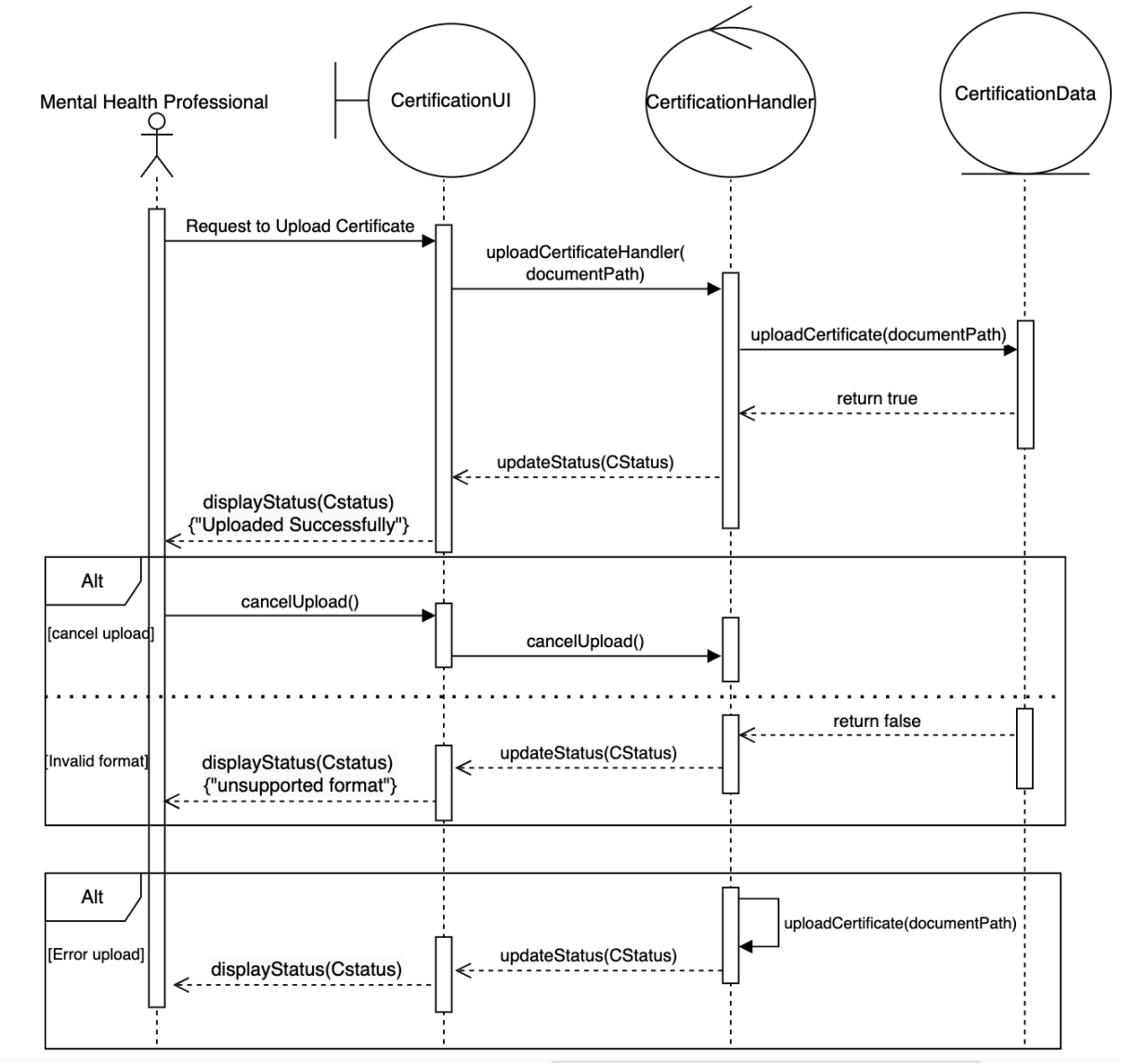


Figure 4.19 Sequence Diagram for <Verify Professional Certificate>

c) SD011: Sequence diagram for Providing Feedback

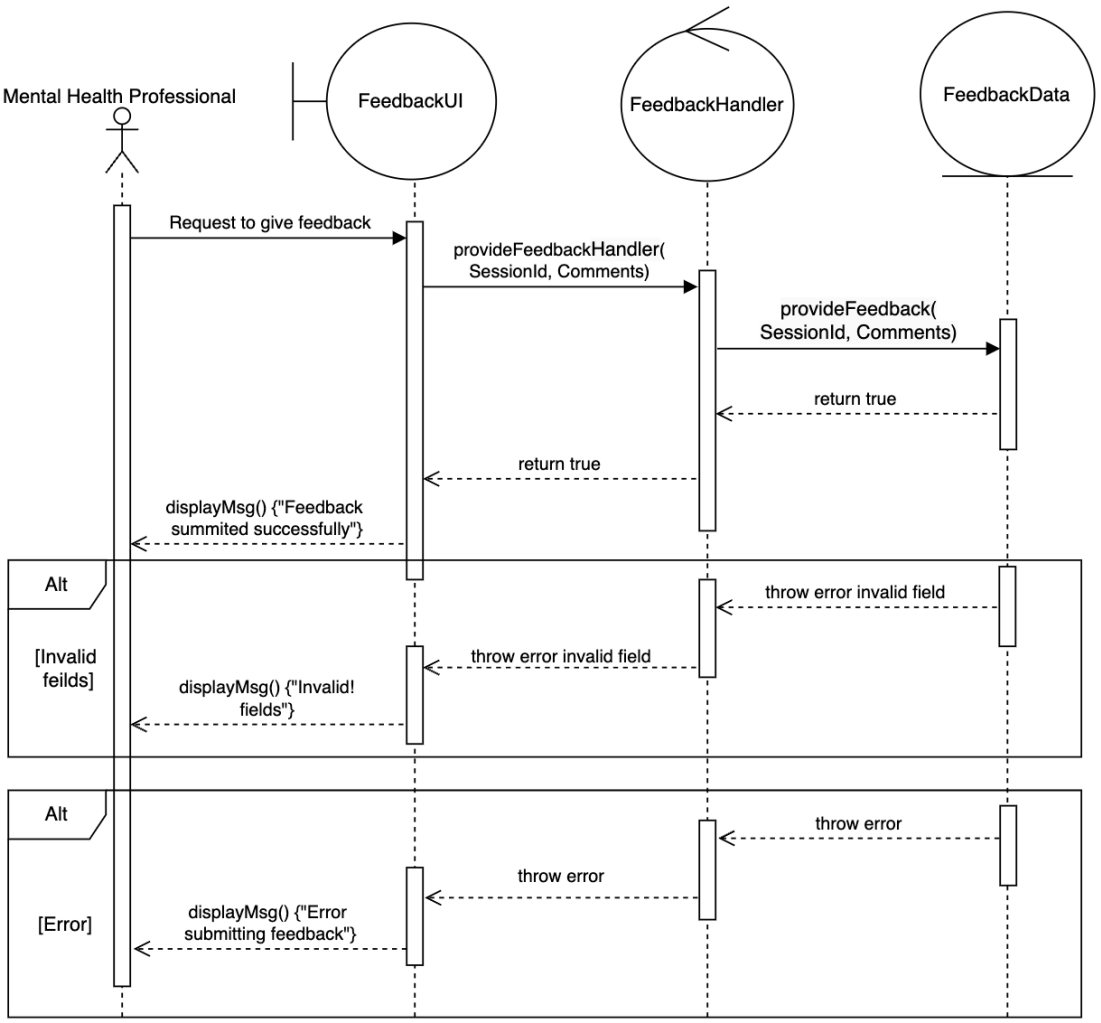


Figure 4.20 Sequence Diagram for <Providing Feedback>

d) SD012: Sequence diagram for Viewing Scheduled Virtual Sessions

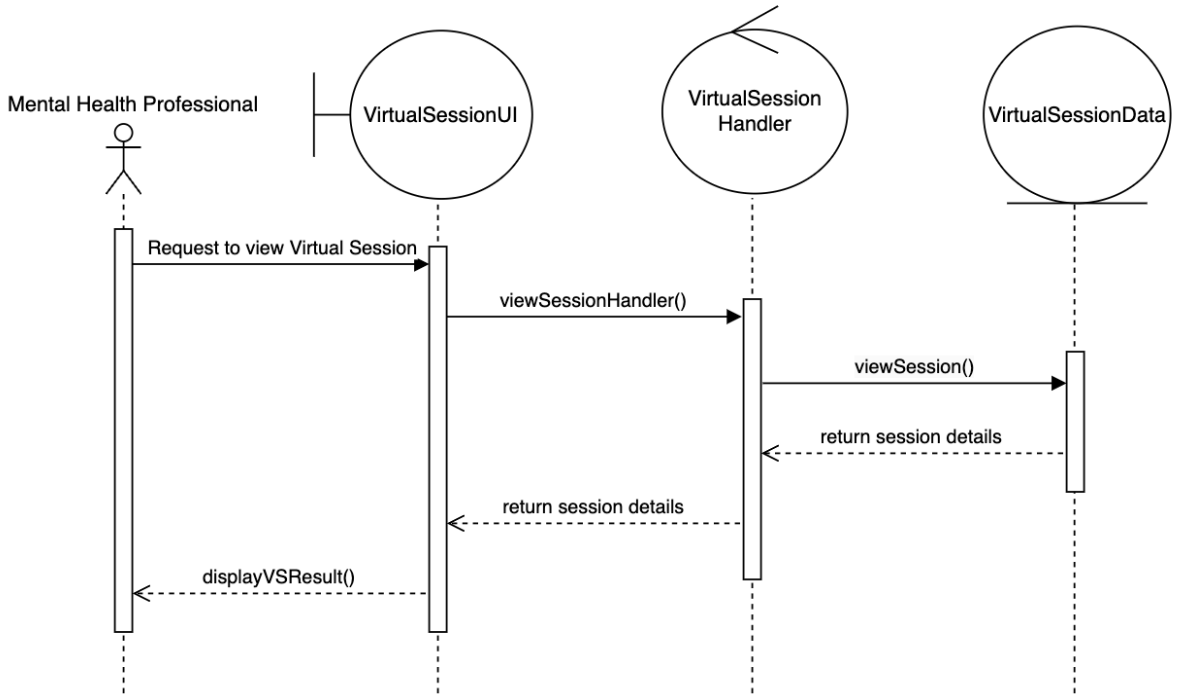


Figure 4.21 Sequence Diagram for <Viewing Scheduled Virtual Sessions>

e) SD013: Sequence diagram for Canceling Scheduled Virtual Session

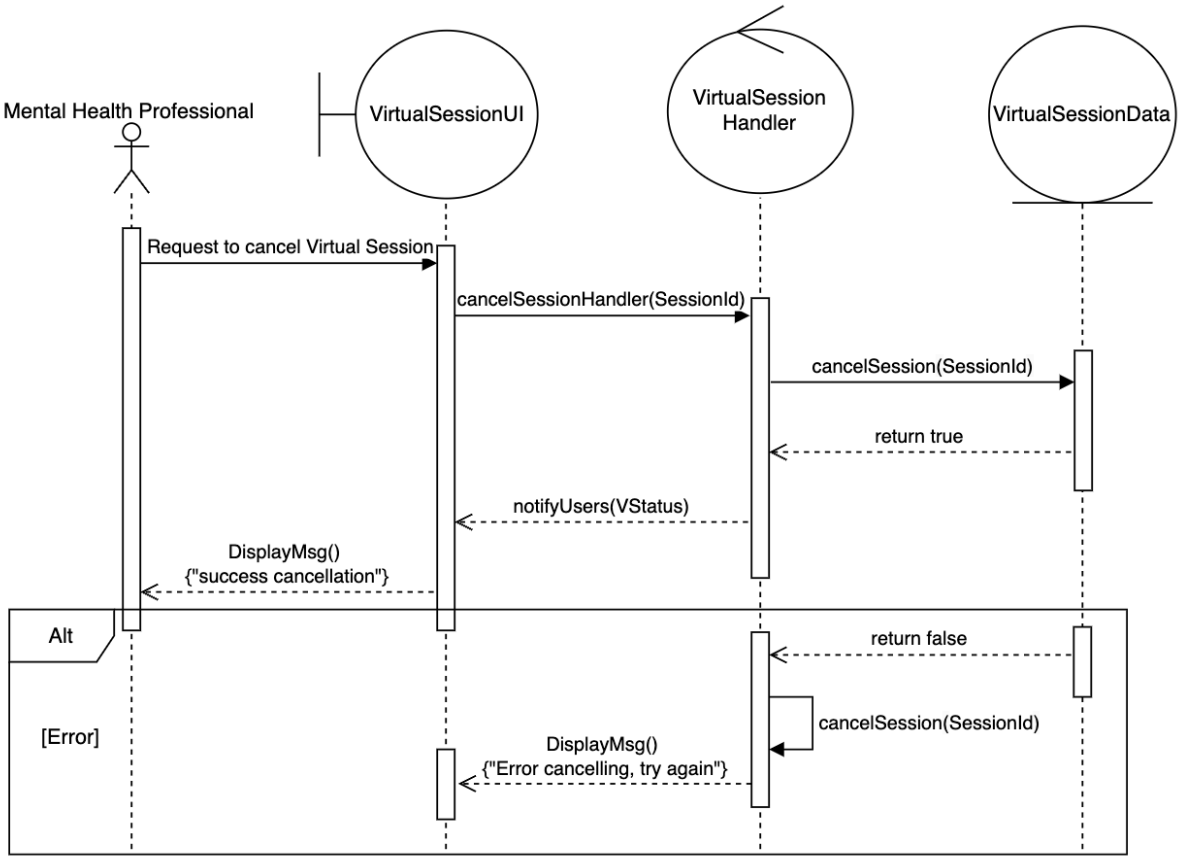


Figure 4.22 Sequence Diagram for <Cancelling Scheduled Virtual Session>

f) SD014: Sequence diagram for Joining Virtual Sessions

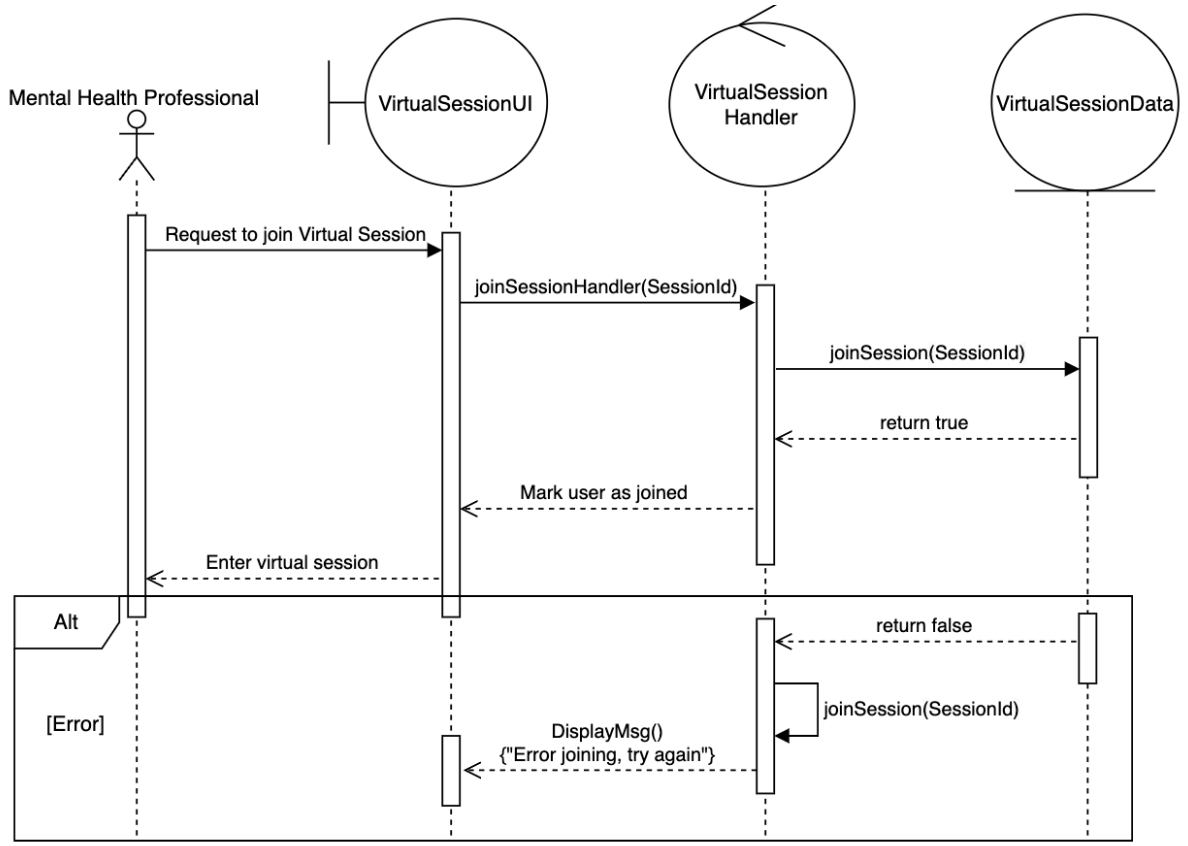


Figure 4.23 Sequence Diagram for <Joining Virtual Sessions>

4.2.5 **P005: <System Monitoring> Subsystem**

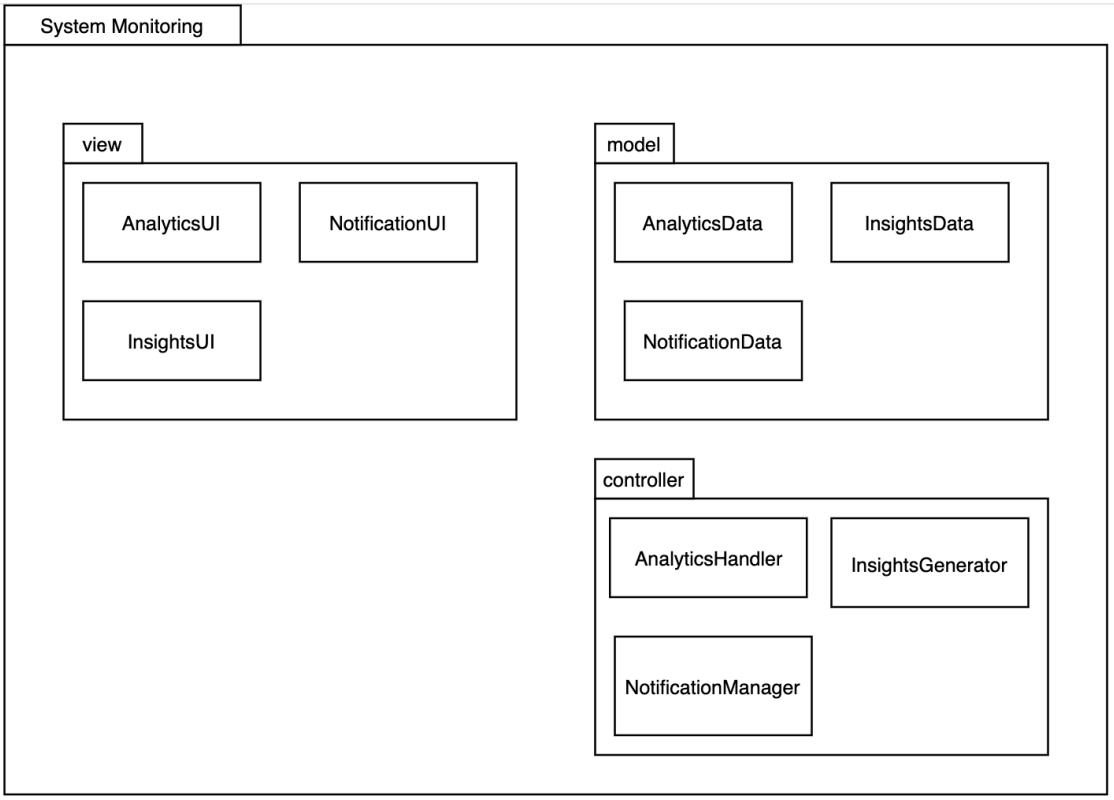


Figure 4.24. Package Diagram for <System Monitoring>

4.2.5.1 Class Diagram

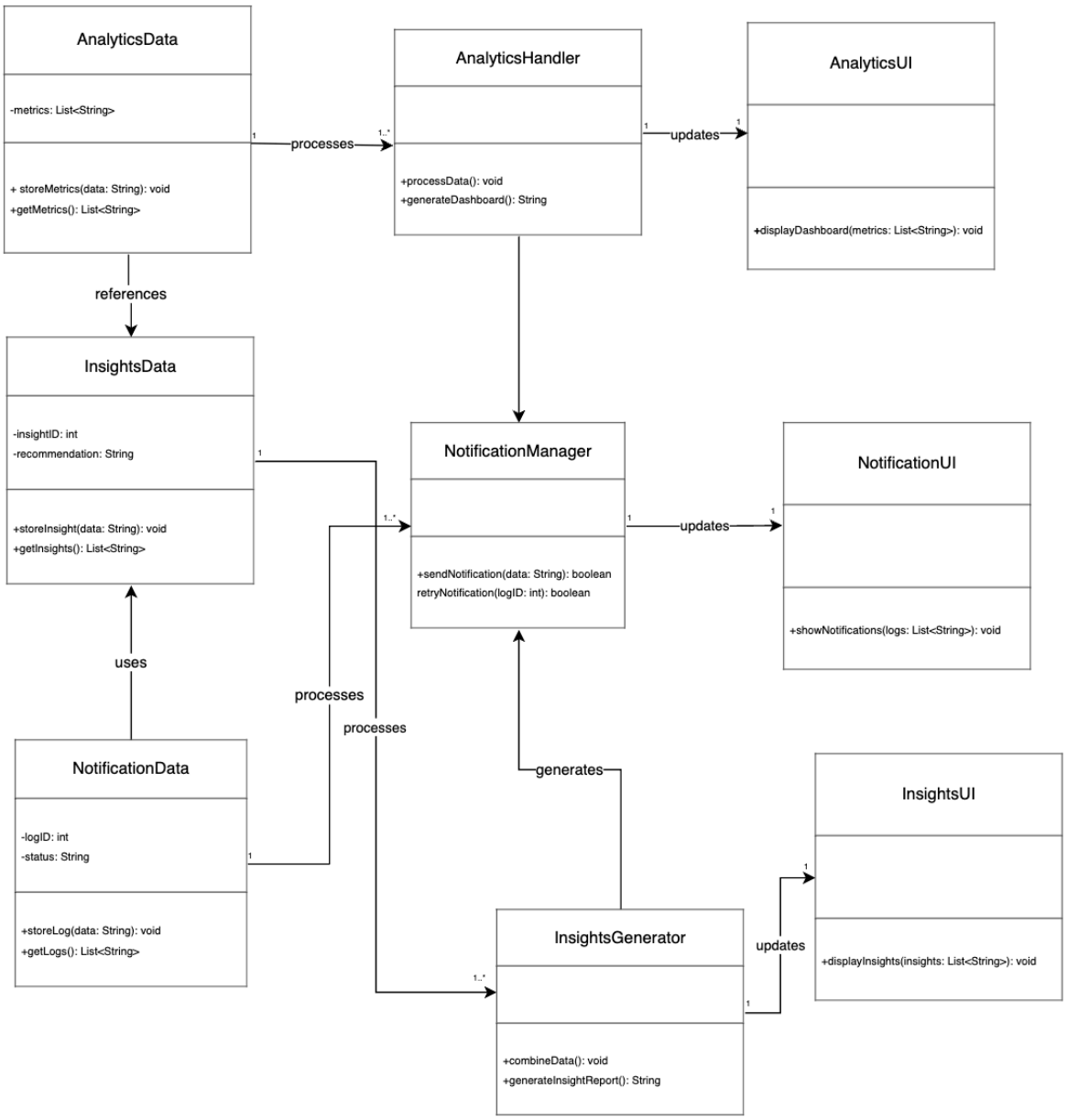


Figure 4.25 Class Diagram for <System Monitoring>

Entity Name	AnalyticsHandler
Method Name	processData()
Input	-
Output	void
Algorithm	1. Start. 2. Get metrics from Analytics Data 3. Process the metrics to compute averages and trends. 4. Store processed metrics. 5. End.

Entity Name	AnalyticsHandler
Method Name	generateDashboard()
Input	-
Output	String (dashboard data)
Algorithm	1. Start 2. Get processed data from processData() 3. Process the metrics to compute averages and trends. 4. Store processed metrics. 5. End.

Entity Name	NotificationManager
Method Name	sendNotification()
Input	String (notification content)
Output	boolean
Algorithm	1. Start. 2. Validate the input notification content. 3. Send the notification through the notification service. 4. If successful, return true. 5. If failed, return false. 6. End.

Entity Name	NotificationManager
Method Name	retryNotification()

Input	Integer (Log ID)
Output	boolean
Algorithm	1. Start. 2. Identify the failed notification using the Log ID. 3. Try to resend the notification. 4. If the operation succeeds, update the log status and return true. 5. If the operation fails, return false. 6. End.

Entity Name	InsightsGenerator
Method Name	combineData()
Input	-
Output	void
Algorithm	1. Start. 2. Get data from AnalyticsData, NotificationData, and InsightsData. 3. Merge data into a single structure. 4. Store the combined data. 5. End

Entity Name	InsightsGenerator
Method Name	generateInsightReport()
Input	-
Output	String (Insight report)
Algorithm	1. Start. 2. Get combined data from combineData() 3. Analyze data to generate recommendations and insights. 4. Format the insights into a report layout. 5. Return the report as a string. 6. End.

Entity Name	AnalyticsUI
Method Name	displayDashboard()

Input	List<String> (Metrics)
Output	void
Algorithm	1. Start. 2. Get metrics from AnalyticsHandler. 3. Display metrics in a graphical dashboard. 4. Display the dashboard to the user. 5. End.

4.2.5.2 Sequence Diagram

a) SD015: Sequence diagram for Analytics

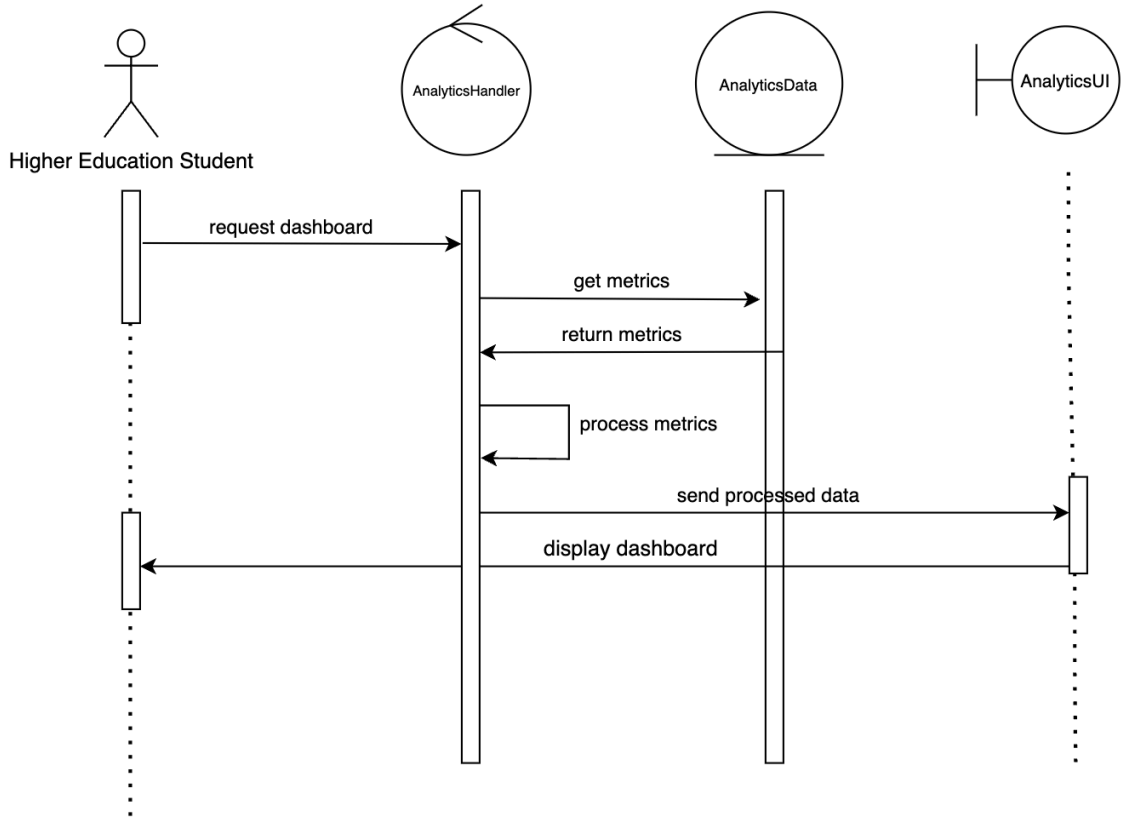


Figure 4.26 Sequence Diagram for <Analytics>

sequence diagrams:
https://drive.google.com/drive/folders/1alo9Va63ODTUoRWPXDUg9-Dn6UOCZK9P?usp=drive_link

5 Data Design

5.1 Data Description

The major data or systems entities are stored into a relational database named as Serene_System_DB processed and organized into *13* entities as listed in Table 5.1.

No.	Entity Name	Description
1	Registration	Stores registration details for users, including personal information and account creation status. The primary key for this entity is RegistrationID
2	Authentication	Manages user authentication details, including attributes UserID such as AuthID, UserID , LoginTimestamp , and Success. The primary key for this entity is AuthID.
3	Certification	Stores certification details for professionals with attributes like CertificateID, DocumentPath, CStatus. The primary key for this entity is CertificateID.
4	Virtual Sessions	Stores details of virtual sessions. This entity contains attributes like SessionID, DateTime, Status. The primary key for this entity is SessionID.
5	Feedback	Stores feedback provided by professionals for students after virtual sessions. This contains SessionID, Comments. SessionID is the foreign key.
6	Assessment	Stores self-assessment tools and their respective data of each user securely. This contains assessmentID, status, result and recommendation. The primary key for this entity is assessmentID.
7	Module	Stores modules to be accessed and read by users with attributes such as moduleID, status and result. The primary key for this entity is moduleID.
8	Chatbot	Stores user input and generates a response to it with attributes such as botID, botName and

		botVersion. The primary key for this entity is botID.
9	PeerSupport	Stores the current user session's details including sessionID, userID and isAnonymous. The primary key is sessionID, while userID is the foreign key.
10	Mental Health Professional	Stores information about mental health with attributes ProfID, Name, Email, Qualification. The primary key for this entity is ProfID.
11	Analytics	Stores system performance and user engagement metrics with attributes MetricID, MetricName, Value, and Timestamp. The primary key for this entity is MetricID, which uniquely identifies each recorded metric.
12	Notification	Stores information about system notifications with attributes LogID, Status, and Message. The primary key for this entity is LogID, which uniquely identifies each notification record.
13	Insights	Stores information about generated insights and recommendations with attributes InsightID, Recommendation. The primary key for this entity is InsightID, which uniquely identifies each insight entry.

Table 5.1: Description of Entities in the Database

5.2 Data Dictionary

5.2.1 Entity: <Registration>

Attribute Name	Type	Description
registrationID	String	Unique identifier for the registration process.
StudID	String	Name provided by the user during registration.

email	String	Email address used for user registration.
password	String	Password chosen by the user for authentication.
registrationTimestamp	DateTime	The time and date when the registration occurred.

5.2.2 Entity: <Authentication>

Attribute Name	Type	Description
authenticationID	String	Unique identifier for the authentication process.
email	String	Email address used to authenticate the user.
password	String	Password entered by the user during login.
authToken	String	Generated token after successful authentication.
authTimestamp	DateTime	The time and date of the authentication event.

5.2.3 Entity: <Certification>

Attribute Name	Type	Description
CertificateID {PK}	String	Uniquely identifies a certificate
DocumentPath	String	Path to find the Certificate from user local system
CStatus	string	Denotes the Status of the Certificate

5.2.4 **Entity: <Virtual Session>**

Attribute Name	**Type**	**Description**
SessionID {PK}	String	Unique identity of a session
DateTime	DateTime	Date and Time of a session
Status	string	Status of a virtual session

5.2.5 **Entity: <Feedback>**

Attribute Name	**Type**	**Description**
SessionID {FK}	string	Unique identity of a session
Comments	string	Comments given for a particular virtual session

5.2.6 **Entity: <Mental Health Professional>**

Attribute Name	**Type**	**Description**
ProfID {PK}	String	Uniquely identifies each mental health professional.
Name	String	Name of the mental health professional.
Email	String	Email of the mental health professional.
Qualification	String	Qualification of the mental health professional.

5.2.7 **Entity: <Assessment>**

Attribute Name	**Type**	**Description**
assessmentID {PK}	string	Unique identity of self-assessment tool
result	string	The outcome of using the self-assessment tool

recommendation	string	Advice to the student using after using the self-assessment tool
status	bool	State of the self-assessment tool whether it has been finished by student or not (true or false)

5.2.8 **Entity: <Module>**

Attribute Name	**Type**	**Description**
moduleID {PK}	string	Unique identity of an interactive module
result	string	The outcome of studying the interactive module
status	bool	State of the interactive module whether it has been finished by student or not (true or false)

5.2.9 **Entity: <Chatbot>**

Attribute Name	**Type**	**Description**
botID {PK}	string	Unique identity of current chatbot.
botName	string	Name of current chatbot.
botVersion	string	Version of current chatbot.

5.2.10 **Entity: <PeerSupport>**

Attribute Name	**Type**	**Description**
supportSessionID {PK}	string	Unique identity of current peer support network.
studID{FK}	string	Name provided by the user during registration.

isAnonymous	bool	State of anonymity of student, whether they decide to be anonymous or not. (true or false)

5.2.11 **Entity: <Analytics>**

Attribute Name	Type	Description
metrics	List<String>	Stores system performance metrics and user engagement data.

5.2.12 **Entity: <Notification>**

Attribute Name	Type	Description
logID {PK}	int	Unique identifier for each notification log.
status	string	status of the notification

5.2.13 **Entity: <Insights>**

Attribute Name	Type	Description
InsightID	int	Unique identifier for each insight.
recommendation	string	Generated recommendation based on analyzed data.

6 User Interface Design

6.1 Overview of User Interface

The **Serene System** is designed to provide a seamless experience for mental health professionals, administrators, and students. Here's how users interact with the system to access its core features:

6.2 Screen Images

6.2.1 Interface for <Mental Health Professional login page>

Figure 6.1: Interface for <Mental Health Professional login page>

6.2.2 Interface for <Mental Health Professional functions page>

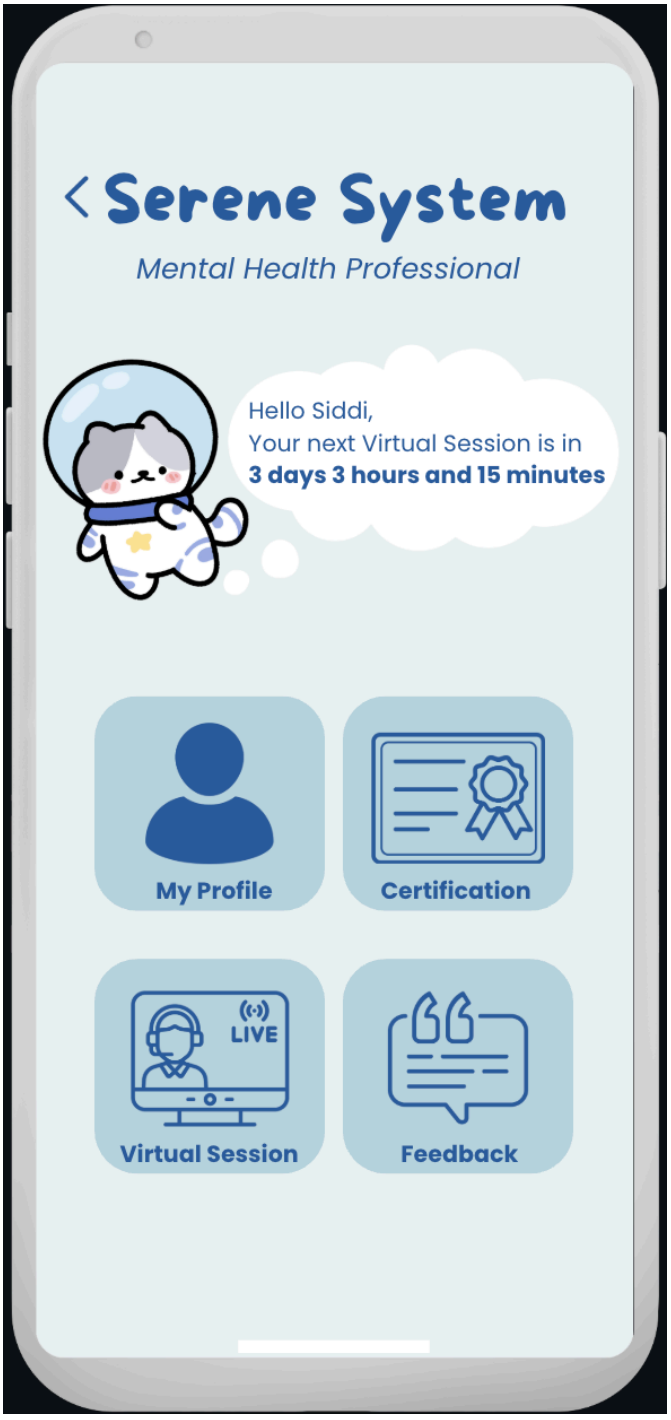


Figure 6.2: Interface for <Mental Health Professional functions page>

6.2.3 **Interface for <Mental Health Professional profile>**

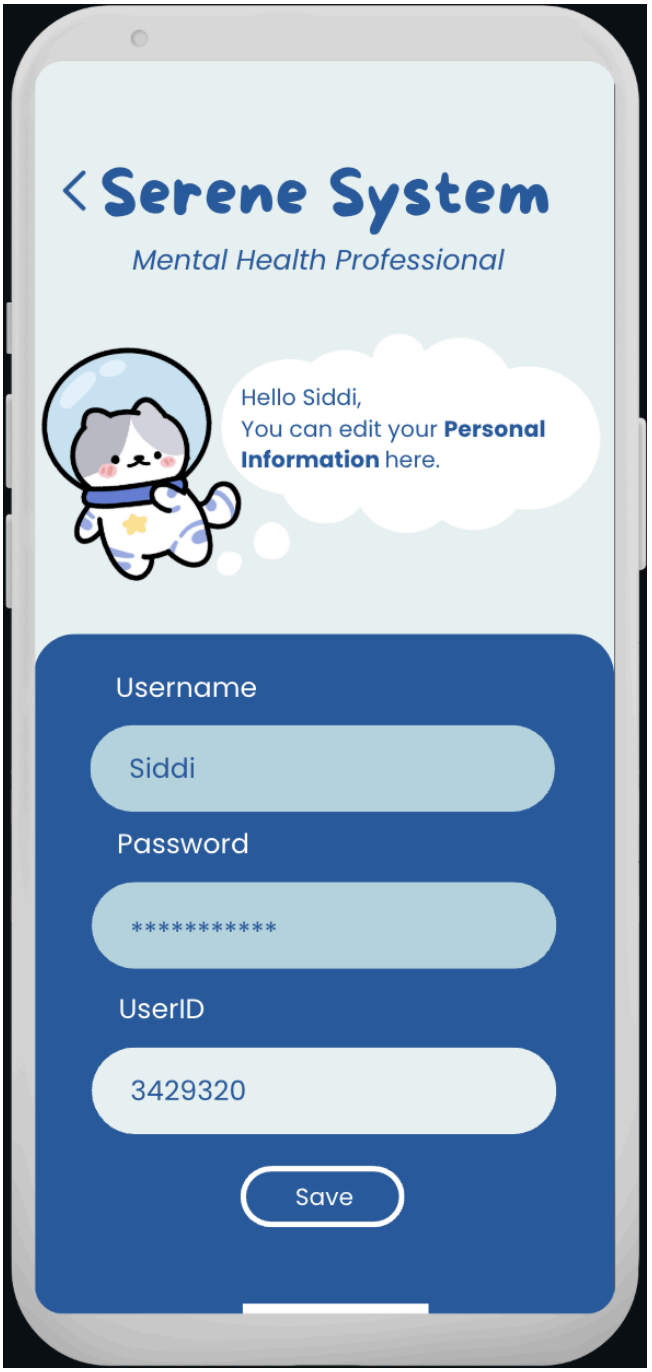


Figure 6.3: Interface for <Mental Health Professional profile>

6.2.4 Interface for <Mental Health Professional Certification>

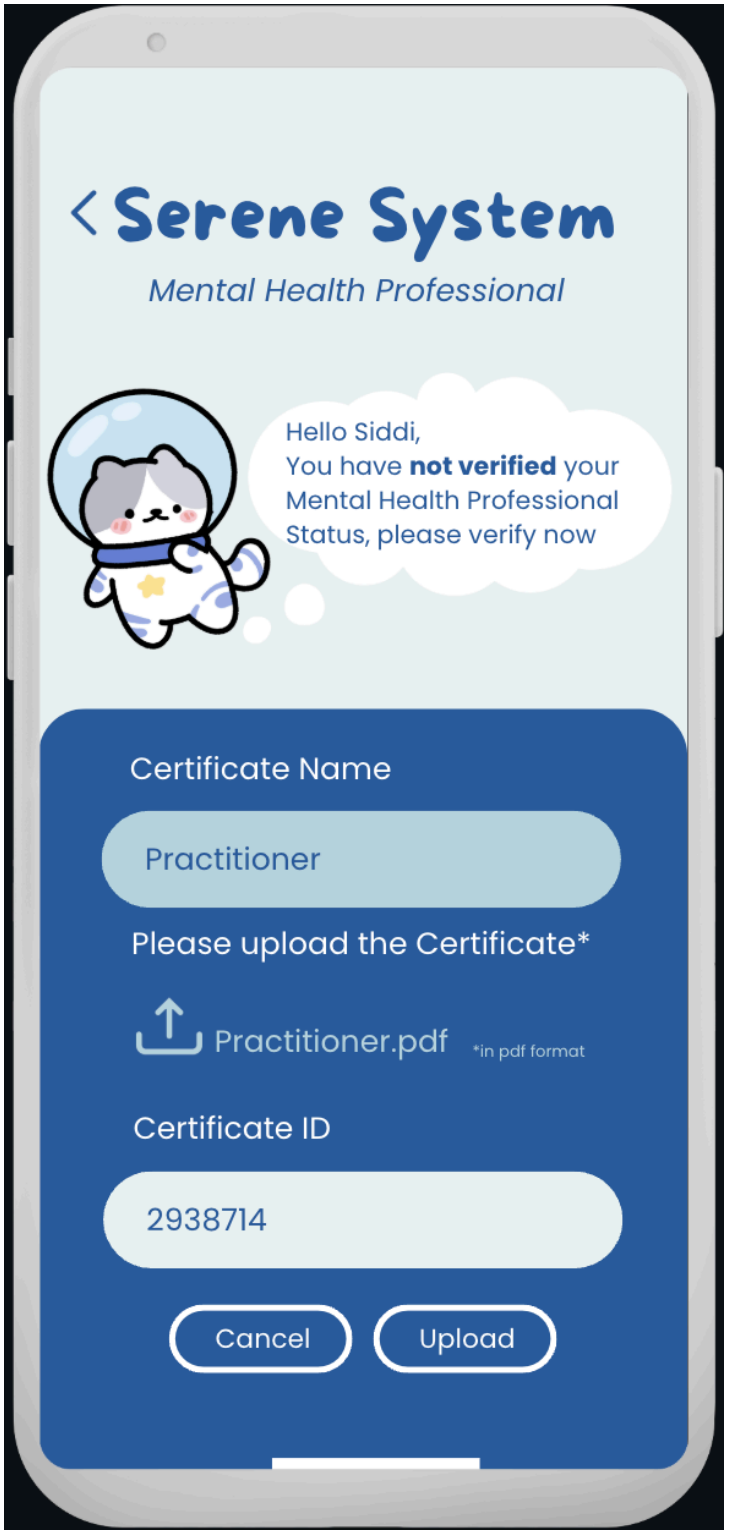


Figure 6.4: Interface for <Mental Health Professional Certification>

6.2.5 Interface for <Mental Health Professional Virtual Session>

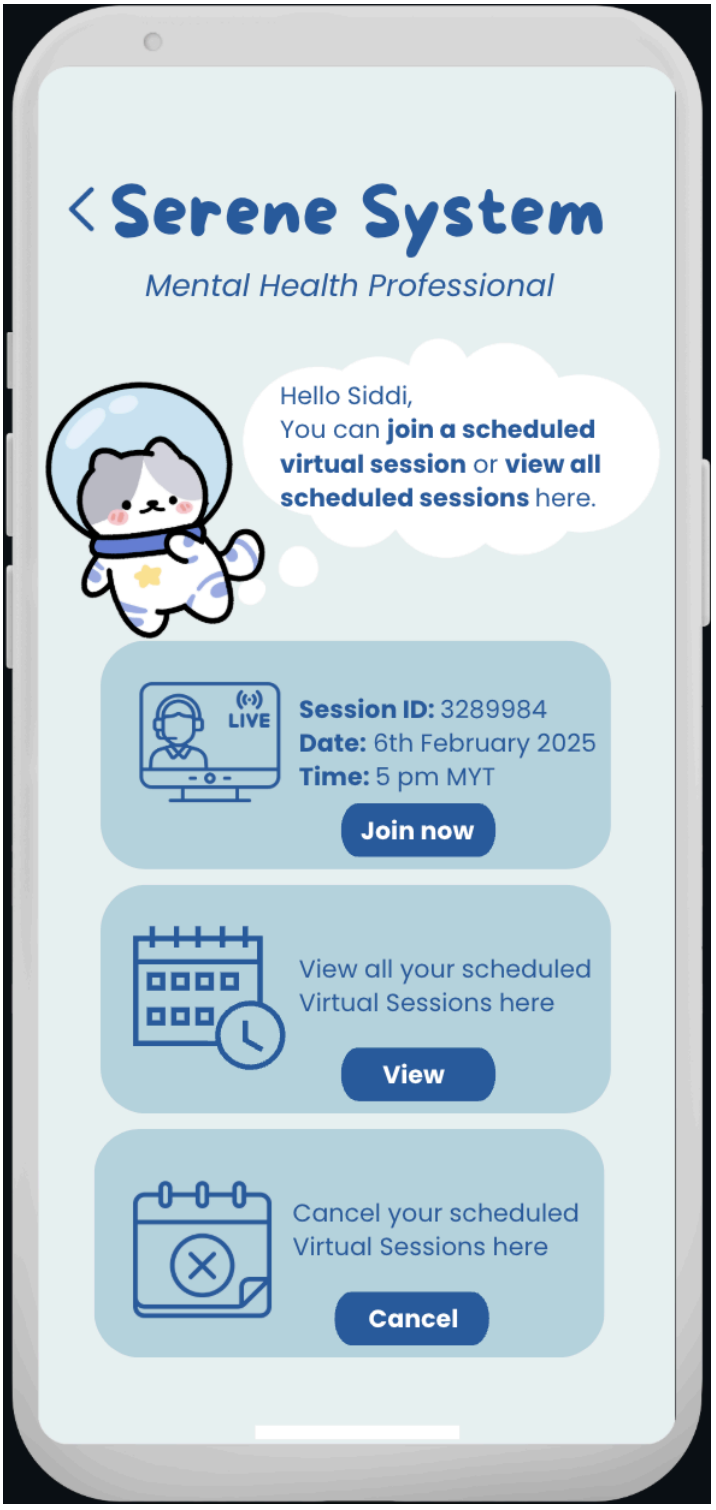


Figure 6.5: Interface for <Mental Health Professional Virtual Session>

6.2.6 **Interface for <Mental Health Professional Feedback>**

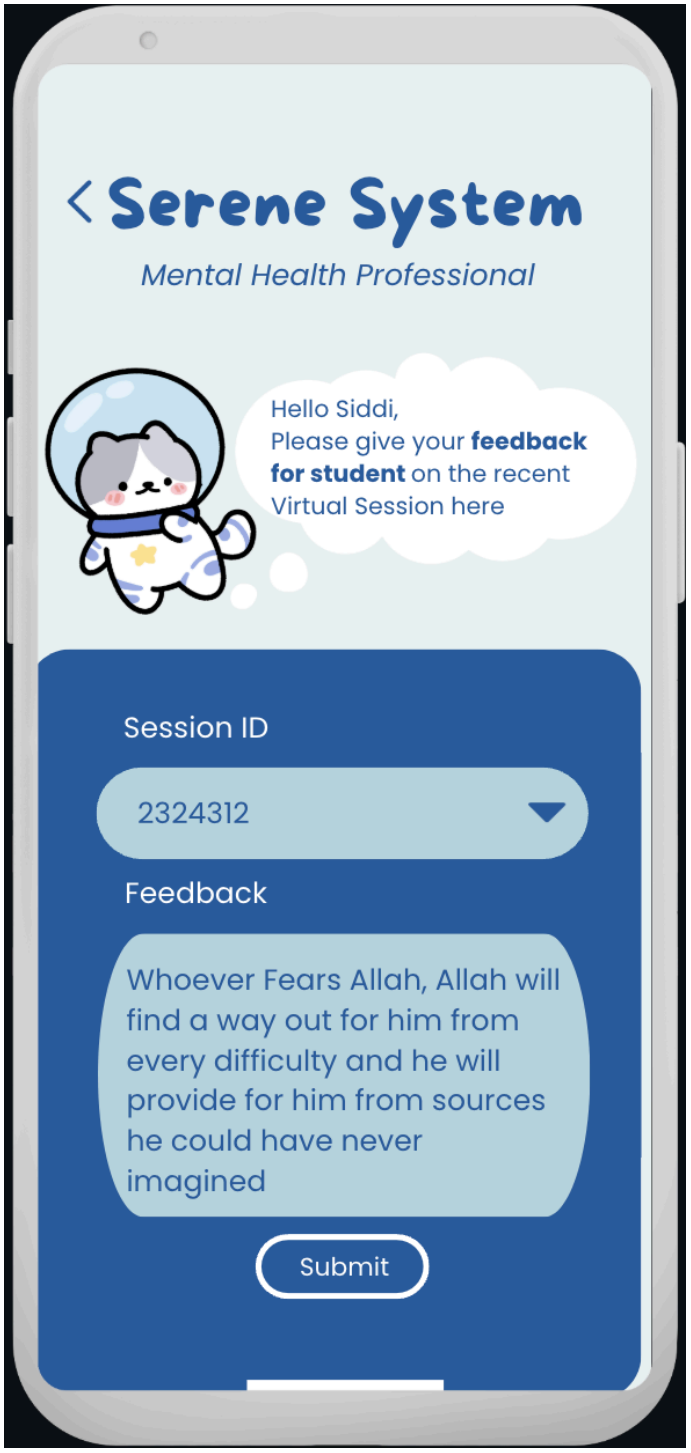


Figure 6.6: Interface for <Mental Health Professional Feedback>

6.2.7 User Registration

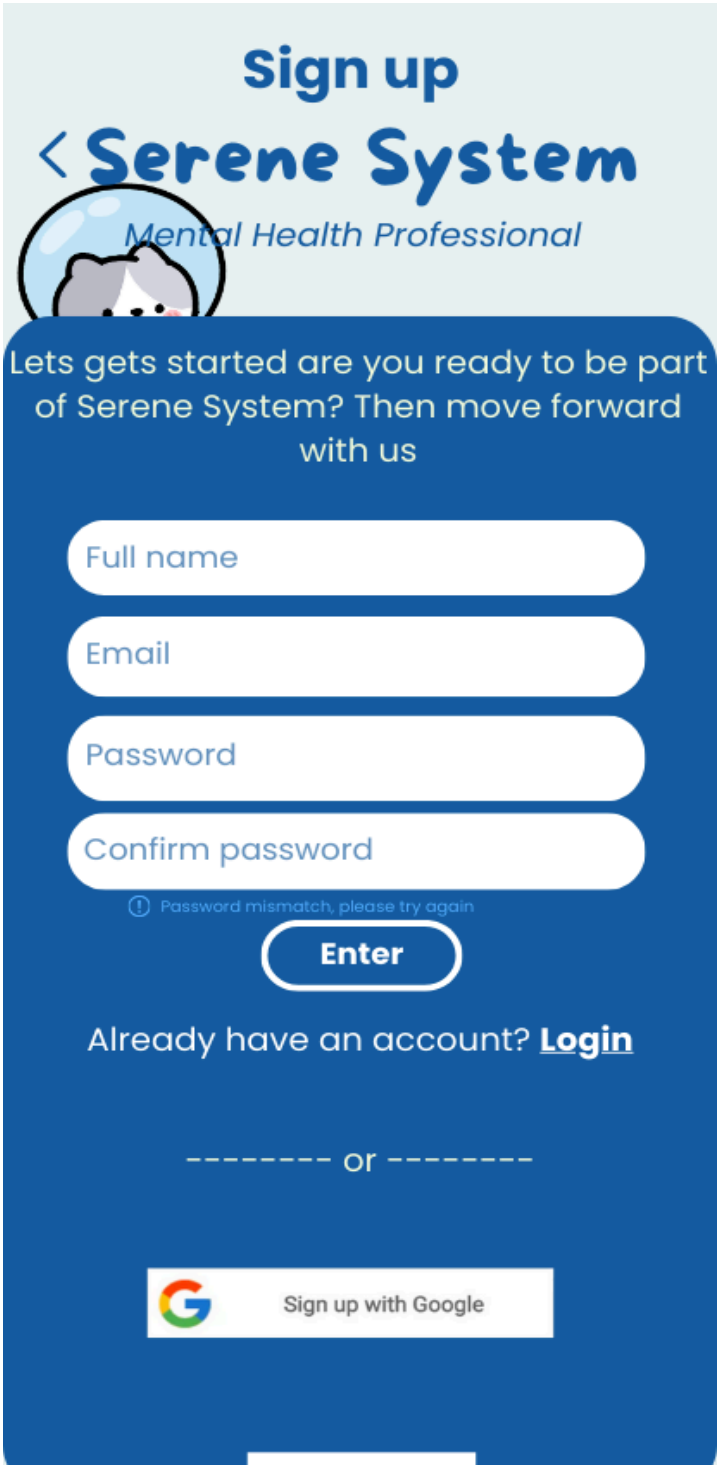


Figure 6.7:Interface for <User Registration>

6.2.8 User Authentication

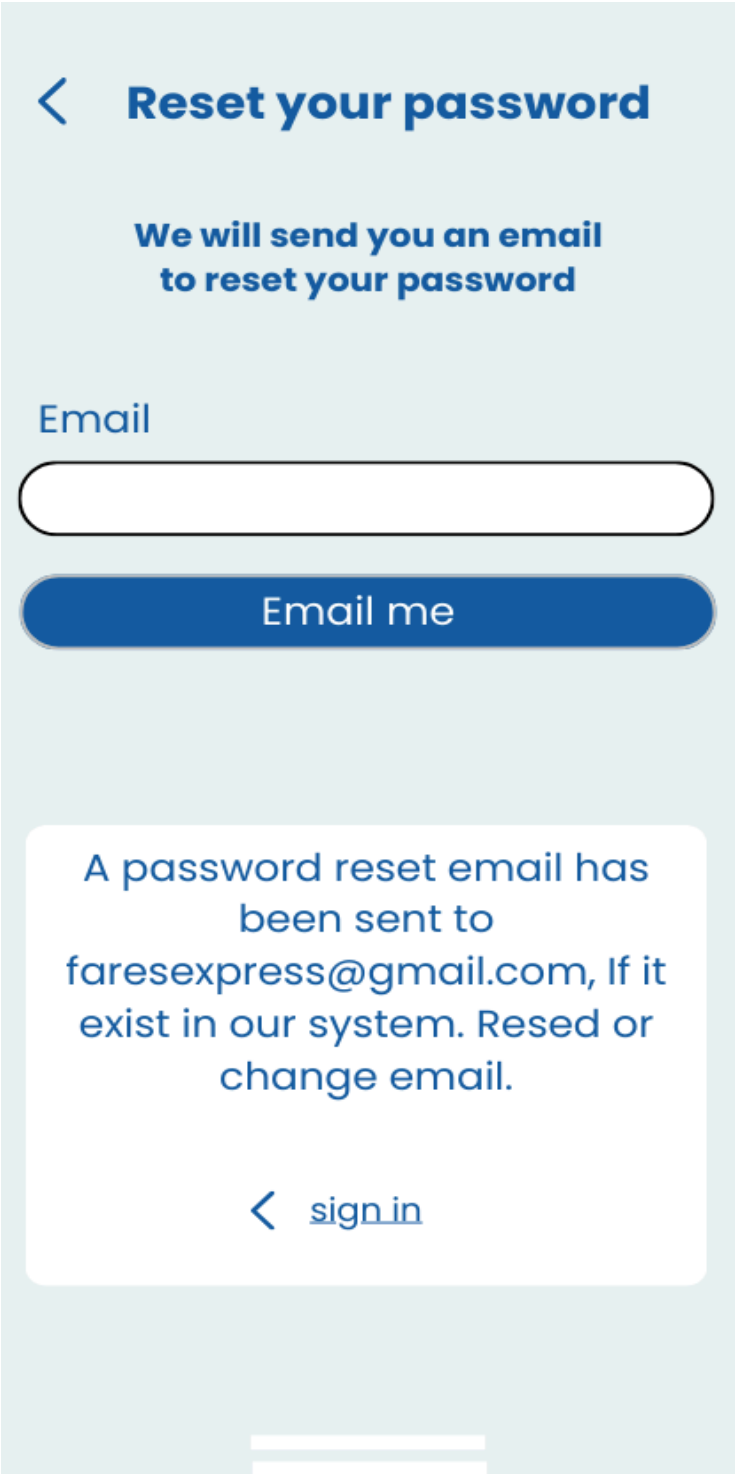


Figure 6.8: Interface for <Reset Password>

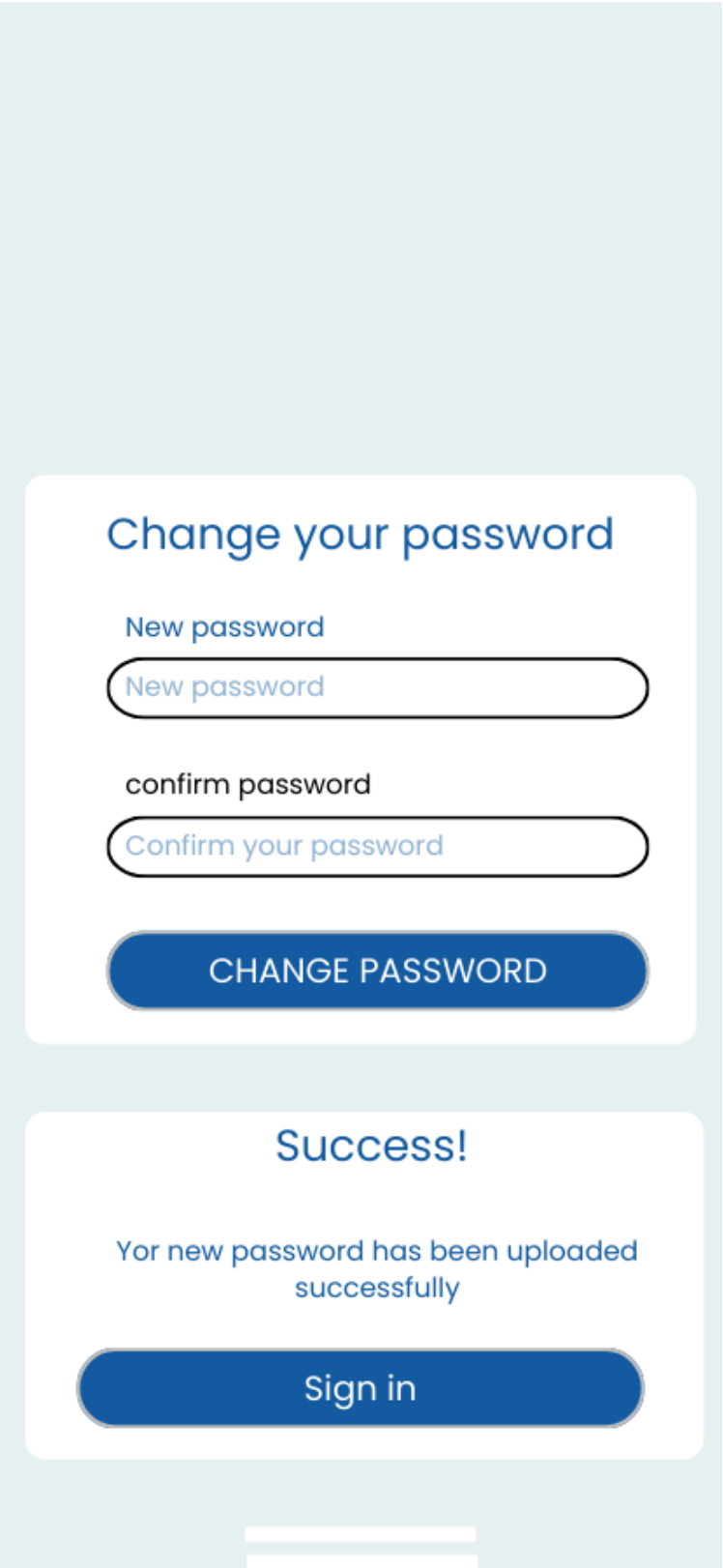


Figure 6.9: Interface for <Update or Renew The Password>

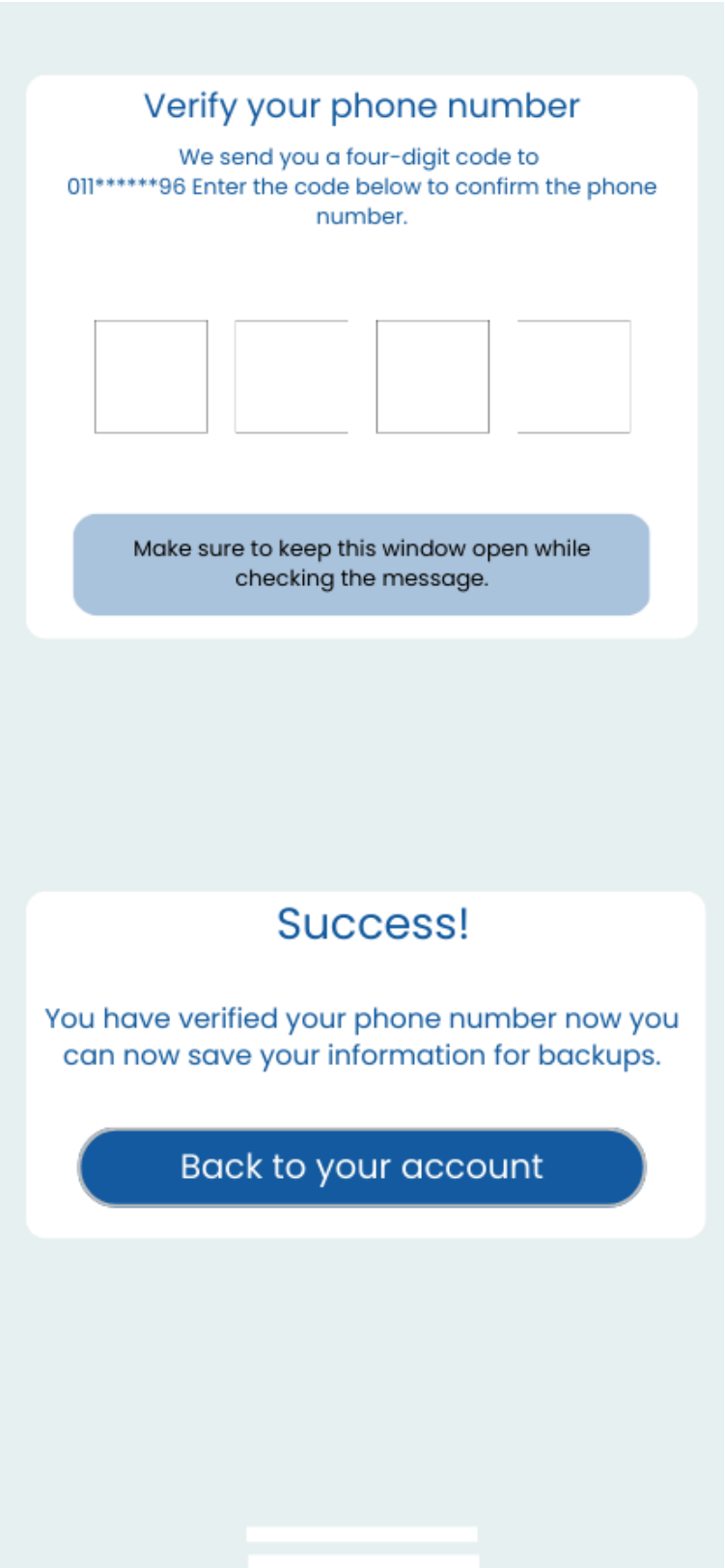


Figure 6.10: Interface for <Verifying Phone number>

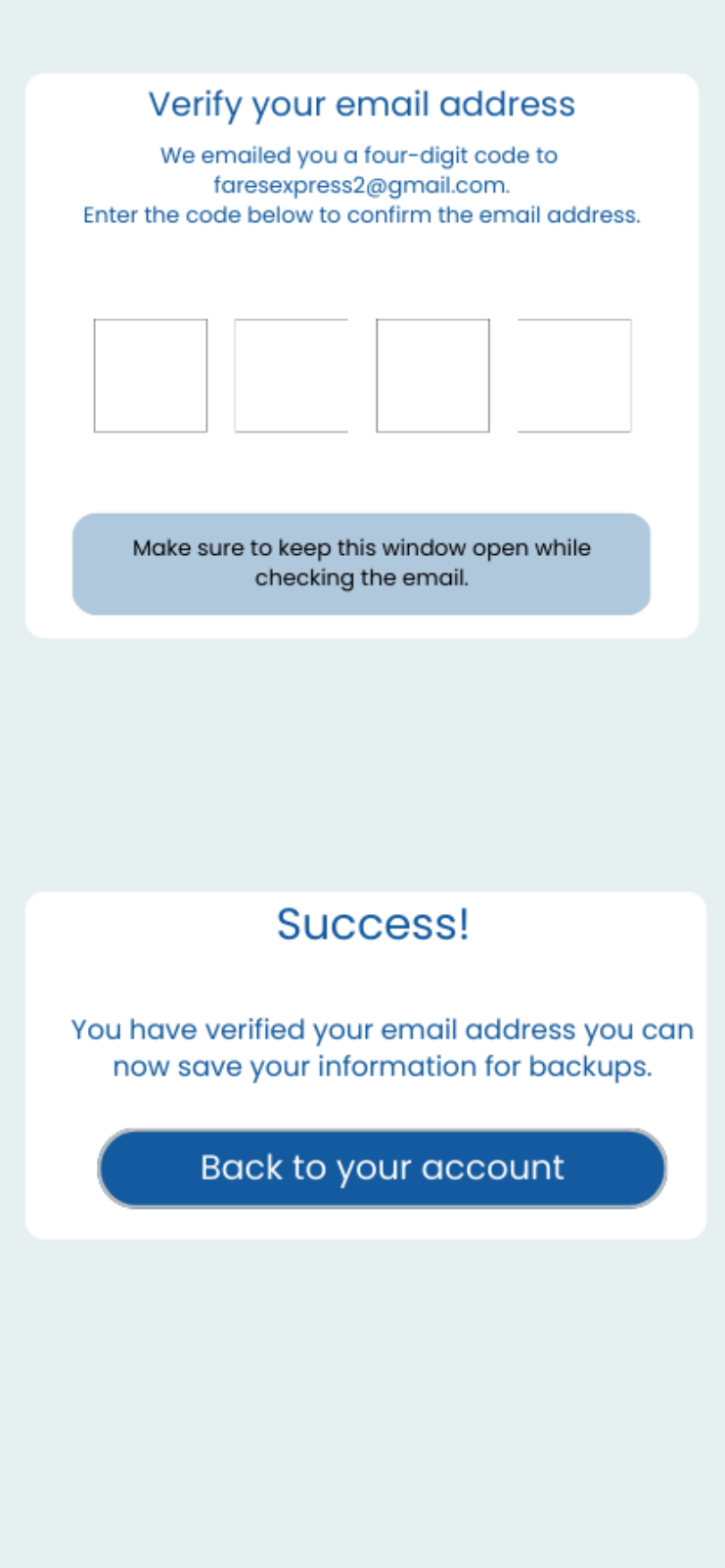


Figure 6.11:Interface for <Verifying Email Address>

6.2.9 Interface for <User Functions Page>

Figure 6.12: Interface for <User Functions Page>

6.2.10 Interface for <Assessment>

Figure 6.13: Interface for <Assessment>

6.2.11 Interface for <Virtual Sessions>

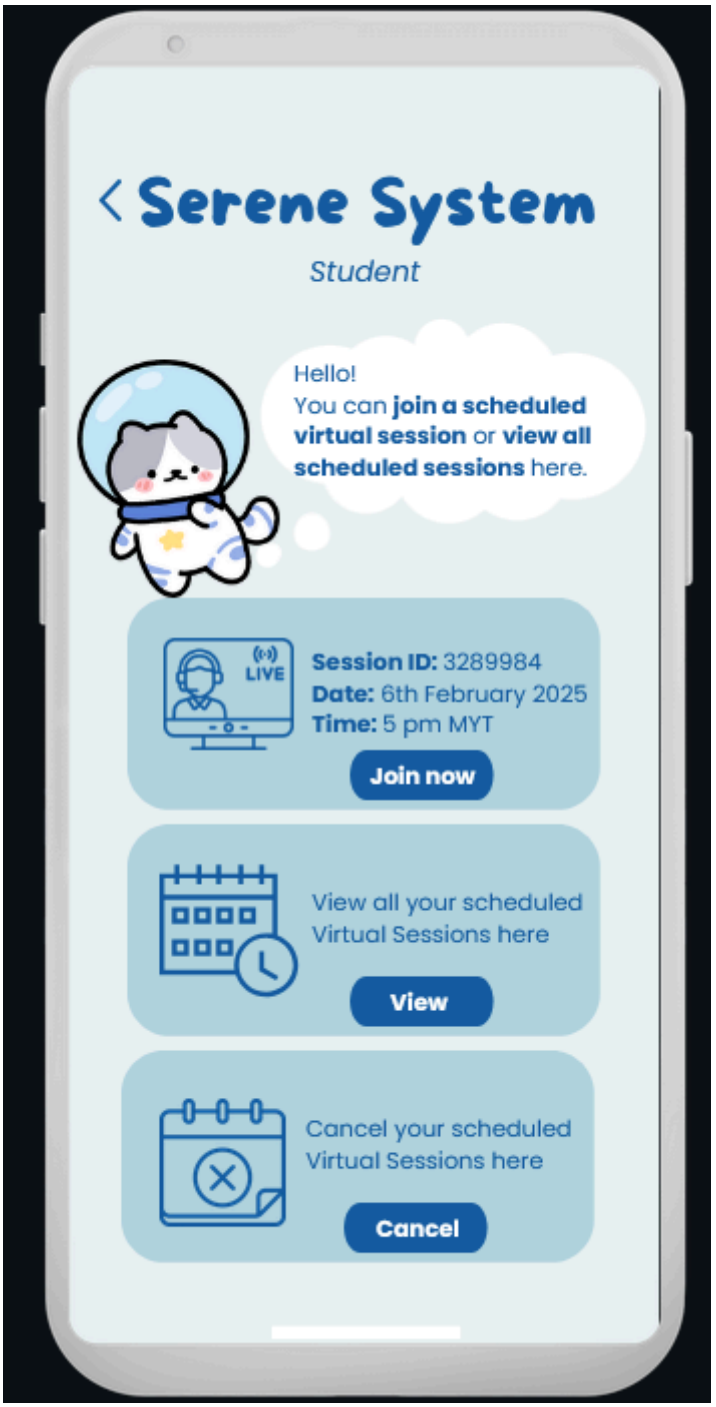


Figure 6.14: Interface for <Virtual Sessions>

6.2.12 Interface for <Module>

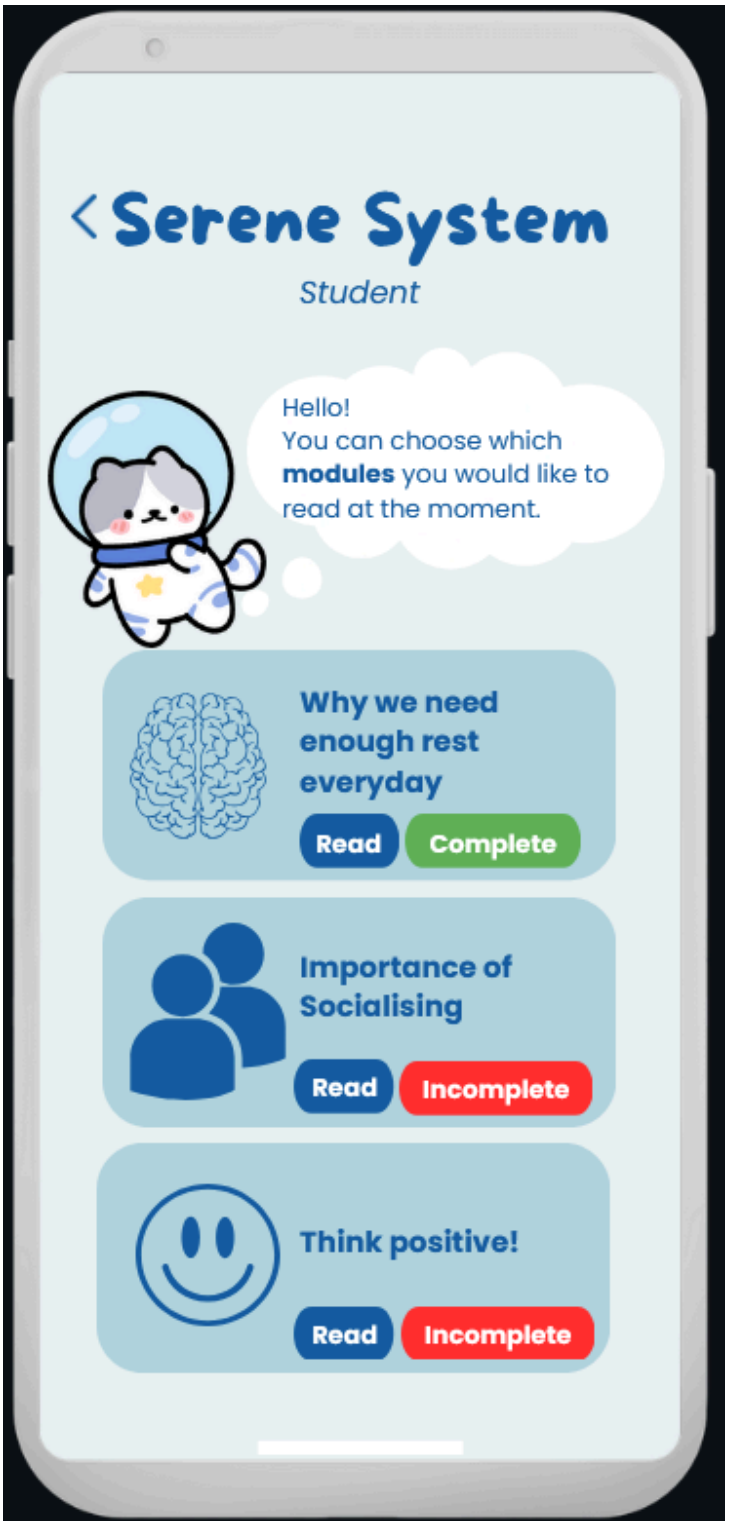


Figure 6.15: Interface for <Module>

6.2.13 Interface for <Chatbot>

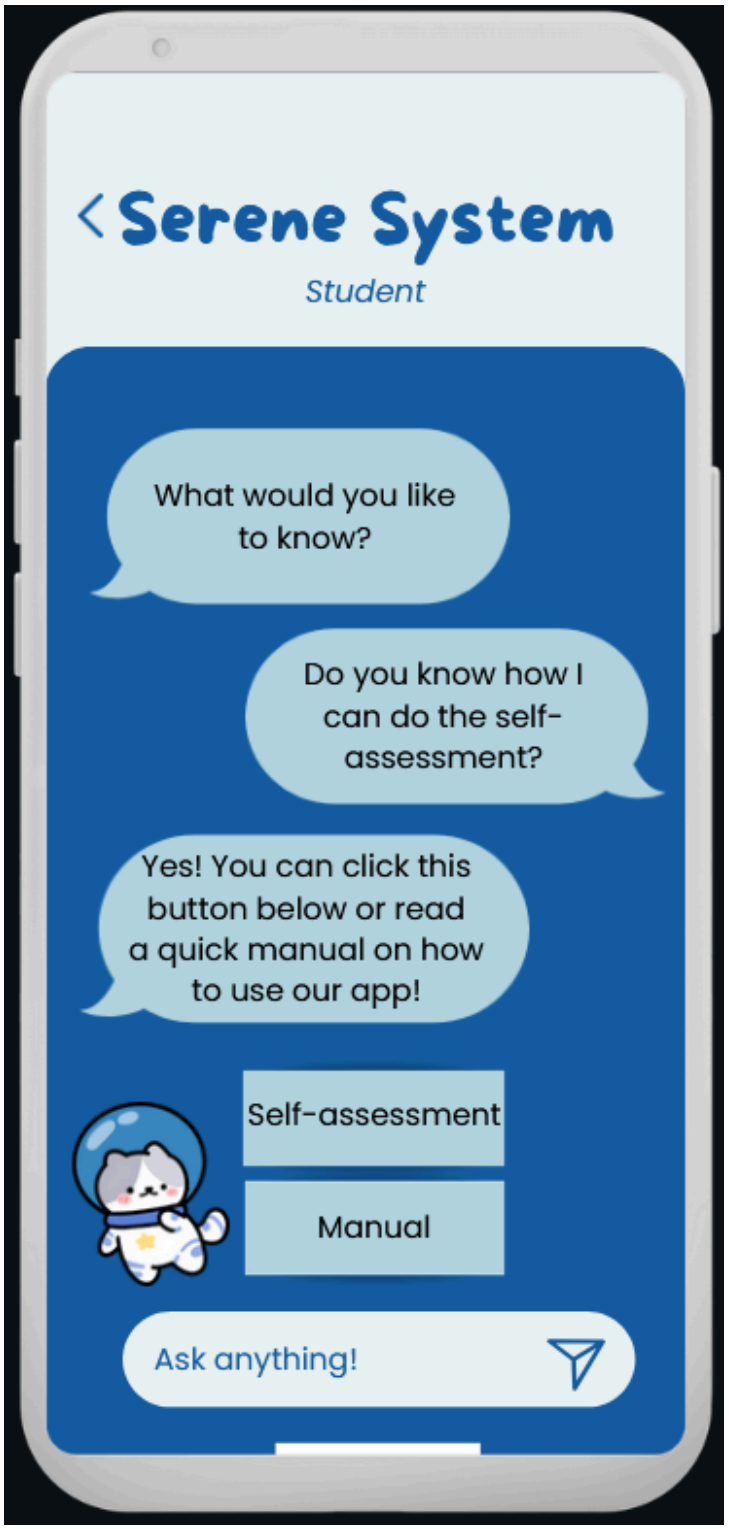
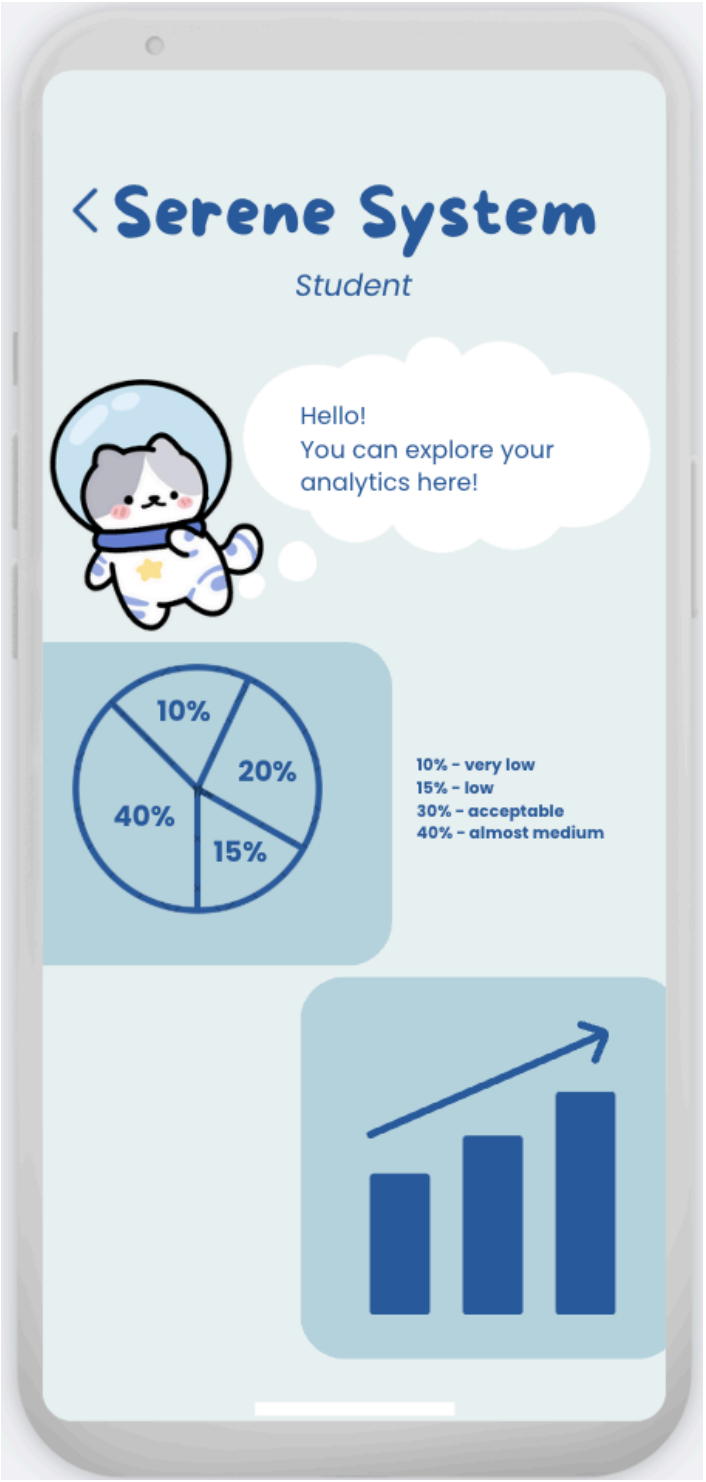


Figure 6.16: Interface for <Chatbot>

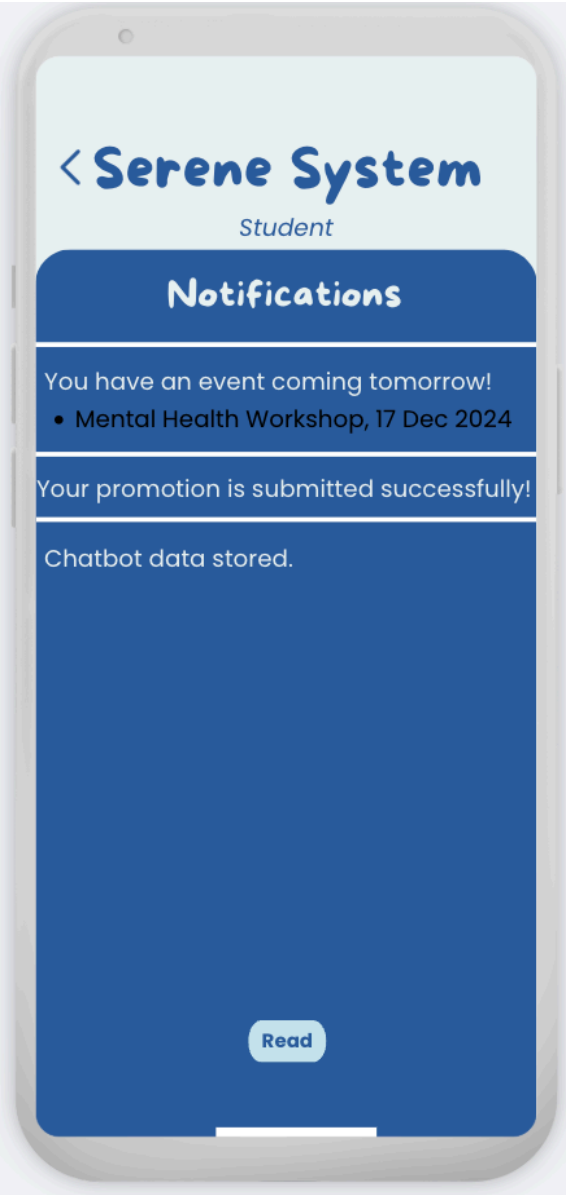
6.2.14 Interface for <PeerSupport>

Figure 6.17: Interface for <Peer Support>

6.2.15 Interface for <Analytics>



6.2.16 Interface for <Notifications>


Serene System
Student
Notifications
You have an event coming tomorrow!
Mental Health Workshop, 17 Dec 2024
Your promotion is submitted successfully!
Chatbot data stored.
Read

6.2.17 Interface for <Insights>

7 Requirements Matrix

The sequence diagrams for each use case vs. corresponding classes (entities) are listed as in Table 7.1.

Table 7.1: Description of Entities in the Database

	User Registration	User Authentication	Learning Modules	Self-Assessment Tools	Chatbot Support	Peer Support Network	Certificate Verification	Instant Feedback	Session Management	Analytics Dashboard	Awareness Tools
P001, UC007, SD001, SD002, SD003	X	X									
P002, UC009, SD004				X							
P002, UC008, SD005			X								
P003, UC005, SD006					X						
P003, UC011, SD007						X					
P003, UC003, SD008									X		
P004, UC001, SD009, SD010							X				

P004, UC002, SD011								X			
P004, UC003, SD012									X		
P004, UC003, SD013									X		
P004, UC003, SD014									X		
P005, UC004, SD015										X	X

8 Test Cases, Data and Expected Results

8.1 Test TC001 for <User Access Registration> : <Register To The System (UC007)>

This test contains the following test cases:

a) TC001_01 : Test <**Register To The System**(SD001 - NF)>
b) TC002_02 : Test <**Register To The System** (SD001 - AF1)>
c) TC003_03 : Test <**Register To The System** (SD001 - EF1)>

8.1.1 TC001_01 : Test <**Register To The System**(SD001 - NF1)>

Test Case ID:	TC001_01	Test Case Description	Successful registration
Created by:	Fares	Version:	
No.	Prerequisites	No	Test Data
1	User is on the registration page	1	username = "new_user"
2	User provides all required information correctly	2	email = "user@mail.com"
Test Condition			
Step #	Step Details	Expected Result	
1	Navigate to the registration page	Registration page is displayed successfully.	
2	Fill out the registration form with valid details	Form accepts input and displays no	

	(username, email, password, etc.)	errors.
3	Submit the registration form	System verifies input, creates account, displays success message, and redirects to login page

8.1.2 TC001_02 : Test <**Register To The System** (SD001# - AF1)>

Test Case ID:	TC001_02	Test Case Description	Duplicate username or email encountered during registration
Created by:	Fares	Version:	1.0
No.	Prerequisites	No	Test Data
1	User is on the registration page	1	username = "existingUser"
2	User provides a registered email	2	email = "existing@mail.com"
Test Condition			
Step #	Step Details	Expected Result	
1	Navigate to the registration page	Registration page is displayed successfully.	
2	Fill out the registration form with a duplicate username or email	System identifies duplicate and displays error: "Username/email already exists."	

3	Provide a unique username and email, then resubmit	Registration is processed successfully.

8.1.3 TC001_03 : Test <**Register To The System** (SD001# - EF)>

Test Case ID:	TC001_03	Test Case Description	Server issue encountered during registration
Created by:	Fares	Version:	1.0
No.	Prerequisites	No	Test Data
1	User is on the registration page	1	username = "user1"
2	Server connection is unstable	2	email = "user@mail.com"
Test Condition			
Step #	Step Details	Expected Result	
1	Navigate to the registration page	Registration page is displayed successfully.	
2	Fill out the registration form and submit	System encounters a server issue and displays error: "Server is unavailable. Please try again later."	
3	Retry registration after the server is back online	System processes registration successfully.	

8.2 Test TC002 for <Learning and Self Assessment> : <Study Interactive Mental Health Modules (UC008)>

This test contains the following test cases:

a) TC002_01 : Test <Study Interactive Mental Health Modules (SD005 - NF)>
b) TC002_02 : Test <Study Interactive Mental Health Modules (SD005 - Alt1)>
c) TC002_03 : Test <Study Interactive Mental Health Modules (SD005 - NF)>
d) TC002_04 : Test <Study Interactive Mental Health Modules (SD005 - Alt2)>

8.2.1 TC002_01: Test <Study Interactive Mental Health Modules(SD005 - NF)>

Test Case ID:	TC002_01	Test Case Description	Requesting a module from the system
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	moduleID = “M001”
2	Navigates to Modules UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Select the module from UI	Successful to fetch the module from data.	
2	Access and study the module	The module can be read and accessed.	

8.2.2 TC002_02: Test <Study Interactive Mental Health Modules(SD005 - Alt1)>

Test Case ID:	TC002_02	Test Case Description	Requesting a module from the system
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	moduleID = “M999”
2	Navigates to Modules UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Select the module from UI	Unsuccessful to fetch the module from data.	
2	Access and study the module	Given an error that module does not exist.	

8.2.3 TC002_03: Test <Study Interactive Mental Health Modules(SD005 - NF)>

Test Case ID:	TC002_03	Test Case Description	Updating status of current module
Created by:	Ivor	Version:	1.0

No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	moduleID = “M001”
2	Currently using a module		
Test Condition			
Step #	Step Details	Expected Result	
1	Finish studying current module	Check if the module has been finished, marking it as finished.	
2	Leave the module	Receive notification that module is complete.	

8.2.4 TC002_04: Test <Study Interactive Mental Health Modules(SD005 - Alt2)>

Test Case ID:	TC002_04	Test Case Description	Updating status of current module
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	moduleID = “M001”
2	Currently using a module		
Test Condition			
Step #	Step Details	Expected Result	
1	Attempt to leave midway of current module	Check if the module has been finished, marking it as unfinished.	
2	Successfully Leave the module	Receive notification that module is incomplete.	

8.3 Test TC003 for <Learning and Self Assessment> : <Evaluate with Self-Assessment tools (UC009)>

This test contains the following test cases:

a) TC003_01 : Test <Evaluate with Self-Assessment tools (SD004 - NF)>
b) TC003_02 : Test <Evaluate with Self-Assessment tools (SD004 - Alt1)>
c) TC003_03 : Test <Evaluate with Self-Assessment tools (SD004 - NF)>
d) TC003_04 : Test <Evaluate with Self-Assessment tools (SD004 - Alt2)>
e) TC003_05 : Test <Evaluate with Self-Assessment tools (SD004 - NF)>
f) TC003_06 : Test <Evaluate with Self-Assessment tools (SD004 - Alt3)>

8.3.1 TC003_01: Test <Evaluate with Self-Assessment tools(SD004 - NF)>

Test Case ID:	TC003_01	Test Case Description	Requesting self-assessment tool
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	assessmentID = “SA001”
2	Navigates to Self-Assessment UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Choose a self-assessment tool	Able to fetch the self-assessment tool from data	
2	Attempt to conduct the	Able to run the self-assessment tool.	

	chosen self-assessment tool	

8.3.2 TC003_02: Test <Evaluate with Self-Assessment tools(SD004 - Alt1)>

Test Case ID:	TC003_02	Test Case Description	Requesting self-assessment tool
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	assessmentID = “SA999”
2	Navigates to Self-Assessment UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Choose a self-assessment tool	Unable to fetch the self-assessment tool from data	
2	Attempt to conduct the chosen self-assessment tool	Unable to run the self-assessment tool and show an error message.	

8.3.3 TC003_03: Test <Evaluate with Self-Assessment tools(SD004 - NF)>

Test	TC003_03	Test Case	Updating status

Case ID:		Description	completion of current self-assessment tool
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	assessmentID = “SA001”
2	In a self-assessment tool UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Finish using self-assessment tool	Update status of the self-assessment tool for the current student to finished.	
2	Leave the current self-assessment tool	Receive notification that assessment is complete.	

8.3.4 TC003_04: Test <Evaluate with Self-Assessment tools(SD004 - Alt2)>

Test Case ID:	TC003_04	Test Case Description	Updating status completion of current self-assessment tool
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	assessmentID = “SA001”
2	In a self-assessment tool UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Attempt to leave self-assessment tool midway	Update status of the self-assessment tool for the current student to unfinished.	
2	Leave the current self-assessment tool	Receive notification that assessment is incomplete.	

8.3.5 TC003_05: Test <Evaluate with Self-Assessment tools(SD004 - NF)>

Test Case ID:	TC003_05	Test Case Description	Requesting self-assessment outcome
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	assessmentID = “SA001”
2	Current self-assessment tool is in finished status		
Test Condition			

Step #	Step Details	Expected Result
1	Request current self-assessment tool outcome	Successfully fetch the grade, result and recommendation of current self-assessment tool due to finished status.
2	Access the self-assessment tool outcome	Student able to access the grade, result and recommendation of current self-assessment tool.

8.3.6 TC003_06: Test <Evaluate with Self-Assessment tools(SD004 - Alt3)>

Test Case ID:	TC003_06	Test Case Description	Requesting self-assessment outcome
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	assessmentID = “SA001”
2	Current self-assessment tool is in unfinished status		
Test Condition			
Step #	Step Details	Expected Result	
1	Request current self-assessment tool outcome	Unsuccessfully fetch the grade, result and recommendation of current self-assessment tool due to unfinished status.	
2	Access the self-assessment tool outcome	Student receive an error message that the self-assessment tool needs to be finished first.	

8.4 Test TC004 for <Communication and Support> : <Interact with 24/7 Chatbot Support (UC005)>

This test contains the following test cases:

a) TC004_01: Test <Interact with 24/7 Chatbot Support(SD006 - NF)>
b) TC004_02: Test <Interact with 24/7 Chatbot Support(SD006 - EF1)>
c) TC004_03: Test <Interact with 24/7 Chatbot Support(SD006 - EF2)>

8.4.1 TC004_01: Test <Interact with 24/7 Chatbot Support(SD006 - NF)>

Test Case ID:	TC004_01	Test Case Description	
Created by:	Anjum	Version:	
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	botID= 99
2	Students navigate to chatbot UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Connect to a bot using a valid botID	Establish connection to the bot with respective botID	
2	Start the chat	The bot sends prompt indicating the chat has been started	
3	Give valid input for processing	The bot generates reply for the input and displays the generated input to the user	

4	User stops the chat	The chat stops and the connection is unestablished and the user can see the chat history

8.4.2 TC004_02: Test <Interact with 24/7 Chatbot Support(SD006 - EF1)>

Test Case ID:	TC004_01	Test Case Description	
Created by:	Anjum	Version:	
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	botID= 123$%
2	Students navigate to chatbot UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Connect to a bot using a invalid botID	Successful connection establishment message is displayed to the user	

8.4.3 TC004_03: Test <Interact with 24/7 Chatbot Support(SD006 - EF2)>

Test	TC004_03	Test Case	

Case ID:		Description	
Created by:	Anjum	Version:	
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	botID= 99
2	Students navigate to chatbot UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Connect to a bot using a valid botID	Establish connection to the bot with respective botID	
2	Start the chat	The bot sends prompt indicating the chat has been started	
3	Give invalid input for processing	Throw error and display invalid input error to user	

8.5 Test TC005 for <Communication and Support> : <Communicate through Peer Support Network (UC011)>

This test contains the following test cases:

a) TC005_01: Test <Communicate through Peer Support Network(SD007 - NF)>

b) TC005_02: Test <Communicate through Peer Support Network(SD007- Alt1)>
c) TC005_03: Test <Communicate through Peer Support Network(SD007 - NF)>
d) TC005_04: Test <Communicate through Peer Support Network(SD007- Alt2>
e) TC005_05: Test <Communicate through Peer Support Network(SD007 - NF)>
f) TC005_06: Test <Communicate through Peer Support Network(SD007 - NF)>

8.5.1 TC005_01: Test <Communicate Through Peer Support Network (SD007 - NF)>

Test Case ID:	TC005_01	Test Case Description	Establish connection to the Peer Support Network
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	supportSessionID = “PS001”
2.	Students navigates to Peer Support Network UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Request to connect to the peer support network	Successfully establish the connection to the peer support network	
2	Access the peer support network	Enable user to access the selected peer support network and start communicating	

8.5.2 TC005_02: Test <Communicate through Peer Support Network(SD007- Alt1)>

Test Case ID:	TC005_02	Test Case Description	Establish connection to the Peer Support Network
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	supportSessionID = “PS999”
2.	Students navigates to Peer Support Network UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Request to connect to the peer support network	Unsuccessfully establish the connection to the peer support network	
2	Access the peer support network	Return error message on failing to secure connection to the peer support network	

8.5.3 TC005_03: Test <Communicate through Peer Support Network(SD007 - NF)>

Test Case ID:	TC005_03	Test Case Description	Change identity in the peer support network
Created by:	Ivor	Version:	1.0

No.	Prerequisites	No	Test Data
1	Students have logged in to the system		
2.	Student currently connected in a peer support network		
Test Condition			
Step #	Step Details	Expected Result	
1	Request to be anonymous in the network	Change identity to be anonymous and return true.	
2	Start sending messages	Current identity correctly shows “Anonymous”	

8.5.4 TC005_04: Test <Communicate through Peer Support Network(SD007- Alt2>

Test Case ID:	TC005_04	Test Case Description	Change identity in the peer support network
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system		
2.	Student currently connected in a peer support network		
Test Condition			
Step #	Step Details	Expected Result	
1	Request to revert anonymous state in the network	Change back to real identity from anonymous and return false.	
2	Start sending messages	Current identity correctly shows the student's own personal name.	

8.5.5 TC005_05: Test <Communicate through Peer Support Network(SD007 - NF)>

Test Case ID:	TC005_05	Test Case Description	Sending messages in the network
Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	message = “Hi!”
2.	Student currently connected in a peer support network		
Test Condition			
Step #	Step Details	Expected Result	
1	Send message in the chat	Successfully sends the message and displayed in the peer support network	

8.5.6 TC005_06: Test <Communicate through Peer Support Network(SD007 - NF)>

Test Case ID:	TC005_06	Test Case Description	Receiving messages in the network

Created by:	Ivor	Version:	1.0
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	message = “Hello! How are you?”
2.	Student currently connected in a peer support network		
Test Condition			
Step #	Step Details	Expected Result	
1	Messages sent to the network	Successfully retrieve the message from data.	
2	Reading the network messages	Messages are displayed with details of the sender.	

8.6 Test TC006 for <Communication and Support> : <Schedule Virtual Counselling Session (UC003)>

This test contains the following test cases:

a) TC006_01: Test <Schedule Virtual Counselling Session(SD008 - NF)>
b) TC006_02: Test <Schedule Virtual Counselling Session(SD008 - AF)>

8.6.1 TC006_01: Test <Schedule Virtual Counselling Session(SD008 - NF)>

Test Case ID:	TC006_01	Test Case Description	
Created by:	Anjum	Version:	
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	DateTime : '22-01-2025' / '08:15:00'
2	Students navigate to Virtual session UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Students clicks on the schedule virtual session	UI to schedule session should show	
2	Choose available date	The availability of the date should be displayed	
3	Choose	The availability of the time slot should be	

	available time	displayed
4	Student clicks schedule icon	The session is scheduled successfully and the date and time should be blocked preventing any duplicate scheduling.

8.6.2 TC006_02: Test <Schedule Virtual Counselling Session(SD008 - AF1)>

Test Case ID:	TC006_01	Test Case Description	
Created by:	Anjum	Version:	
No.	Prerequisites	No	Test Data
1	Students have logged in to the system	1	DateTime : '22-01-2025' / '08:15:00'
2	Students navigate to Virtual session UI		
Test Condition			
Step #	Step Details	Expected Result	
1	Students clicks on the schedule virtual session	UI to schedule session should show	
2	Choose unavailable date	The unavailability of the date should be displayed	
3	Choose unavailable time	The unavailability of the time slot should be displayed	
4	Student clicks	Prompt the student to select available date	

	schedule icon	and time

8.7 Test TC007 for <Professional Management System> : <Verify Professional Certification (UC001)>

This test contains the following test cases:

a) TC007_01 : Test <Verify Professional Certification (SD010 - NF)>
b) TC007_02 : Test <Verify Professional Certification (SD010 - AF1)>
c) TC007_03 : Test <Verify Professional Certification (SD010 - AF2)>
d) TC007_04 : Test <Verify Professional Certification (SD010 - AF3)>

8.7.1 TC007_01 : Test <Verify Professional Certification (SD0010 - NF)>

Test Case ID:	TC007_01	Test Case Description	
Created by:	Anjum	Version:	
No.	Prerequisites	No	Test Data
1	Mental health professional login into their account	1	DocumentPath = “Android\Documents\Certificate.pdf”
2	Navigates to Certification page		
Test Condition			
Step #	Step Details	Expected Result	
1	Select the document from	The file path should be correct and correct file is fetched	

	local system	
2	Click on upload	The fetched file is successfully uploaded to the system

8.7.2 TC007_02 : Test <Verify Professional Certification (SD010 - AF1)>

Test Case ID:	TC007_02	Test Case Description	
Created by:	Anjum	Version:	
No.	Prerequisites	No	Test Data
1	Mental health professional login into their account	1	DocumentPath = “Android\Documents\Certificate.pdf”
2	Navigates to Certification page		
Test Condition			
Step #	Step Details	Expected Result	
1	Select the document from local system	The file path should be correct and correct file is fetched	
2	Error on upload	update the CStatus as Error Uploading	

8.7.3 TC007_03 : Test <Verify Professional Certification (SD010 - AF2)>

Test Case ID:	TC007_03	Test Case Description	
Created by:	Anjum	Version:	
No.	Prerequisites	No	Test Data
1	Mental health professional login into their account	1	DocumentPath = “Android\Documents\Certificate.pdf”
2	Navigates to Certification page		
Test Condition			
Step #	Step Details	Expected Result	
1	Select the document from local system	The file path should be correct and correct file is fetched	
2	Cancel upload	The uploaded file is removed from memory and no draft is saved and exits from page	

8.7.4 TC007_04 : Test <Verify Professional Certification (SD010 - AF3)>

Test	TC004_04	Test Case	

Case ID:		Description	
Created by:	Anjum	Version:	
No.	Prerequisites	No	Test Data
1	Mental health professional login into their account	1	DocumentPath = “Android\Documents\Certificate.pdf”
2	Navigates to Certification page		
Test Condition			
Step #	Step Details	Expected Result	
1	Select document from local system with incorrect format	The file is not fetched and “Incorrect format of file” will be displayed and asks user to upload again	

8.8 Test TC008 for <Professional Management System> : <Provide Instant Feedback(UC002)>

This test contains the following test cases:

d) TC008_01 : Test <Provide Instant Feedback(SD011 - NF)>
e) TC008_02 : Test <Provide Instant Feedback (SD011 - AF1)>
f) TC008_03 : Test <Provide Instant Feedback (SD011 - EF1)>

8.8.1 TC008_01 : Test <Provide Instant Feedback (SD011 - NF)>

Test Case ID:	TC008_01	Test Case Description	
Created by:	Anjum	Version:	
No.	Prerequisites	No	Test Data
1	Mental health professional login into their account	1	Session_ID = 2323421
2	Navigates to Feedback page	2	Comments = “I recommend that you…”
Test Condition			
Step #	Step Details	Expected Result	
1	Click on the SessionID	The system should show a dropdown of Sessions that was completed but not yet given feedback	
2	Enter Comments	The Comments should be entered as per user input	
3	Click on submit	The user should be able to get a “Feedback Submitted successfully” message.	

8.8.2 TC008_02 : Test <Provide Instant Feedback (SD011 - AF1)>

Test Case ID:	TC008_02	Test Case Description	
Created by:	Anjum	Version:	

No.	Prerequisites	No	Test Data
1	Mental health professional login into their account	1	Session_ID = 2323421
2	Navigates to Feedback page	2	Comments = “ “
Test Condition			
Step #	Step Details	Expected Result	
1	Click on the SessionID	The system should show a dropdown of Sessions that was completed but not yet given feedback	
2	No Comments Entered	No comment is shown on the field	
3	Click on submit	The user should be able to get a “Invalid! fields “ message and prompt user to enter again	

8.8.3 TC008_03 : Test <Provide Instant Feedback (SD011 - AF1)>

Test Case ID:	TC008_03	Test Case Description	
Created by:	Anjum	Version:	

No.	Prerequisites	No	Test Data
1	Mental health professional login into their account	1	Session_ID = 2323421
2	Navigates to Feedback page	2	Comments = ““I recommend that you…”
Test Condition			
Step #	Step Details	Expected Result	
1	Click on the SessionID	The system should show a dropdown of Sessions that was completed but not yet given feedback	
2	Enter Comments	The Comments should be entered as per user input	
3	Click on submit	The system throws an error due to some internal issues. The error should be handled properly and it should throw a message “ Error Submitting Feedback”	

8.9.1 Test TC009 for System Monitoring: <Review Analytics Dashboard (UC004)>

This test contains the following test cases:

a) TC009_01 : Test <**Review Analytics Dashboard** (SD015 - NF)>
b) TC009_02 : Test <**Review Analytics Dashboard** (SD015 - Alt1)>

a) TC005_01 : Test <**Review Analytics Dashboard** (SD015 - NF)>

Test Case ID:	TC005_01	Test Case Description	Test <**Review Analytics Dashboard**> (SD015 - NF)>
Created by:	Sheyla	Version:	1.0
No.	Prerequisites	No	Test Data
1	User is logged in and the server is stable	1	username = “user1”
			email = “user@gmail.com”
Test Condition	User sends a request to view analytics dashboard		
Step #	Step Details	Expected Result	
1	Navigate to the analytics dashboard.	Dashboard page is displayed successfully.	
2	Request metrics to display on the dashboard	Metrics are retrieved and displayed visually.	

b) TC005_02 : Test <**Review Analytics Dashboard**> (SD015 - Alt1)>

Test Case ID:	TC005_02	Test Case Description	Test <**Review Analytics Dashboard**> (SD015 - Alt1)>
Created	Sheyla	Version:	1.0

by:			
No.	Prerequisites	No	Test Data
1	Server connection is unstable.	1	username = “user1”
			email = “user@gmail.com”
Test Condition	User attempts to access the analytics dashboard during unstable server conditions.		
Step #	Step Details	Expected Result	
1	Navigate to the analytics dashboard page.	Dashboard page fails to load.	
2	Retry loading the page after resolving the issue.	Dashboard is displayed successfully.	

8.10.1 Test TC010 for System Monitoring: <Send Notification (UC004)>

This test contains the following test cases:

a) TC010_01 : Test <**Send Notification** (SD004 - NF)>
b) TC010_02 : Test <**Review Analytics Dashboard** (SD004 - Alt1)>

a)TC010_01 : Test <**Send Notification** (SD004 - NF)>

Test Case ID:	TC010_01	Test Case Description	Test <**Send Notification** (SD004 - NF)>
Created by:	Sheyla	Version:	1.0
No.	Prerequisites	No	Test Data
1	Notification settings are enabled.	1	logID = “101”
			status = “Pending”
Test Condition	User activate a notification for a specific event.		
Step #	Step Details	Expected Result	
1	Navigate to the notification settings page.	Notification settings are displayed.	
2	Activate a notification for wanted events.	Notification is sent and appears in the UI.	

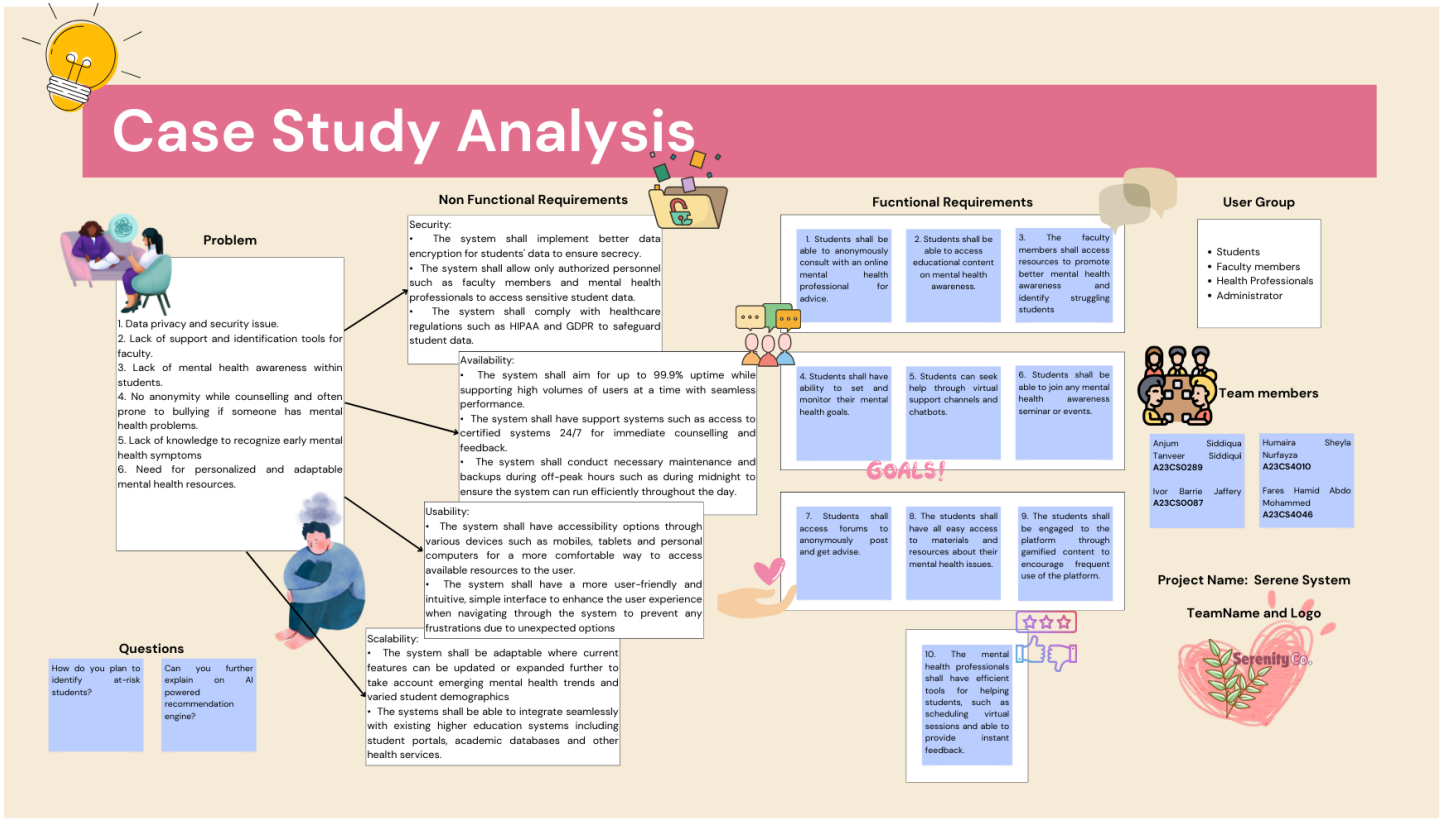
b)TC010_02 : Test <**Review Analytics Dashboard** (SD004 - Alt1)>

Test Case ID:	TC010_02	Test Case Description	Test <**Review Analytics Dashboard** (SD004 - Alt1)>
Created by:	Sheyla	Version:	1.0

No.	Prerequisites	No	Test Data
1	Server is experiencing high load.	1	logID = “102”
			status = “Queued”
Test Condition	Notification is triggered during high server load.		
Step #	Step Details	Expected Result	
1	Trigger a notification for a specific event.	Notification is queued for delivery.	
2	Retry sending notification after load stabilizes.	Notification is sent successfully.	

Appendices:

Appendix A: Evidence of Requirements Elicitation Artefacts



Appendix B: System Design Artefacts

https://www.canva.com/design/DAGcI_KTjlA/GMLTP3ZRUbqwlFv-brYN5g/view?mode=prototype

Appendix C: Traceability Matrix

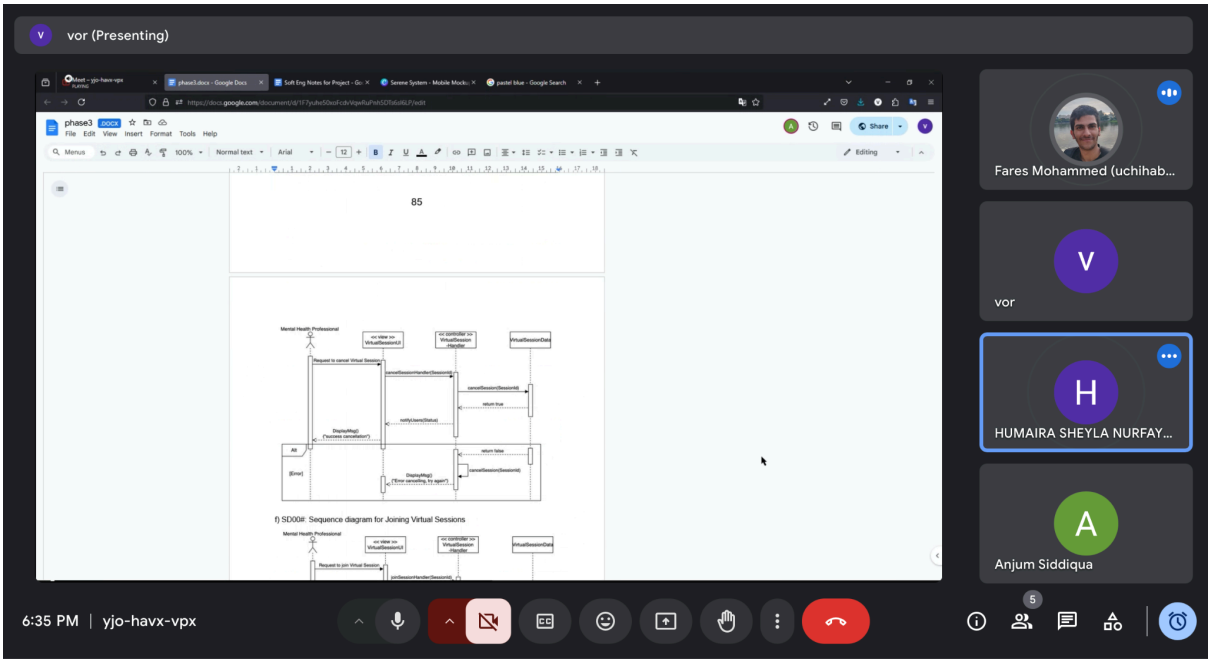
Test Case ID	Use Case ID	Package ID
TC001 for <User Access Registration> Subsystem • TC001_01 • TC001_02 • TC001_03	UC007	P001
TC002 for <Learning and Self Assessment > Subsystem • TC002_01 • TC002_02 • TC002_03 • TC002_04	UC008	P002
TC003 for <Learning and Self Assessment > Subsystem • TC003_01 • TC003_02 • TC003_03 • TC003_04 • TC003_05 • TC003_06	UC009	P002
TC004 for <Communication and Support> Subsystem • TC004_01 • TC004_02 • TC004_03	UC005	P003
TC005 for <Communication and Support> Subsystem • TC005_01 • TC005_02 • TC005_03	UC011	P003

• TC005_04 • TC005_05 • TC005_06		
TC006 for <Communication and Support> Subsystem • TC006_01 • TC006_02	UC003	P003
TC007 for <Professional Management System> Subsystem • TC007_01 • TC007_02 • TC007_03 • TC007_04	UC001	P004
TC008 for <Professional Management System> Subsystem • TC008_01 • TC008_02 • TC008_03	UC002	P004
TC009 for <System Monitoring> Subsystem • TC009_01 • TC009_02	UC004	P005
TC009 for <System Monitoring> Subsystem • TC010_01 • TC010_02	UC004	P005

Appendix D : Complete Package Diagram

https://drive.google.com/file/d/1kidusc6aXiFlOvDaxb5d-VMnBvMJr59r/view?usp=share_link

Appendix E: Group Meeting


vor (Presenting)
phase3
85
f) SD00#: Sequence diagram for Joining Virtual Sessions
Fares Mohammed (uchihab...
vor
HUMAIRA SHEYLA NURFAY...
Anjum Siddiqua
6:35 PM | yjo-havx-vpx