SECP3204: Software Engineering

System Requirements Documentation (SD)

UTM Faculty of Computing Student Engagement System

Version 1.0

09/06/2023

Faculty of Computing

Prepared by: Systema

Revision Page

a. Overview

This is the first draft of the Software Requirements Specification (SRS) for Student Engagement System.

b. Target Audience

The target audience are Universiti Teknologi Malaysia (UTM) Faculty Computing's students, lecturers, Student Representative Council members, and admins.

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
Bernice Lim Jing Xuan	Project Manager	Project planning	Complete
		Contact Stakeholder	Complete
Chai Yu Tong	Developer	System Design	Partially complete
		System Implementation	Incomplete
Kek Jesslyn	Support Engineer	Product Testing	Incomplete
		Bug Reporting and Documentation	Incomplete
		Troubleshooting	Incomplete
Wong Qiao Ying	UI / UX Designer	Design interface	Incomplete
		Testing interface	Incomplete
Yeoh Chong Yi	Technical Writer	Conduct research	Partially complete

d. Version Control History

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Yeoh Chong Yi	Completed Section 2.1	21/05/2023
1.0	Bernice Lim Jing Xuan	Completed Section 1.1, 1.2, 1.3, 1.5, 2.2.20, 2.2.21, 2.2.22, 2.2.23	22/05/2023

1.0	Kek Jesslyn	Completed Section 2.2.14, 2.2.15, 2.2.16, 2.2.17, 2.2.18, 2.2.19	22/05/2023
1.0	Chai Yu Tong	Completed Section 2.2.5, 2.2.6, 2.2.7, 2.2.8, 2.2.9	22/05/2023
1.0	Wong Qiao Ying	Completed Section 2.2.10, 2.2.11, 2.2.12, 2.2.13	22/05/2023
1.0	Yeoh Chong Yi	Partially Completed Section 2.2.1, 2.2.2, 2.2.3, 2.2.4	22/05/2023
1.0	Yeoh Chong Yi	Completed Section 2.2.1, 2.2.2, 2.2.3, 2.2.4	27/05/2023
1.0	Yeoh Chong Yi	Update Section 2.2.1	03/06/2023
1.0	Wong Qiao Ying	Partially Completed Section 2.3	08/06/2023
1.0	Yeoh Chong Yi	Completed Section 1.4	09/06/2023
1.0	Kek Jesslyn	Completed Section 2.3 and 2.4	09/06/2023
1.0	Bernice Lim Jing Xuan	Completed Section 2.2	10/06/2023

Note:
This System Documentation (SD) template is adapted from IEEE Recommended Practice for Software Requirements Specification (SRS) (IEEE Std. 830-1998), Software Design Descriptions (SDD) (IEEE Std. 10161998 1), and Software Test Documentation (IEEE Std. 829-2008) that are simplified and customized to meet the need of SECJ2203 course at Faculty of Computing, UTM. Examples of models are from Arlow and Neustadt (2002) and other sources stated accordingly.

Table of Contents

1. Introduction

1.1 Purpose

This document's goal is to provide a detailed Software Requirements Specification (SRS) for the Student Engagement System project. Without actual implementation, this SRS serves as a guide for documenting the system's functional and non-functional requirements. It outlines the system's specifications and features, if it were to be developed

1.2 Scope

The objective of this project is to develop an extensive blueprint for the student engagement system. This document will encompass functional requirements, user interfaces, data management, system constraints, and other relevant aspects of the system. The primary purpose is to offer a comprehensive comprehension of the system's architecture and functionalities, while also establishing a foundation for future development or potential implementation.

1.3 Definitions, Acronyms and Abbreviation

In order to ensure clarity and consistency throughout this document, the following definitions, acronyms and abbreviations are used.

- SRS: Software Requirements Specification.
- Engagement System : An engagement system is a software or platform designed to facilitate and enhance interaction, participation, and collaboration among users. It aims to actively involve and engage individuals by providing various features and functionalities that encourage their active involvement, communication, and contribution. An engagement system can be used in different contexts, such as educational institutions, online communities, workplace environments, or social networking platforms. The system typically includes components for user registration, content creation and sharing, communication tools (e.g., messaging, forums), gamification elements (e.g., rewards, badges), and analytics to measure and track user engagement levels. The ultimate goal of an engagement system is to foster a sense of belonging, motivation, and continuous involvement among its users.
- Sd: Sequence diagram
- Act: Activity diagram

1.4 **References**

Specify a complete list of references using a standardized reference format.

1. Dubey, P., Pradhan, R.L. and Sahu, K.K. (2023), "Underlying factors of student engagement to E-learning", *Journal of Research in Innovative Teaching & Learning*, Vol. 16 No. 1, pp. 17-36. https://doi.org/10.1108/JRIT-09-2022-0058
2. Zweekhorst, M.B.M. and Maas, J. (2015), "ICT in higher education: students perceive increased engagement", *Journal of Applied Research in Higher Education*, Vol. 7 No. 1, pp. 2-18. https://doi.org/10.1108/JARHE-02-2014-0022
3. Mason, Roger. (2011). Student Engagement with, and Participation in, an e-Forum. Educational Technology and Society. 14. 258-268.
4. Vijayavalsalan, Beena. (2018). Students' Impressions on the Effectiveness of Online Discussion Forums. Research in Social Sciences and Technology. 3. 86-108. 10.46303/ressat.03.01.6.
5. Ansari, J.A.N., Khan, N.A. Exploring the role of social media in collaborative learning the new domain of learning. *Smart Learn. Environ.* 7, 9 (2020). https://doi.org/10.1186/s40561-020-00118-7

1.5 **Overview**

The SRS document is structured into various sections that offer a comprehensive overview of the student engagement system. Each section delves into specific areas, including system requirements, functional and non-functional specifications, user interfaces, data management guidelines, system constraints, and future enhancements. These sections provide in-depth descriptions, specifications, and requirements to ensure a thorough understanding of the student engagement system's objectives.

This document serves as a valuable reference for project stakeholders, including project managers, development teams, and other individuals involved in comprehending and conceptualizing the student engagement system. It captures the requirements and specifications of the system, facilitating effective

communication, decision-making, and potential future development by providing a clear and comprehensive outline of the proposed system.

2. Specific Requirements

2.1 User characteristics

The UTM Student Engagement System will be used by several types of users, including UTM students, UTM lecturers, the Student Representative Council (SRC), and administrators in the Faculty of Computing.

1. UTM Students:

- Expected to have basic computer skills and familiarity with web-based applications.
- Will use the system to submit feedback to SRC.
- I will use the system to make posts.
- Will use the system to discuss on forums.

2. UTM Lecturers:

- Expected to have basic computer skills and familiarity with web-based applications.
- Will use the system to engage with students, provide educational materials, and make announcements.
- May have varying levels of technical expertise but will receive training on how to use the system effectively.

3. Student Representative Council (SRC):

- Expected to have basic computer skills and familiarity with web-based applications.
- Will use the system to communicate with students, gather feedback, organize student events, and update information about the council.
- May require access to specific administrative features of the system.

4. Administrators:

- Expected to have advanced computer skills and familiarity with system administration.
- Will use the system to manage user accounts and reports.
- May require access to comprehensive data and analytics to monitor accounts and manage reports.

All users, including UTM students, UTM lecturers, the SRC, and administrators, will be required to register for the system by providing personal information such as name and email address. The system will prioritize privacy and security measures to protect user information.

2.2 System Features

Registration and Authentication Module

Descriptions

The system should support registration and authentication for UTM Faculty of Computing's students, lecturers, Student Representative Council members, and administrators.

Requirements

1. Sign Up (UTM Email): The system shall allow users to create an account by supplying the necessary data, including their name, UTM email address, and password.
2. Email Verification: The system shall allow users to validate their email addresses upon registration.
3. Reset Password: The system shall allow users to reset their password via email.
4. Login: The system shall allow users to login using their registered email/username and password.

User Information Module

Descriptions

The system enables users to continue their post and forum drafts, edit their profile and update information boards for SRC.

Requirements

1. Drafts Update : The system shall allow users to continue their post and forum drafts.
2. Update Profile: The system shall allow users to edit their profile information.
3. Delete Profile: The system shall allow users to delete their account.

4. Update SRC Information Board: The system should provide a special accessibility to the SRC for editing, posting and updating the information board on the main page of the system.

Processing Module

Descriptions

The system allows the administrator to filter the posts and validate users feedback to avoid provocative content and make improvements on the system.

Requirements

1. Post Filtration: The system shall offer post filtration capabilities by implementing rules and regulations for content filtering.
2. Validate Feedback: The system should include a feature for the administrator to validate user feedback.

Admin and Reporting Module

Descriptions

The system enables administrators to generate reports to gain insights from the engagement system and to further understand the users.

Requirements

1. Generate Report: The system shall allow administrators to generate reports from the system database from time to time.
2. View Report: The system shall allow administrators to view reports generated by the system.
3. Manage Report: The system shall allow administrators to delete, archive or view details of reports generated.
4. Download Report: The system shall allow administrators to download reports generated by system.

Personalized Dashboard Module

Descriptions

This is a dashboard that allows users to share their daily opinions or expressions on anything. Users can customize this dashboard to become what they wanted it to be.

Requirements

1. Post Creation: The system manages to let users post on what they want and share it to their friends as well as the post will be able to save as a draft if the internet connections are not available.
2. Edit Post: This system allows users to edit the posts that have been posted and the edited post will be able to save as a draft if there are no internet connections.
3. Delete Post: Users will be able to delete unwanted posts so that it would not be seen by other people again.
4. Customize Dashboard: This module would also allow users to customize the dashboard as they wanted it to be.

Feedback Module

Descriptions

This module enables users to post their feedback or complaints on anything and it would be seen by the respective people in charge such as Student Representative Council(SRC) and SRC can view them as well as taking actions.

Requirements

1. View Feedback: The system would allow SRC to view all the feedback submitted.
2. Submit Feedback: Users can submit their feedback by clicking the submit button.
3. Manage Feedback: Admins can manage the feedback by filtering based on their priority so that SRC can take effective actions.

Anonymous Forum Module

Descriptions

UTM Faculty of Computing's students and lecturers are able to create forums about opinions, facts, reports, researches, surveys and many more through this forum without showing their own name on any of the forum.They are also allowed to edit and delete their forum descriptions and even comment in every forum..

Requirements

1. Forum Creation: Provide functionality for creating multiple forums where users can discuss different topics.
2. Forum Reply: Allow users to reply to existing forums, facilitating discussion and interaction.
3. Threaded Discussions: Support threaded discussions to allow for organized conversations within each forum, enabling users to follow specific topics or responses.

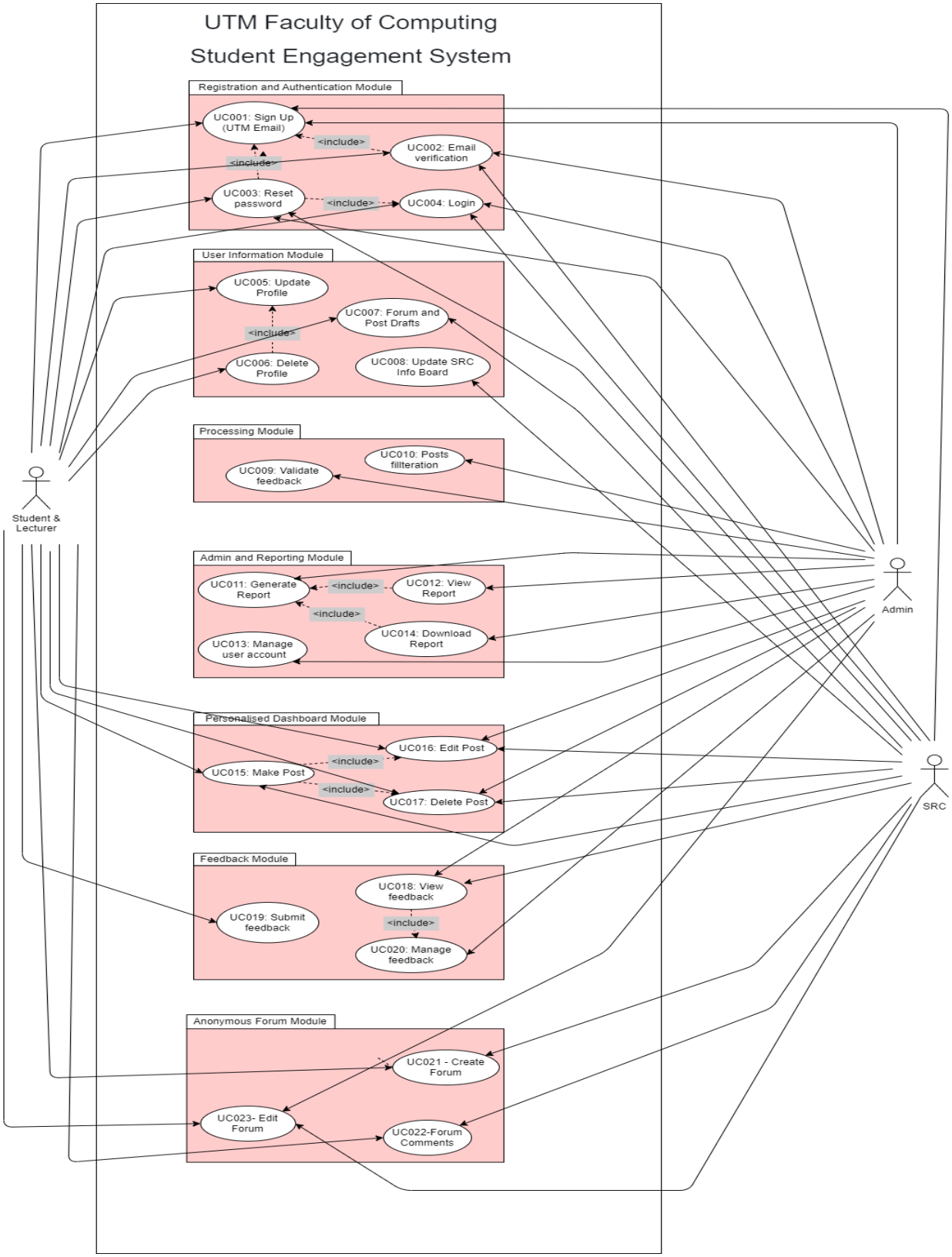

Figure 2.2.0.1 Use Case Diagram for <UTM Faculty of Computing Student Engagement System>

Table 2.2.0: Description of Module and Functions for UTM Faculty of Computing Student Engagement System.

Module	***Function***	***Description***
Registration and Authentication	UC001 Sign Up	Users can sign up using their UTM email.
	UC002 Email Verification	Users can verify the email used to sign up.
	UC003 Reset Password	Users can reset their password.
	UC004 Login	Users can login with their username and password.
User Information	UC005 Update Profile	Users can edit their profile
	UC006 Delete Profile	Users can delete their profile
	UC007 Forum and Post Draft	Users can save their forum/post into the draft box if they haven't post it
	UC008 Update SRC Information Board	Student Representative Council (SRC) can get access to update the information board
Processing	UC009 Posts Filtration	Posts submitted by users will undergo content filtering.
	UC010 Validate Feedback	Users' feedback will be reviewed and validated by the team in order to make improvements
Admin and Reporting	UC011 Generate Report	Admins can generate report based on dates
	UC012 View Report	Admins can view report after generating report
	UC013 Manage Report	Admins can manage report
	UC014 Download Report	Admins can download report
Personalized dashboard	UC015 Make Post	Users can make posts in their dashboard.
	UC016 Edit Post	Users can edit posted posts.
	UC017 Delete Post	Users can delete any posts that they do not want.
Feedback	UC018 View Feedback	Users can have a glance at their posted feedback.
	UC019 Submit Feedback	Users can submit their feedback.
	UC020 Manage Feedback	Users can manage submitted feedback.
Anonymous Forum	UC021 Create Forum	Users are able to create a forum.
	UC022 Forum Comments	Users are allowed to comment at any post.
	UC023 Edit Forum	Users are allowed to edit their post after posting it.

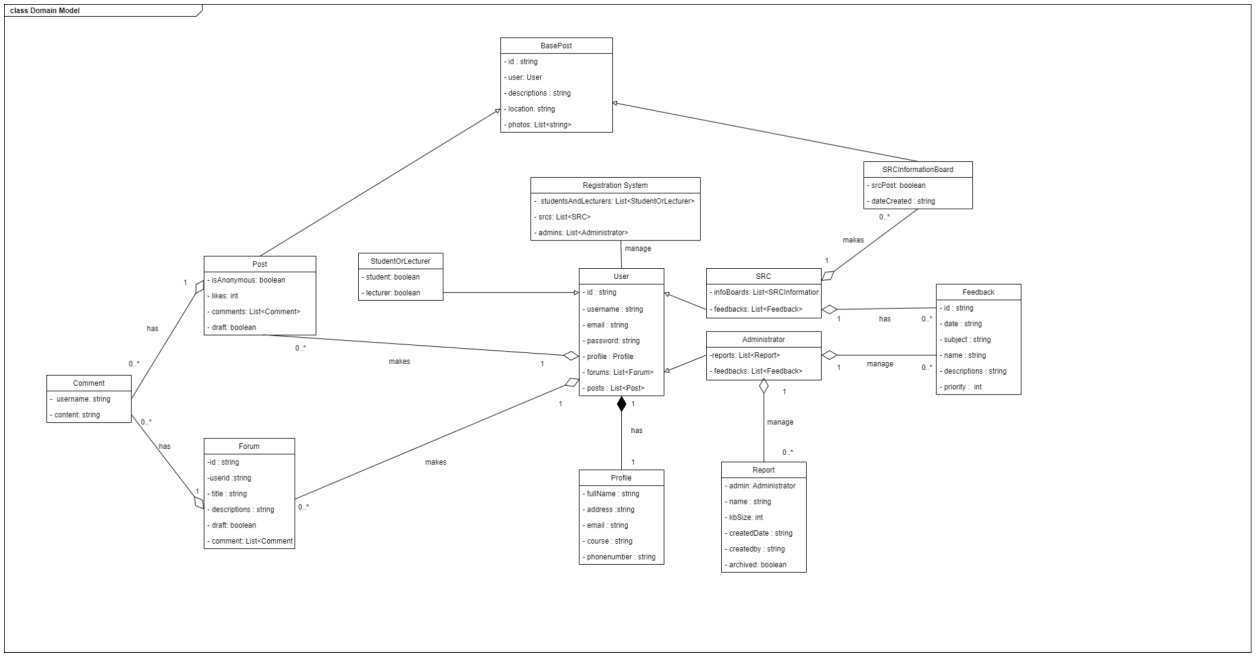

***Figure 2.2.0.2 Domain Model for* <UTM Faculty of Computing Student Engagement System>**

2.2.1 **UC001: Use Case <Sign Up (UTM Email)>**

Table 2.2.1: Use Case Description for Sign Up (UTM Email)

Use case: Sign Up (UTM Email)
Use Case ID: UC001
Use Case Name: Sign Up (UTM Email)
Actors: 1. Faculty of Computing's Students 2. Faculty of Computing's Lecturers 3. Faculty of Computing's Student Representative Council Members 4. Administrators
Preconditions: Has a stable internet environment.
Flow of events: 1.The use case starts when the user clicks on "Sign up" at the main login page. 2. The system displays page asking user to choose whether to sign up as student / lecturer, Student Representative Council member, or administrator. 3. The user chooses to click on "Student / Lecturer", "SRC", or "Admin". 4.The system displays suitable sign up page containing registration form based on whether user chooses to click on "Student / Lecturer", "SRC", or "Admin". 5. The user fills in the required details (UTM email, password etc). 6. The user clicks on "Sign up" button. 7. The system validates the registration form.
Postconditions: 1. The system adds the user account registered into user database. 2. System redirects user back to login page.
Alternative flow *n*: If user tries to register an account with one or more invalid details such as entering existing username or UTM email address: a. Alternative flow starts after step 4 of the main flow. b. The system finds one or more invalid details c. The system displays invalid sign up message. d. The system displays sign up page again. e. The user can re-enter the correct details
Postconditions: None
Exception Flow: None

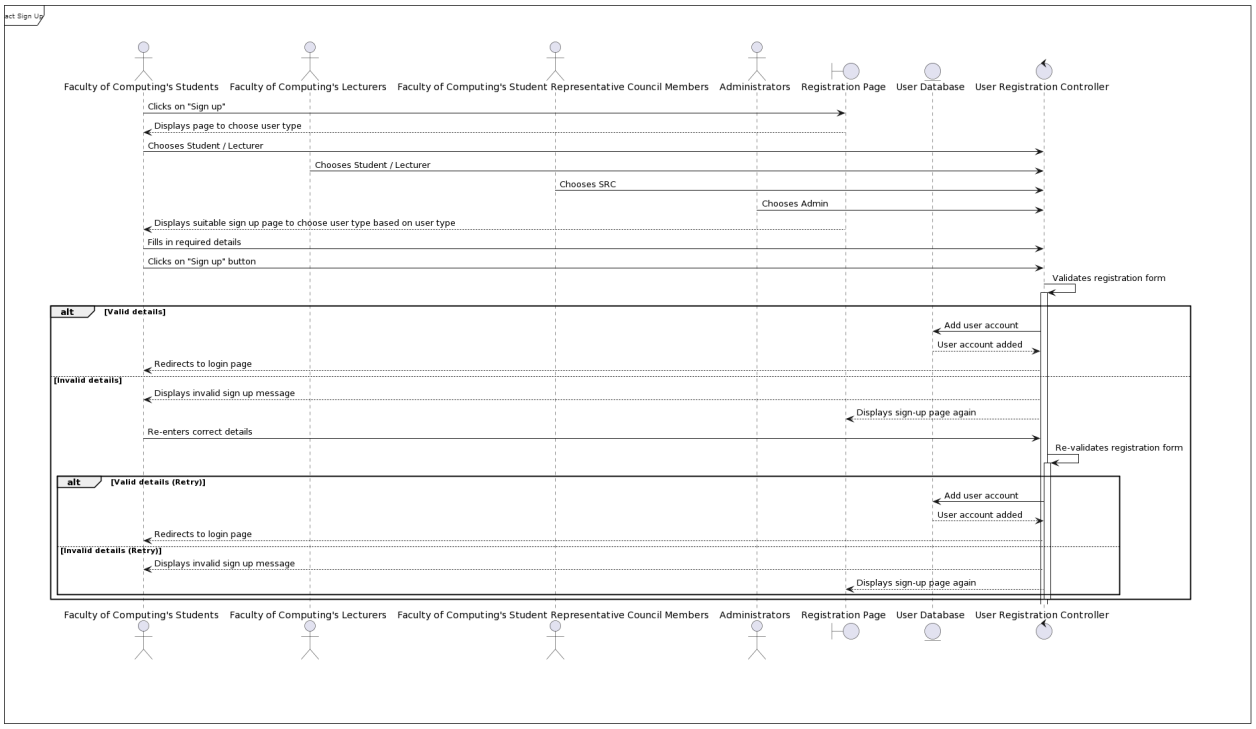

Figure 2.2.1.1: Sequence Diagram for <Sign Up (UTM Email)>

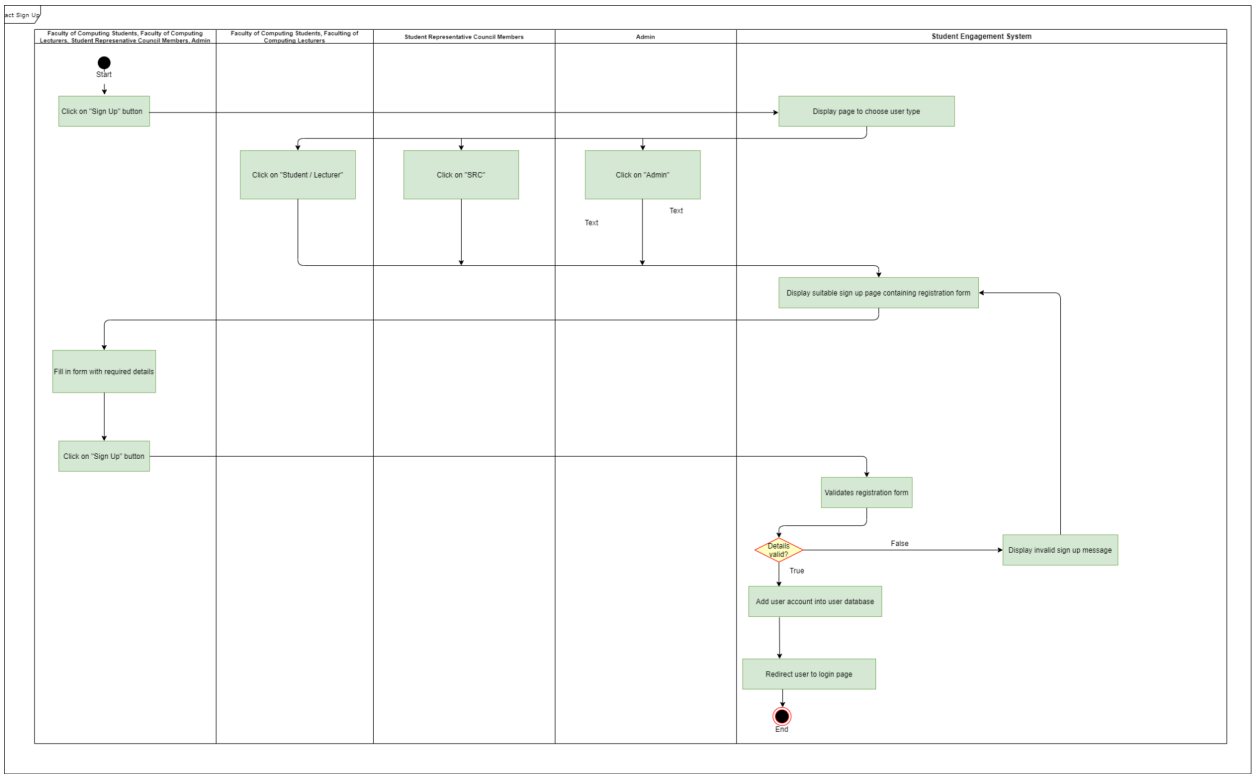

Figure 2.2.1.2: Activity Diagram for <Sign Up (UTM Email)>

2.2.2 **UC002: Use Case <Email Verification>**

Table 2.2.2: Use Case Description for Email Verification

Use case: Email Verification
Use Case ID: UC002
Use Case Name: Email Verification
Actors: 1. Faculty of Computing's Students 2. Faculty of Computing's Lecturers 3. Faculty of Computing's Student Representative Council Members 4. Administrators
Preconditions: 1. Has a stable internet environment. 2. User has completed the registration form with no errors.
Flow of events: 1. After the user completes the registration form, the system generates a verification link unique to the user's account. 2. The system sends an email to the user's registered email address. 3. The email includes a message informing the user about the need for email verification and provides a verification link. 4. The user checks their email inbox and opens the email received from the system. 5. The user clicks on the verification link provided in the email. 6. The system receives the verification request and verifies the link's authenticity. 7. If the verification link is valid and has not expired, the system marks the user's account as verified. If the verification link is expired, alternate flow follows. If the verification link is invalid or tampered, Exception 2 follows. 8. The system redirects the user to a confirmation page and displays a success message indicating that their email has been successfully verified. 9. The user can now log in to their account and access the system's features and functionalities.
Postconditions: 1. The user's email address is verified, and their account status is marked as verified. 2. The user can login using the registered account.
Alternative flow *n*: If verification link is expired: a. Alternative flow starts at step 4 of the main flow. b. The system displays an error message indicating that the link has expired. c. The user can request a new verification email. d. Alternative flow proceeds in the same way as the main flow starting from step 2 of the main flow.
Postconditions: None

Exception Flow:

1. Expired verification link

1.1. The system displays an error message indicating that the link has expired.

1.2. The user can request a new verification email.

2. Invalid or tampered verification link

2.1. The system displays an error message indicating that the verification failed.

2.2. The user can contact support for assistance.

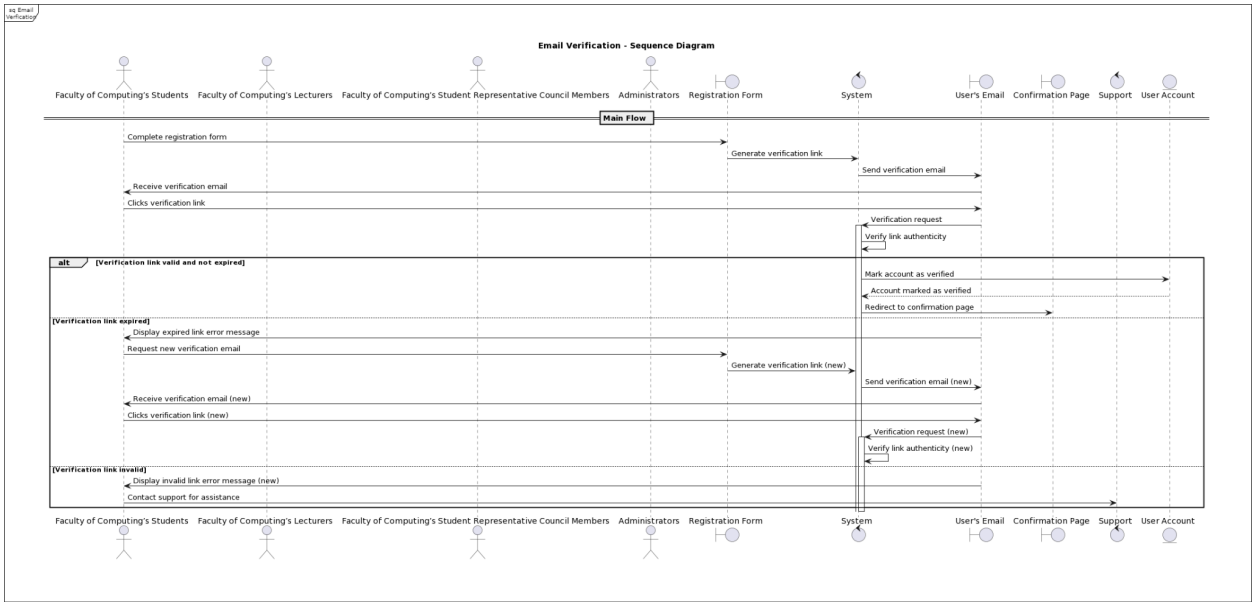

Figure 2.2.2.1: Sequence Diagram for <Email Verification>

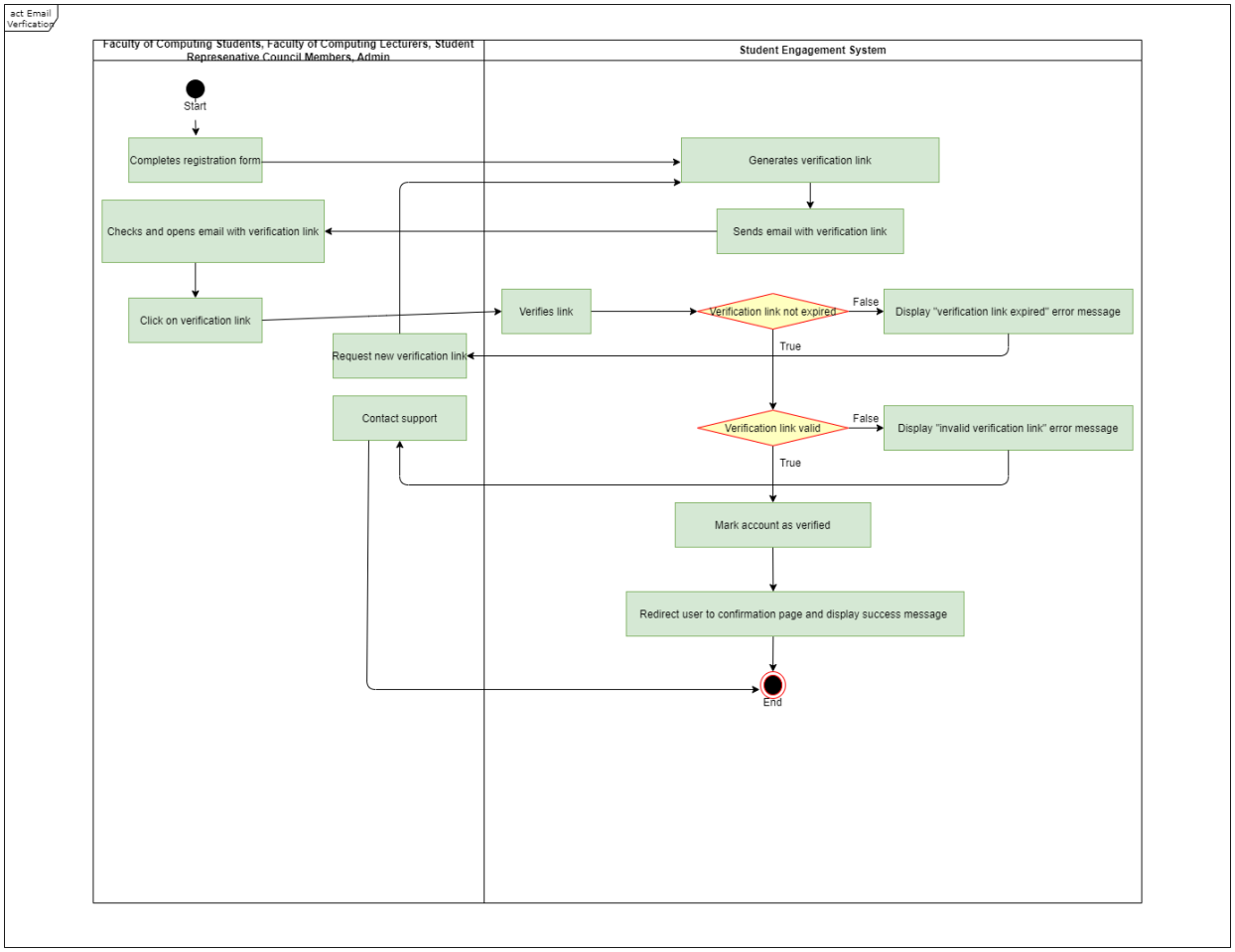

Figure 2.2.2.2: Activity Diagram for <Email Verification>

2.2.3 **UC003: Use Case <Reset Password>**

Table 2.2.3: Use Case Description for Reset Password

Use case: Reset Password
Use Case ID: UC003
Use Case Name: Reset Password
Actors: 1. Faculty of Computing's Students 2. Faculty of Computing's Lecturers 3. Faculty of Computing's Student Representative Council Members 4. Administrators
Preconditions: 1. Has a stable internet environment. 2. User has successfully signed up and has verified his/her account. 3. User has forgotten their account password and wants to reset it.
Flow of events: 1. The user clicks the "Reset Password" link on the login page. 2. The system displays a password reset request form. 3. The user enters their registered UTM email address associated with the account for which they want to reset the password. 4. The user submits the password reset request form. 5. The system verifies the provided email address. If the email address is invalid, Exception 1 follows. 6. The system generates a unique password reset link associated with the user's account. 7. The system sends an email to the user's registered email address. 8. The email includes a message informing the user about the password reset process and provides a password reset link. 9. The user checks their email inbox and opens the email received from the system. 10. The user clicks on the password reset link provided in the email. 11. The system receives the password reset request and verifies the link's authenticity. 12. If the password reset link is valid and has not expired, the system allows the user to enter a new password. If the password reset link is expired, Exception 2 follows. 13. The user enters a new password following the system's password requirements. 14. The user submits the new password. 15. The system validates the password entered. If the user enters a password that does not meet the system's password requirements, Exception 3 follows. 16. The system updates the user's account with the new password. 17. The system redirects the user to a confirmation page or displays a success message indicating that their password has been successfully reset. 18. The user can now log in to their account using the new password.
Postconditions: 1. The user's account password is successfully reset.

2. The user can login using the new account password.
Alternative flow *n*: If the user enters an invalid email address: a. The system displays an error message indicating that the entered email address is invalid. b. The system displays an error message indicating that the entered email address is invalid. c. The system prompts the user to enter a valid email address. If the user does not receive the password reset email within a reasonable time: a. Alternative flow starts after step 9 of the main flow. b. User selects the “Request reset password link” again. c. Alternative flow proceeds in the same way as the main flow starting from step 7 of the main flow. If the user’s password reset link is expired: a. The system should display an error message indicating that the password reset link has expired. b. The user can request a new password reset email. If the user enters an invalid password: a. The system displays an error message indicating that the password does not meet the system's password requirements and indicating the requirements. b. The system prompts the user to enter a valid password.
Postconditions: None
Exception Flow: None

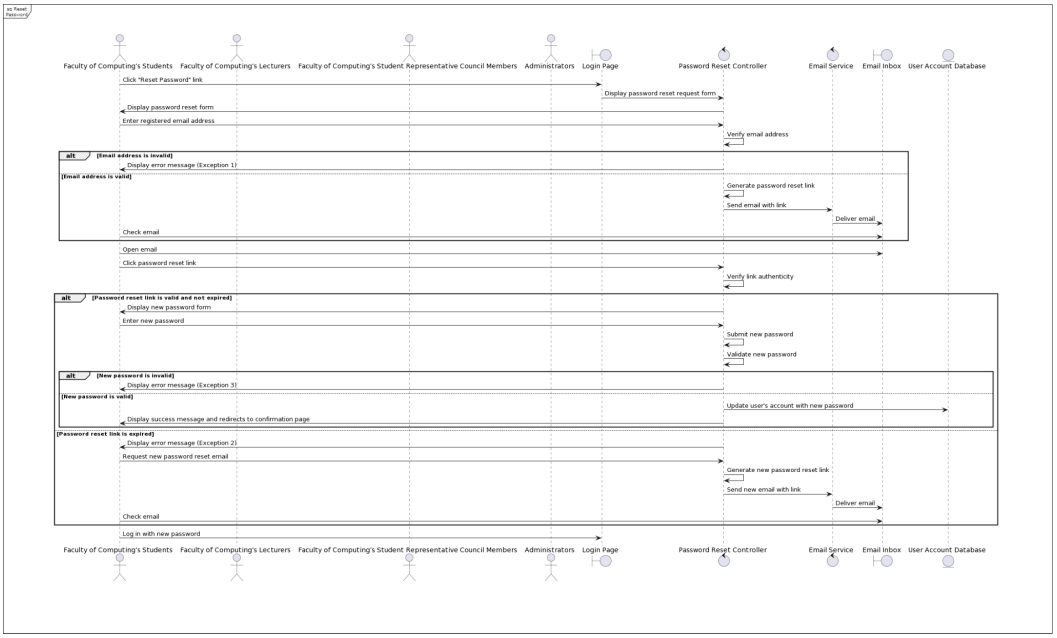

Figure 2.2.3.1: Sequence Diagram for <Reset Password>

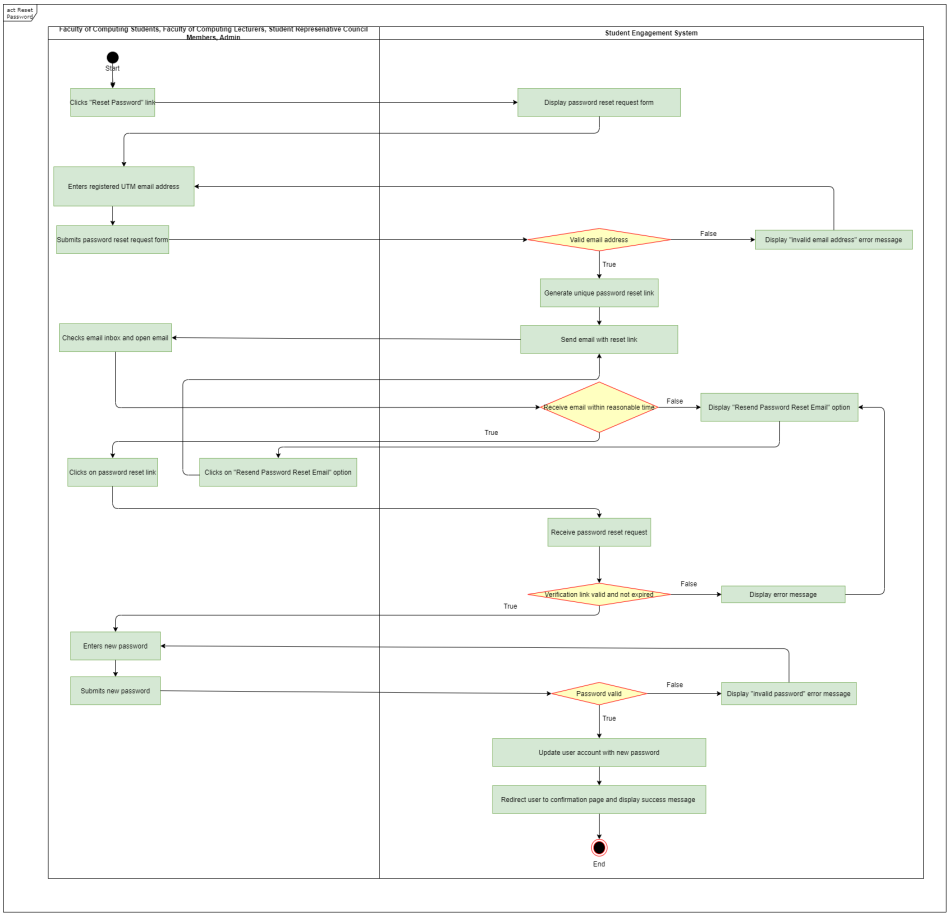

Figure 2.2.3.2: Activity Diagram for <Reset Password>

2.2.4 **UC004: Use Case <Login>**

Table 2.2.4: Use Case Description for Login

Use case: Login
Use Case ID: UC004
Use Case Name: Login
Actors: 1. Faculty of Computing's Students 2. Faculty of Computing's Lecturers 3. Faculty of Computing's Student Representative Council Members 4. Administrators
Preconditions: 1. Has a stable internet environment. 2. User has successfully signed up and has verified his/her account.
Flow of events: 1. The user opens the login page of the system. 2. The system displays the login form, prompting the user to enter their username/email and password. 3. The user enters their username/email and password in the respective input fields. 4. The user submits the login form. 5. The system receives the login request and verifies the provided credentials. 6. The system authenticates the user's identity by comparing the entered credentials with the stored credentials. 7. If the credentials are valid, the system grants access to the user and logs them into their account. If the user account is not verified, Exception 1 follows. 8. The system redirects the user to the system's dashboard page. 9. The user gains access to their account and can utilize the system's features and functionalities.
Postconditions: 1. The user is successfully logged into their account. 2. The user can access their account and utilize the system's features and functionalities.
Alternative flow *n*: If the user enters invalid credentials (e.g., incorrect username/email or password): a. Alternative flow starts after step 5 of the main flow. a. The system displays an error message indicating login failure due to invalid credentials. b. Alternative flow proceeds in the same way as the main flow starting from step 3 of the main flow.
Postconditions: 1. The user is successfully logged into their account.

2. The user can access their account and utilize the system's features and functionalities.
Exception Flow: 1. User account not verified 1.1. The system displays an error message instructing the user to verify their email address before logging in. 2. Encountered technical issues/errors 2.1. The system display an appropriate error message related to the technical issue/error 2.2. The system provides support or assistance options.

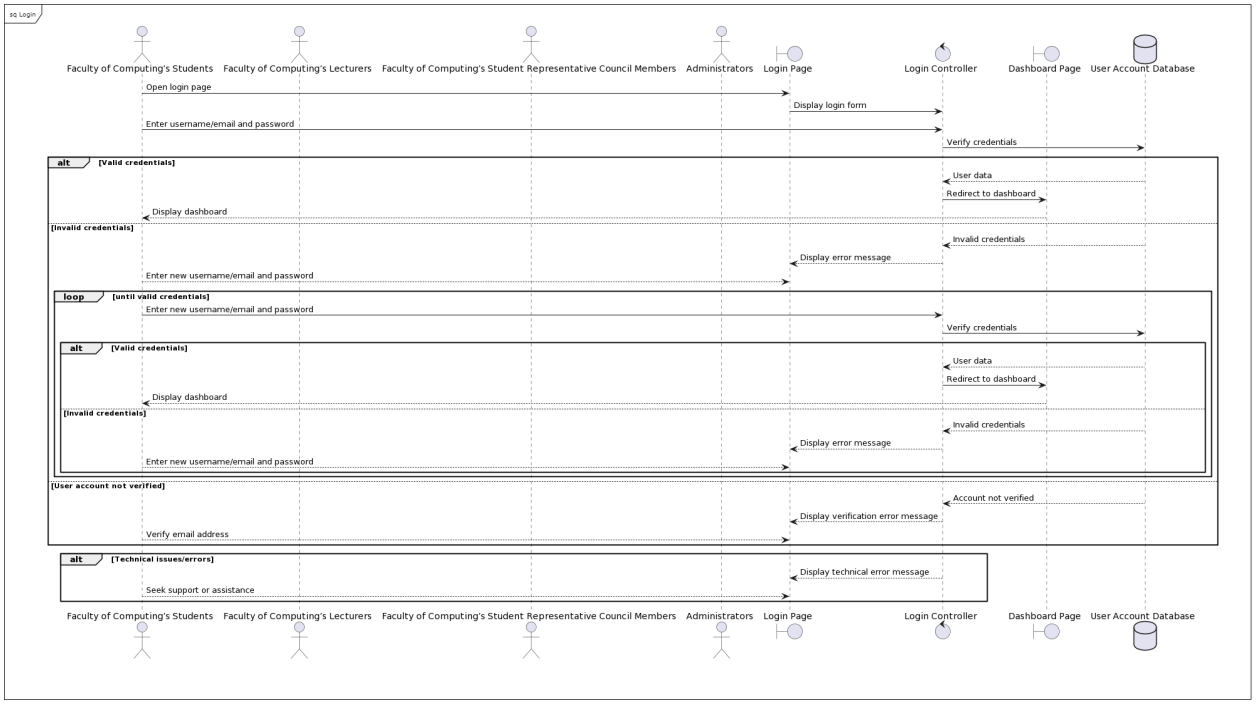

Figure 2.2.4.1: Sequence Diagram for <Login>

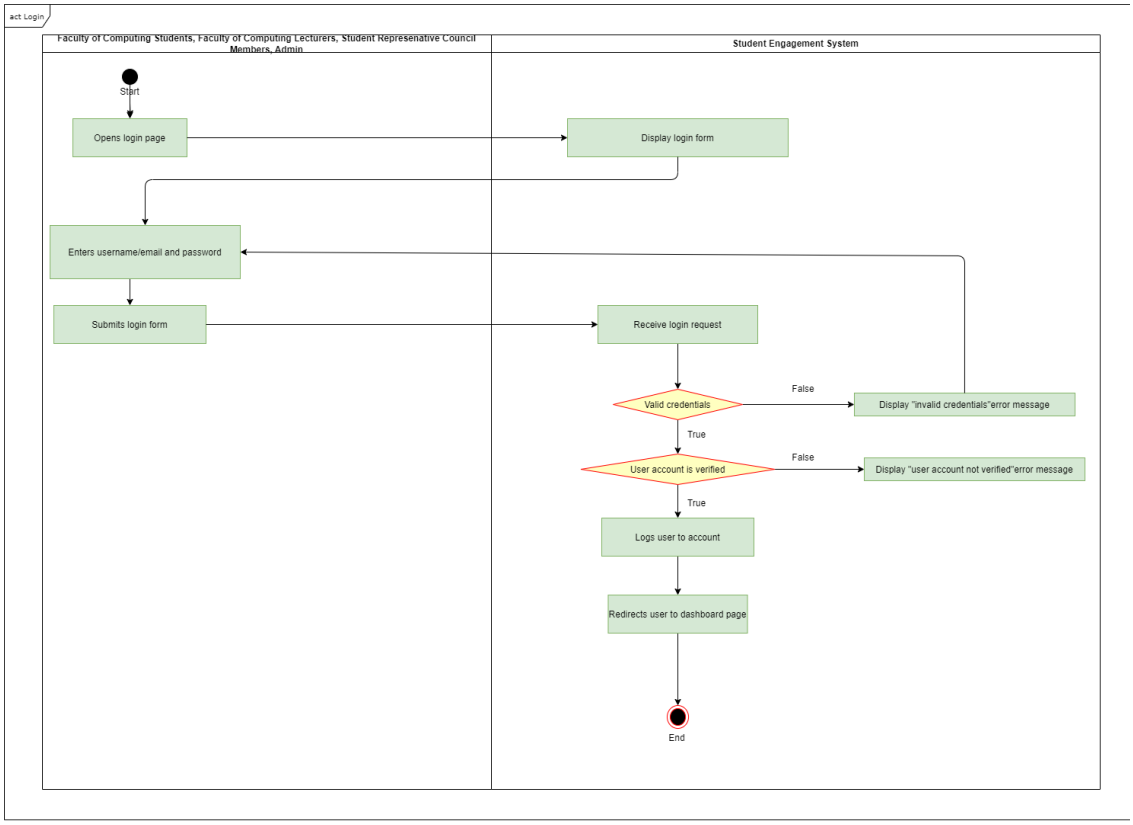

Figure 2.2.4.2: Activity Diagram for <Login>

2.2.5 **UC005 : Use Case <Update Profile>**

Table 2.2.5: Use Case Description for Updating Profile

Use case: Update Profile
Use Case ID: UC005
Use Case Name: Update Profile
Actors: 1.Faculty Computing's Students 2.Faculty Computing's Lecturers
Preconditions: 1. The user must be authenticated and logged into the system 2. The user's profile information must exist in the system
Flow of events: 1. The user selects the "Update Profile" option from the dashboard/system's menu 2. The system displays the user's current profile information 3. The user modifies the desired fields in the profile such as name, contact information and biography 4. The user submits the updated profile information 5. The system validates the updated information 5.1. Valid updated information 5.1.1. The system saves the changes and displays a success message 5.2. Invalid updated information 5.2.1. The system displays an error message and prompts the user to correct the invalid fields
Postconditions: 1. The user's profile information is updated in the system.
Alternative flow *n*: 1. If the user decides to cancel the profile update, they can choose to discard the changes and exit the use case 2. The system reverts to displaying the user's previous profile information 3. The use case ends
Postconditions: 1. No changes are made to the user's profile information
Exception flow: 1. The user encountered a technical error or system failure 1.1 The system displays an error message indicating the issue 1.2 The user need to try again later or contact customer service 1.3 The use case ends without any changes

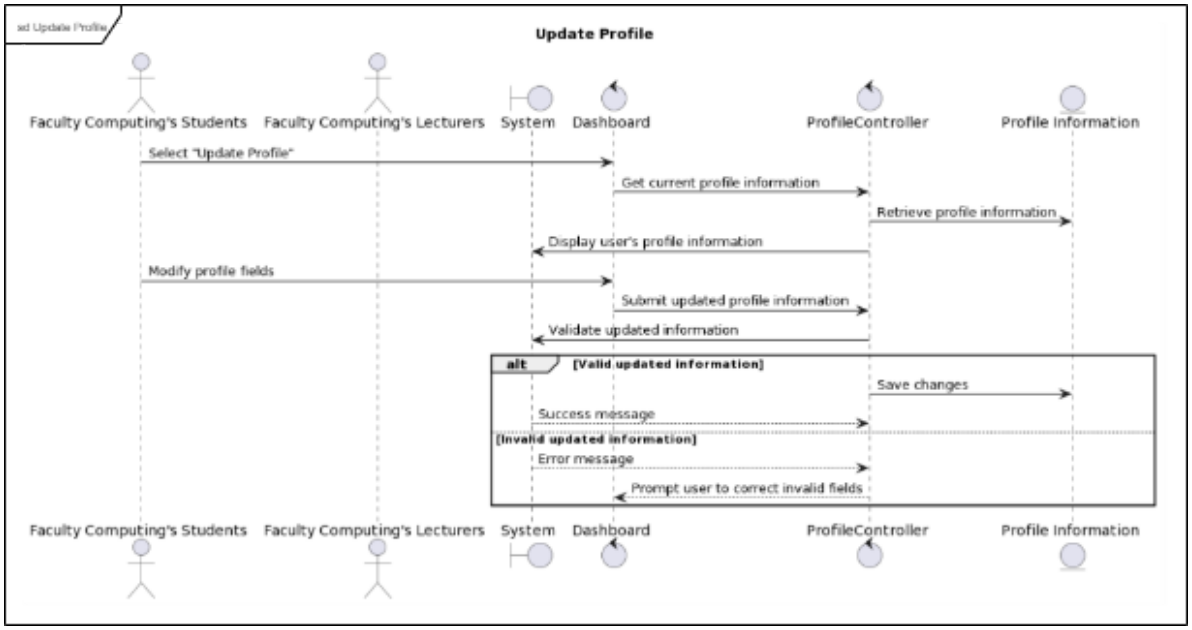

Figure 2.2.5.1: Sequence Diagram for <Update Profile>

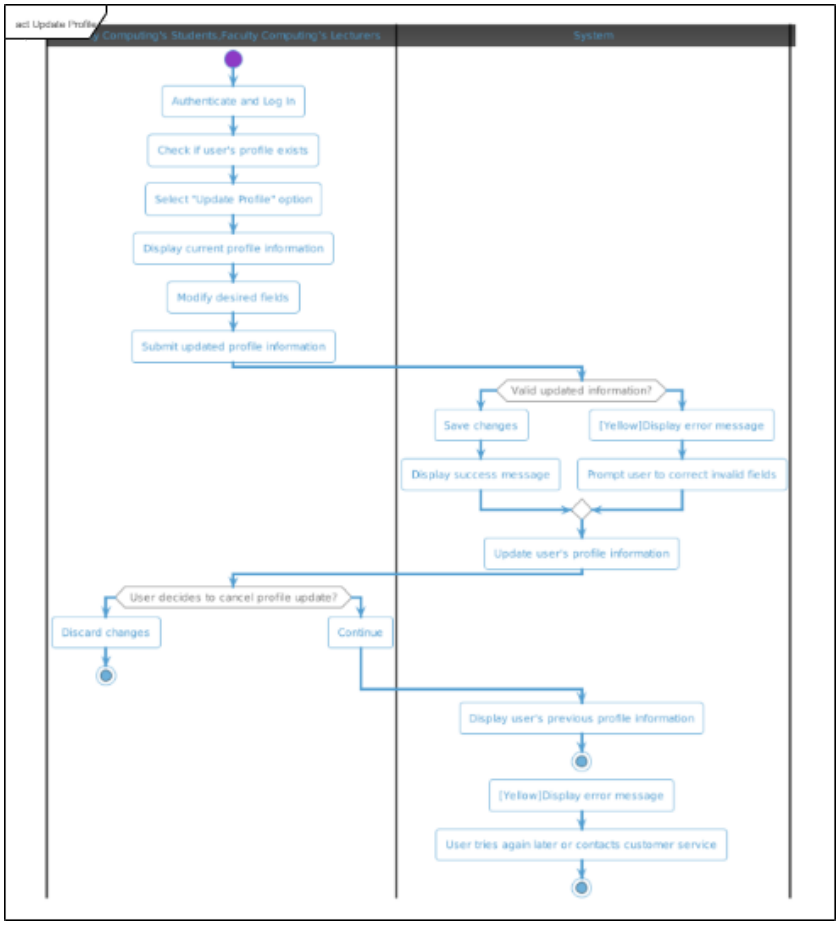

Figure 2.2.5.2: Activity Diagram for <Update Profile>

2.2.6 **UC006 : Use Case <Delete Profile>**

Table 2.2.6: Use Case Description for Deleting Profile

Use case: Delete Profile
Use Case ID: UC006
Use Case Name: Delete Profile
Actors: 1.Faculty Computing's Students 2.Faculty Computing's Lecturers
Preconditions: 1. The user must be authenticated and logged into the system 2. The user's profile information must exist in the system
Flow of events: 1. The user selects the "Delete Profile" option from the dashboard/system's menu 2. The system displays a confirmation message to ensure the user wants to proceed with profile deletion 3. The user confirms their intent to delete the profile 4. The system holds the user's profile for 7 days after confirmation. 5. If the user do not cancel the request after 7 days, the system removes the user's profile and associated data from the system 6. The system displays a success message indicating that the profile has been deleted
Postconditions: 1. The user's profile and associated data are deleted from the system.
Alternative flow *n*: 1. The user cancels the deletion request within 7 days 1.1 The user's profile and associated data will be restored in the system
Postconditions: 1. The user's profile and account remains
Exception flow: 1. The user encountered a technical error or system failure 1.1 The system displays an error message indicating the issue 1.2 The user need to try again later or contact customer service 1.3 The use case ends without any changes

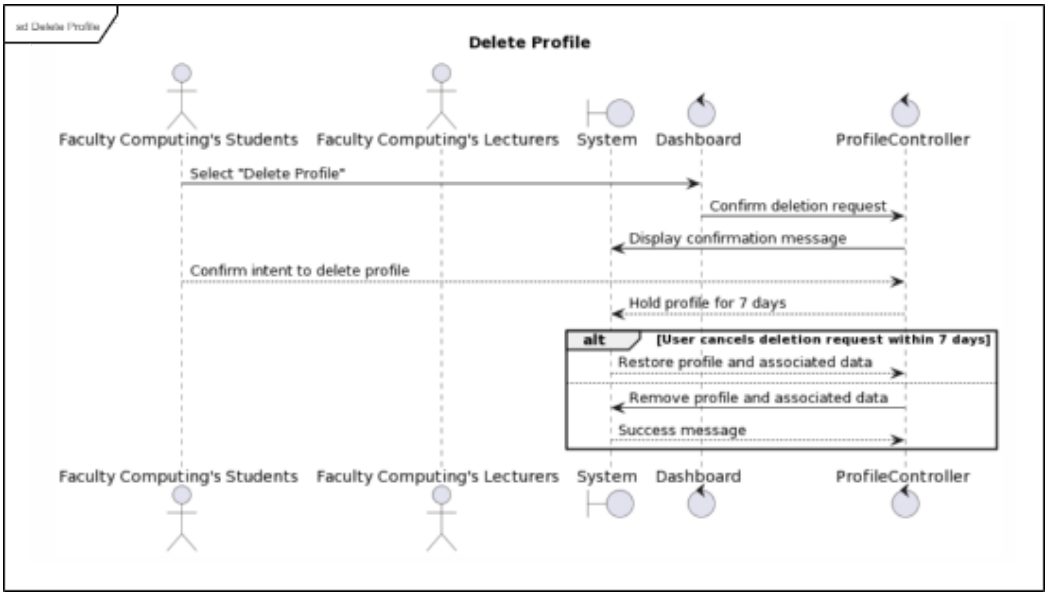

Figure 2.2.6.1: Sequence Diagram for <Delete Profile>

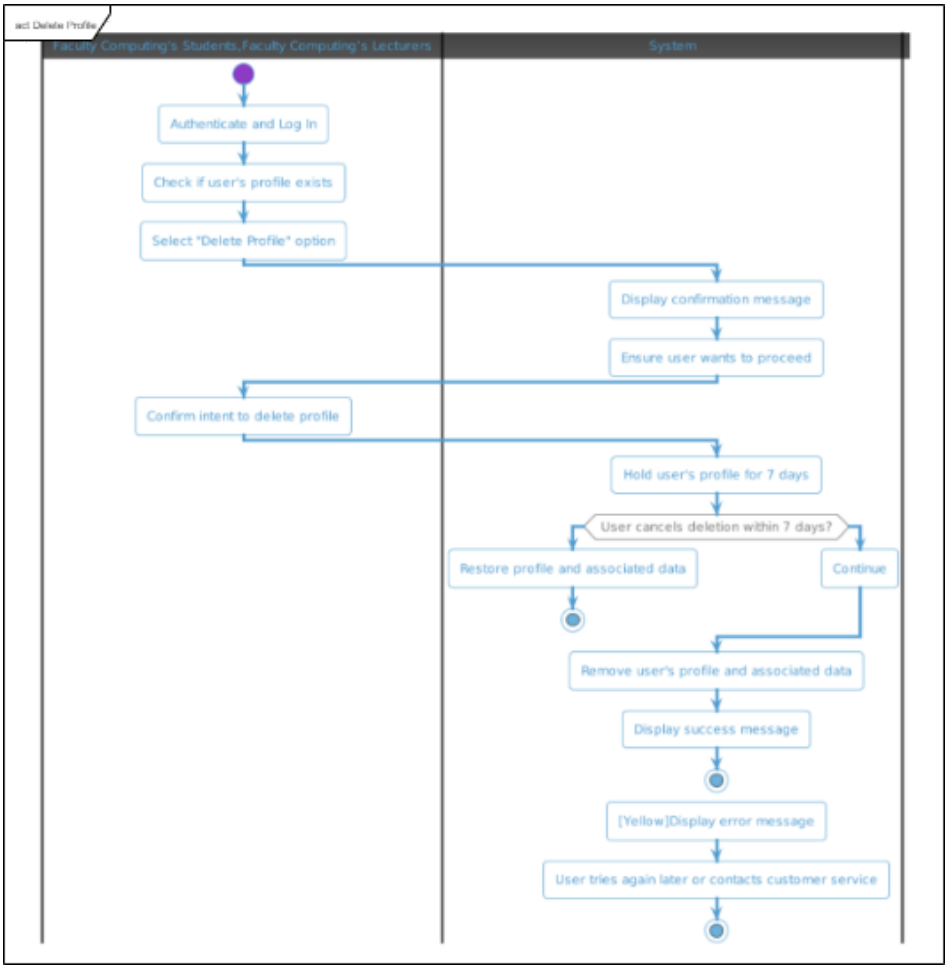

Figure 2.2.6.2: Activity Diagram for <Delete Profile>

2.2.7 **UC007 : Use Case <Forum and Post Draft>**

Table 2.2.7: Use Case Forum and Post Draft

Use case: Forum and Post Draft
Use Case ID: UC007
Use Case Name: Forum and Post Draft
Actors: 1. Student Representative Council (SRC) 2. Student and Lecturer
Preconditions: 1. The user must be authenticated as a member of the Student Representative Council (SRC) 2. Users must have stable internet 3. Users must have a usable account
Flow of events: 1. After the user has decided to cancel editing their forum or post, a message will pop up to confirm their actions. 2. Then, after users confirm to cancel another message will pop up to see whether they wanted to save the forum/posts as draft or discard them. 3. If they click to save as draft, the forum/posts will be saved into the draft box. 4. Then, if they want to access the posts/forum again, they can go to the draft box. 5. Then, they can edit them again.
Postconditions: 1. The draft is successfully saved. 2. Users can access them again in the profile section.
Alternative flow *n*: None
Exception flow: None

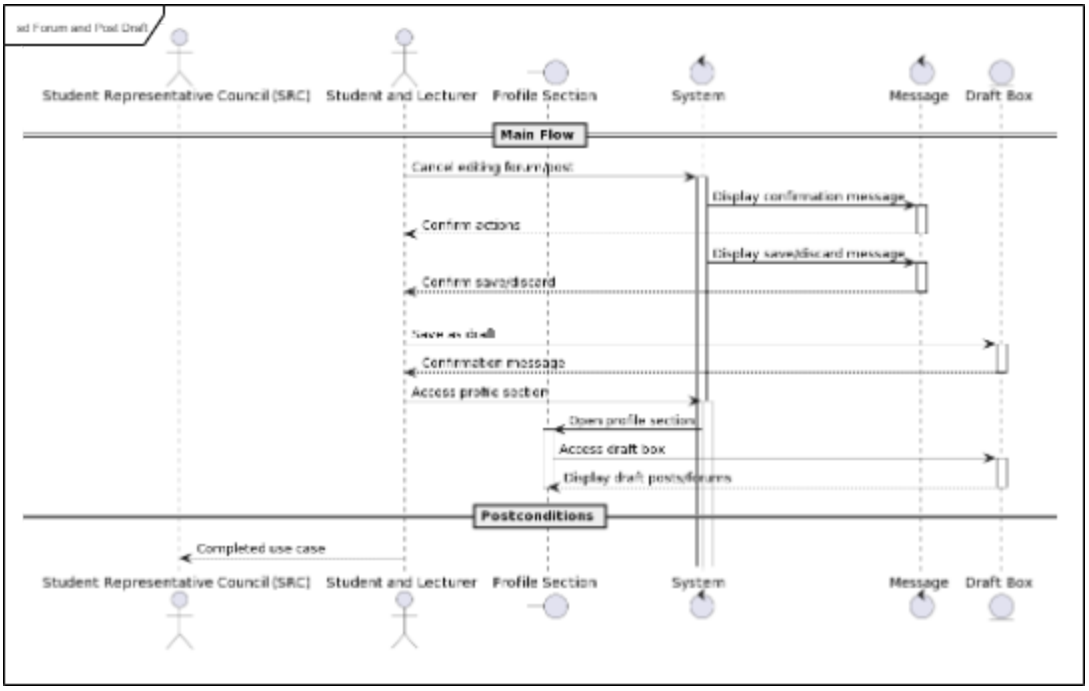

Figure 2.2.7.1: Sequence Diagram for <Forum and Post Draft>

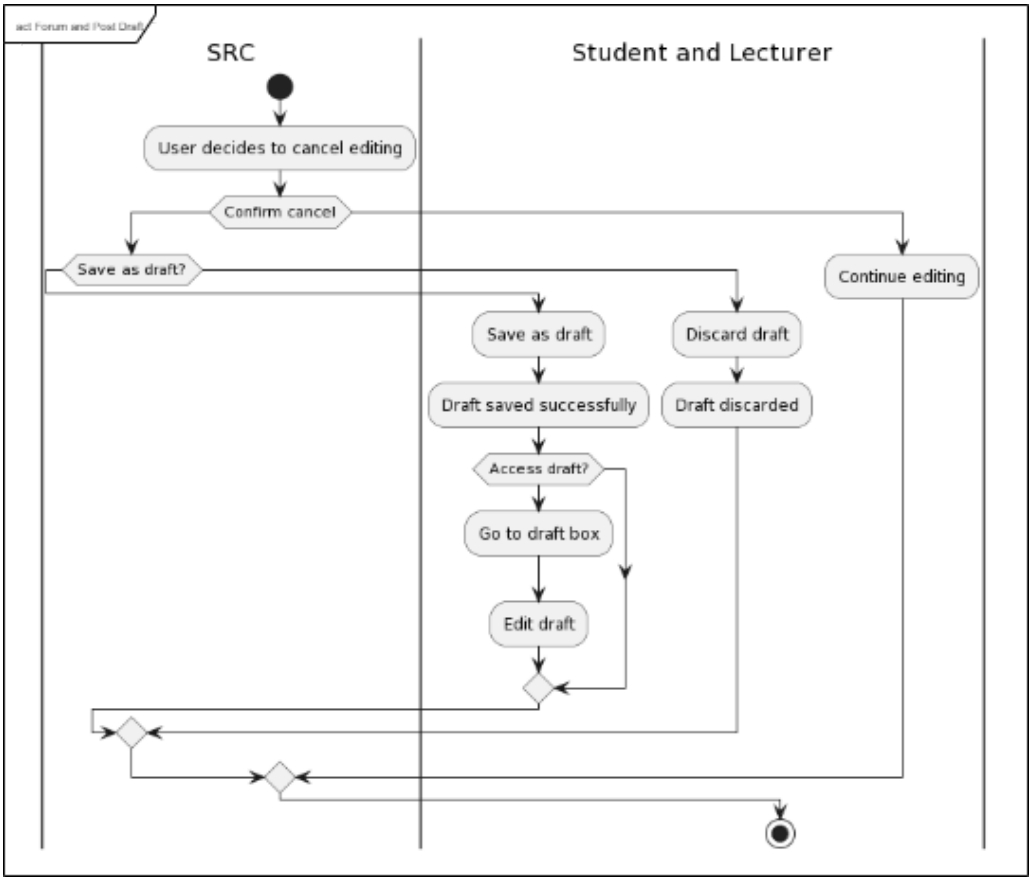

Figure 2.2.7.2: Activity Diagram for <Forum and Post Draft>

2.2.8 **UC008 : Use Case <Update SRC Information Board>**

Table 2.2.8: Use Case Description for Updating SRC Information Board

Use case:Update SRC Information Board
Use Case ID: UC008
Use Case Name: Delete Profile
Actors: 1. Student Representative Council (SRC)
Preconditions: 1. The user must be authenticated as a member of the Student Representative Council (SRC) 2. The user must have permissions to update the SRC information board
Flow of events: 1. The user selects the “Update SRC Information Board” option from the system’s menu 2. The system presents the user with current content of the information board 3. The user edits or adds content to the information board 4. The user saves the changes 5. The system updates the SRC information board with the new content 6. The system displays a success message indicating that the information board has been updated
Postconditions: 1. The SRC information board is updated with the new content
Alternative flow *n*: None
Exception flow: 1. The user encountered a technical error or system failure 1.1 The system displays an error message indicating the issue 1.2 The user need to try again later or contact customer service 1.3 The use case ends without any changes

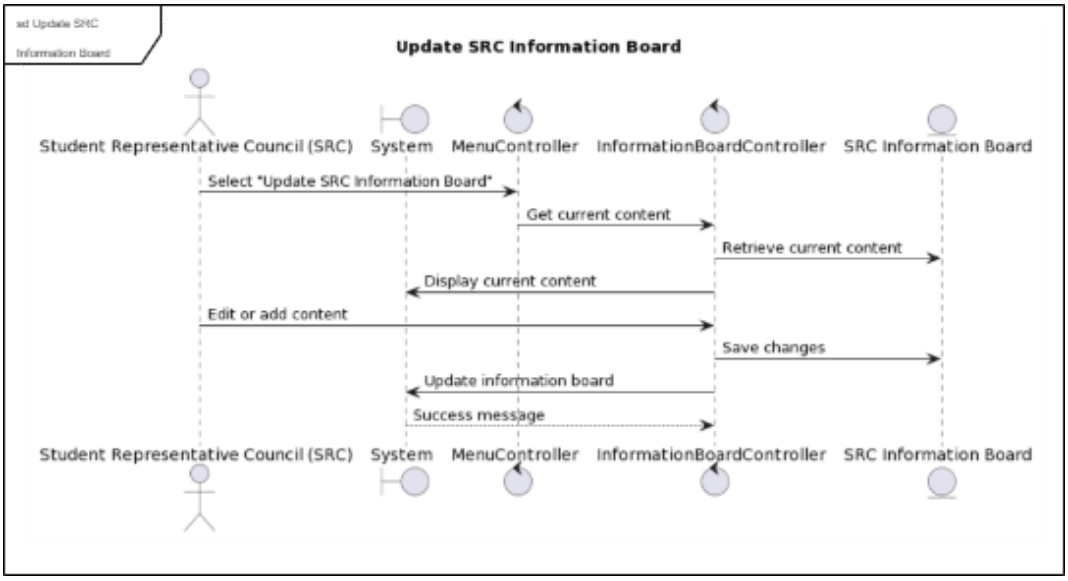

Figure 2.2.8.1: Sequence Diagram for <Update SRC Information Board>

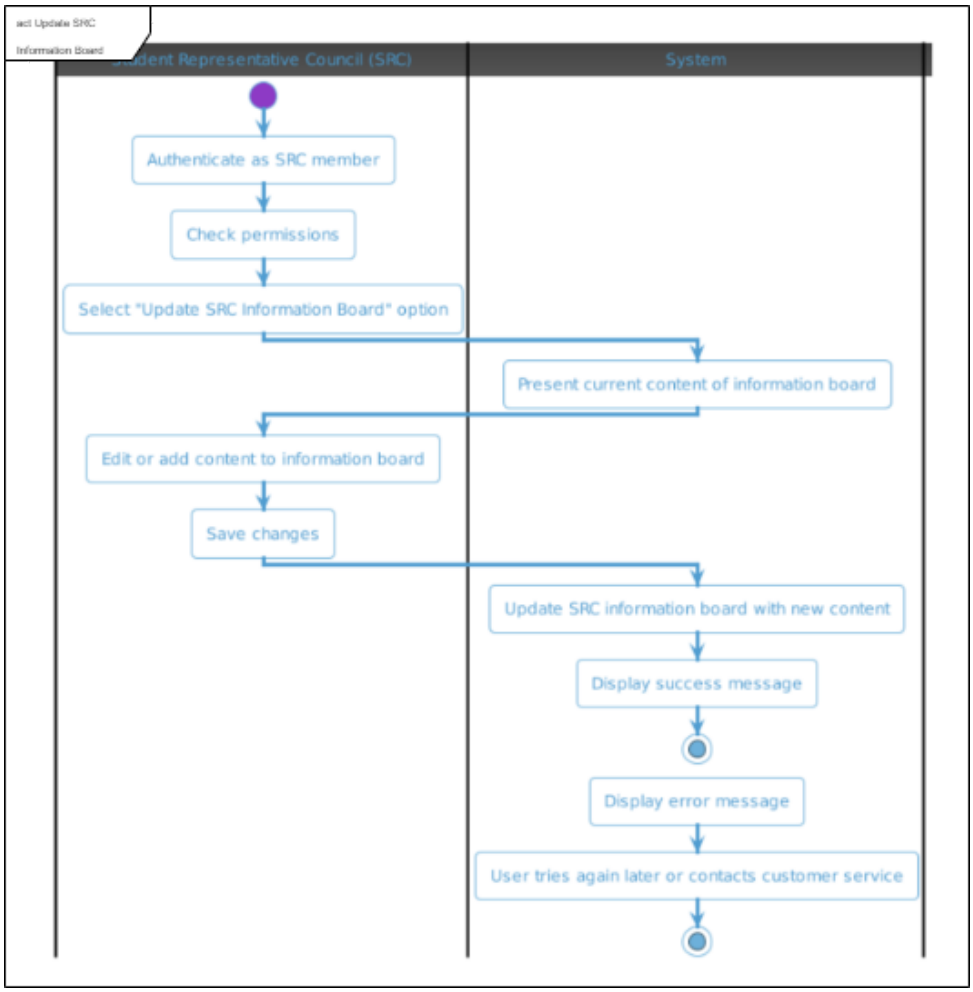

Figure 2.2.8.2: Activity Diagram for <Update SRC Information Board>

2.2.9 **UC009 : Use Case <Posts Filtration>**

Table 2.2.9: Use Case Description for Posts Filtration

Use case: Post Filtration
Use Case ID: UC009
Use Case Name: Posts Filtration
Actors: 1. Administrative
Preconditions: 1. The user must be an administrative user with permissions to perform posts filtration
Flow of events: 1. The administrative user selects the “Post Filtration” option from the system’s menu 2. The system presents a list of available filtering options and criteria 3. The administrative user selects the desired filtering options and criteria 4. The system applies the selected filters to the posts 5. The system displays the filtered posts to the administrative user
Postconditions: 1. The posts are filtered based on the selected options and criteria 2. The administrative user can take further actions on the filtered posts if needed
Alternative flow *n*: None
Exception flow: The user encountered a technical error or system failure 1.1 The system displays an error message indicating the issue 1.2 The user need to try again later or contact customer service 1.3 The use case ends without any changes

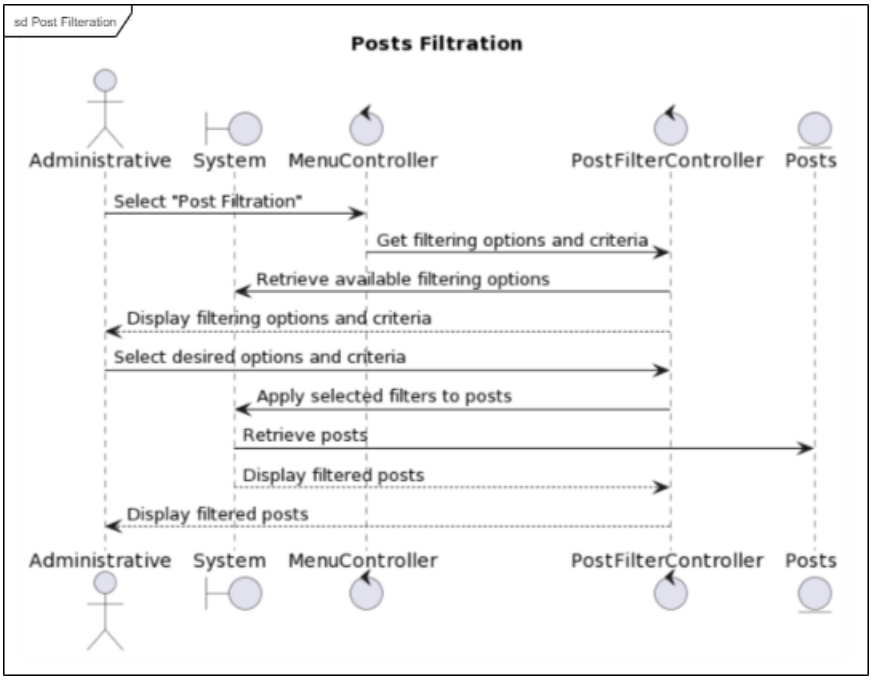

Figure 2.2.9.1: Sequence Diagram for <Post Filtration>

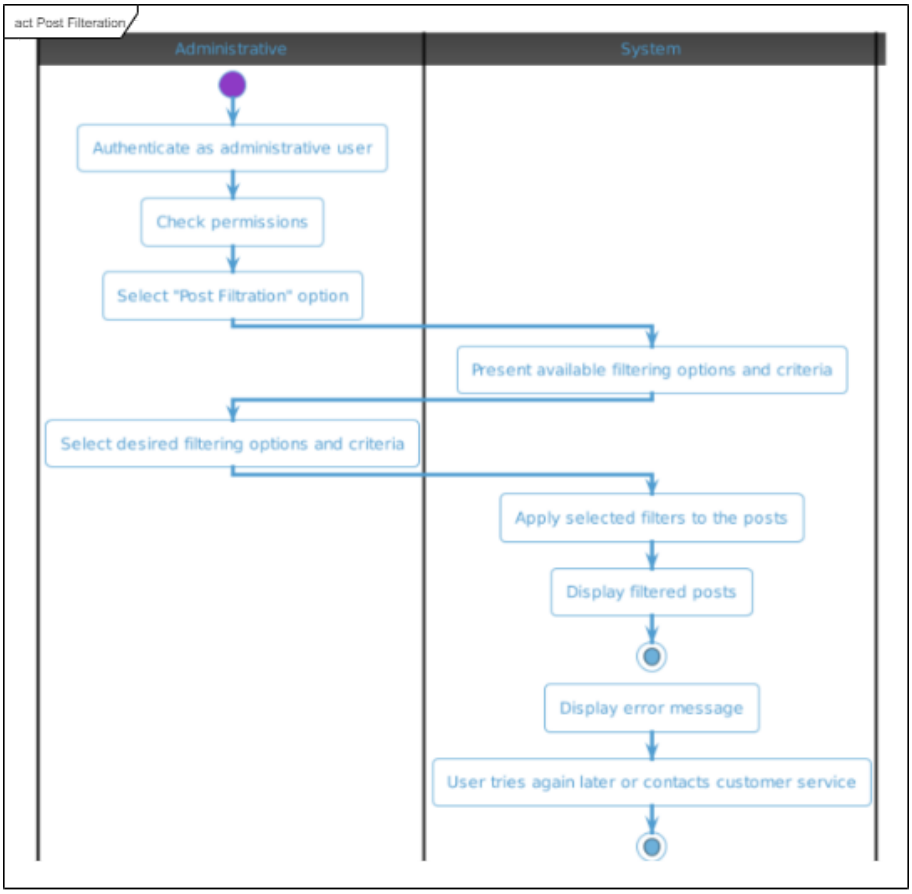

Figure 2.2.9.2: Activity Diagram for <Post Filtration>

2.2.10 **UC010 : Use Case <Validate Feedback>**

Table 2.2.10: Use Case Description for Feedback Validation

Use case: Validate feedback
Use Case ID: UC010
Use Case Name: Validate Feedback
Actors: 1. Administrative
Preconditions: 1. The user must be an administrative user with permissions to perform posts filtration
Flow of events: 1. The administrative user selects the “Validate Feedback” option from the system's menu 2. The system presents a list of feedback items that require validation 3. The administrative user reviews each feedback item 4. For each feedback item, the administrative user decides whether to validate or invalidate it based on predefined criteria 5. Validate feedback 5.1 The system marks it as validated and takes actions such as forwarding it to the relevant team for further processing 6. Invalidated feedback 6.1 The system marks it as invalidated and may notify the user who provided the feedback 7. The administrative user can add comments or notes regarding the validation decision 8. The administrative user can also perform additional actions on the validated feedback, such as archiving it or deleting it
Postconditions: 1. The feedback items are validated or invalidated based on the administrative user’s decisions 2. Validated feedback is processed further according to the system’s workflow 3. Invalidated feedback may be communicated to the user who provided it and mat undergo additional actions based on the system policies
Alternative flow *n*: None
Exception flow: 1. The user encountered a technical error or system failure 1.1 The system displays an error message indicating the issue 1.2 The user need to try again later or contact customer service 1.3 The use case ends without any changes

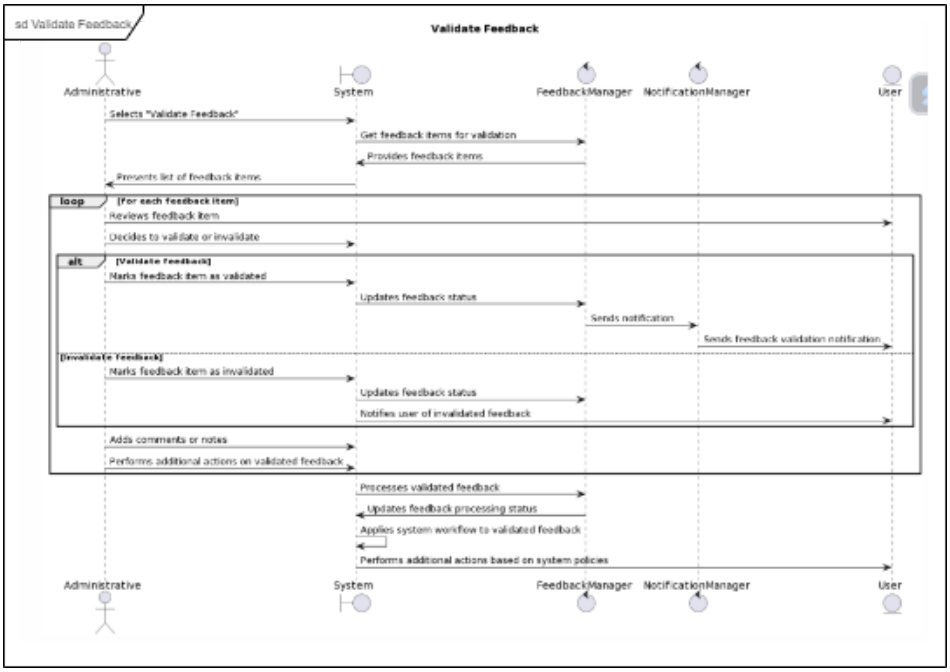

Figure 2.2.10.1: Sequence Diagram for <Validate Feedback>

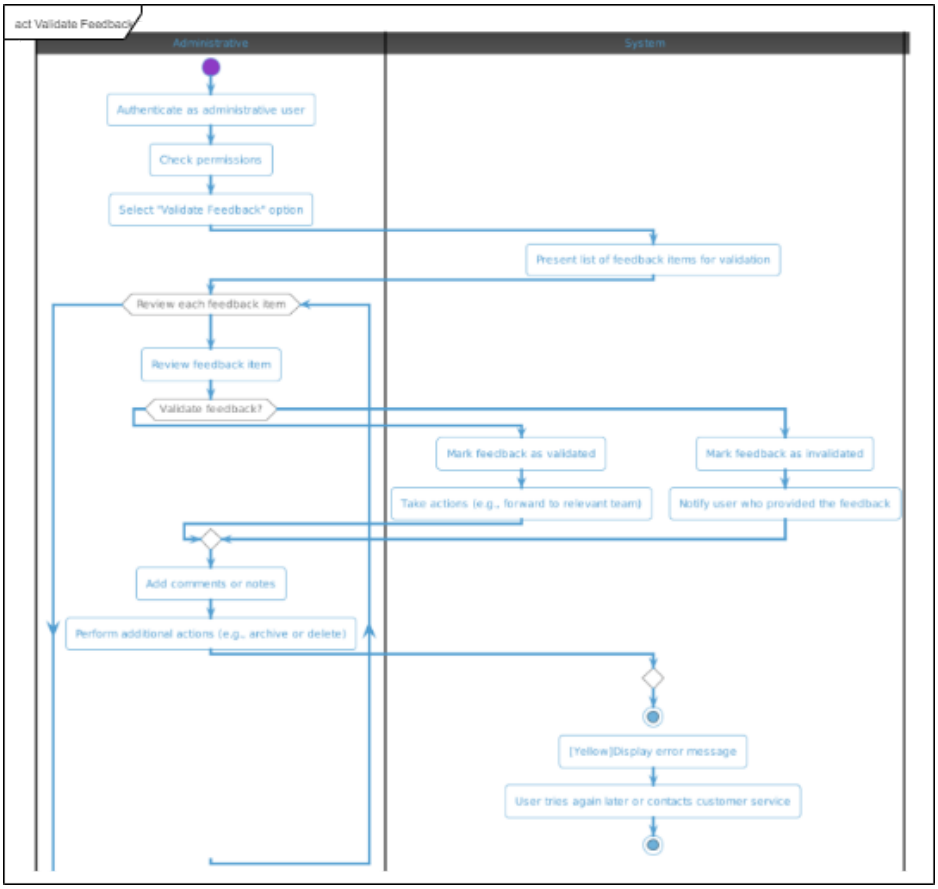

Figure 2.2.10.2: Activity Diagram for <Validate Feedback>

2.2.11 **UC011 : Use Case <Generate Report>**

Table 2.2.11: Use Case Description for Generate Report

Use case: Generate Report
Use Case ID: UC011
Use Case Name: Generate Report
Actors: 1. Administrative
Preconditions: 1. The user must be an administrative user with permissions to generate report
Flow of events: 1. The administrative user selects the “Generate Report” option from the system's menu 2. The administrative choose the date and type of report 3. The system stores generated report under “View Report”
Postconditions: 1. A report is saved under “View Report” menu
Alternative flow *n*: None
Exception flow: 1. Administrative encountered a technical error or system failure 1.1 The system displays an error message indicating the issue 1.2 Administrative need to try again later 1.3 The use case ends without any changes

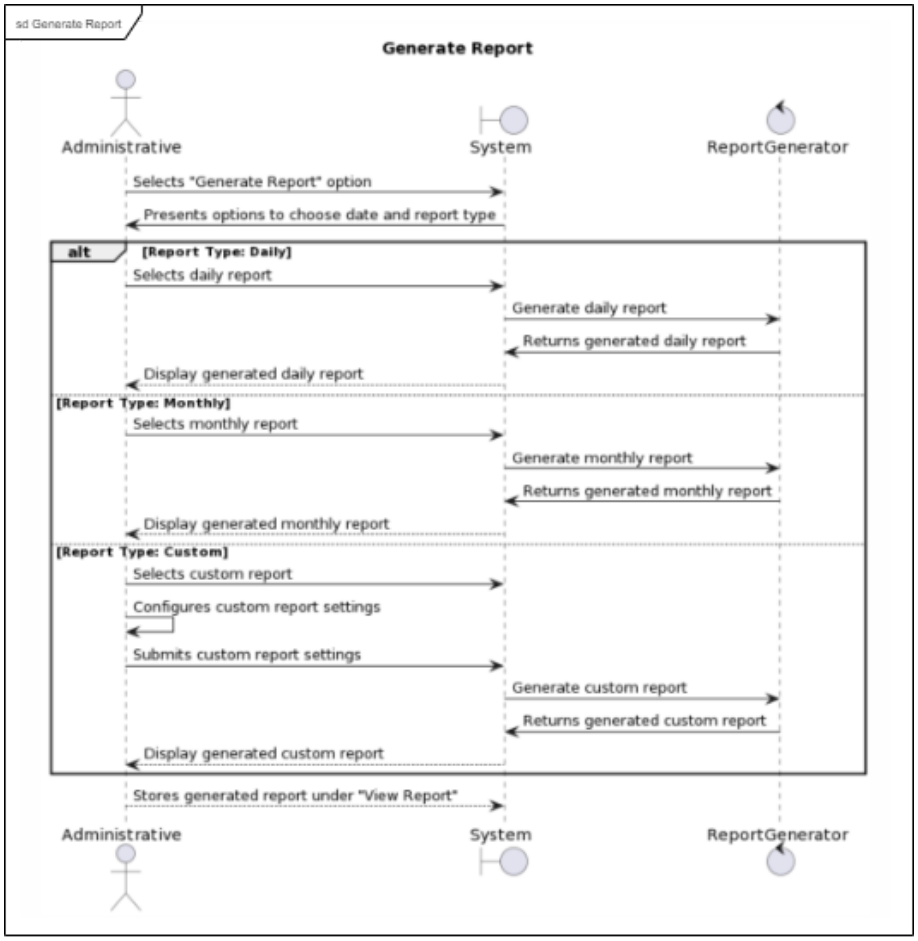

Figure 2.2.11.1: Sequence Diagram for <Generate Report>

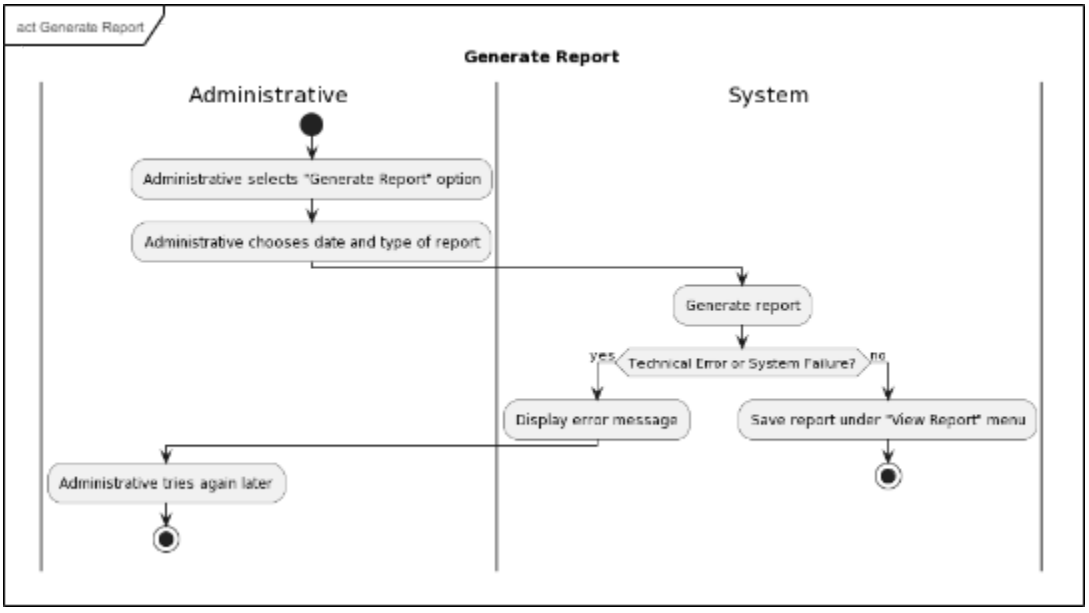

Figure 2.2.11.2: Activity Diagram for <Generate Report>

2.2.12 **UC012 : Use Case <View Report>**

Table 2.2.12: Use Case Description for View Report

Use case: View Report
Use Case ID: UC012
Use Case Name: View Report
Actors: 1. Administrative
Preconditions: 1. The user must be an administrative user with permissions to view report
Flow of events: 1. The administrative user selects the “View Report” option from the administration's menu 2. The administrative choose a report to be viewed 3. The system displays the report
Postconditions: N/A
Alternative flow *n*: N/A
Exception flow: 1. Administrative encountered a technical error or system failure 1.1 The system displays an error message indicating the issue 1.2 Administrative need to try again later 1.3 The use case ends without any changes

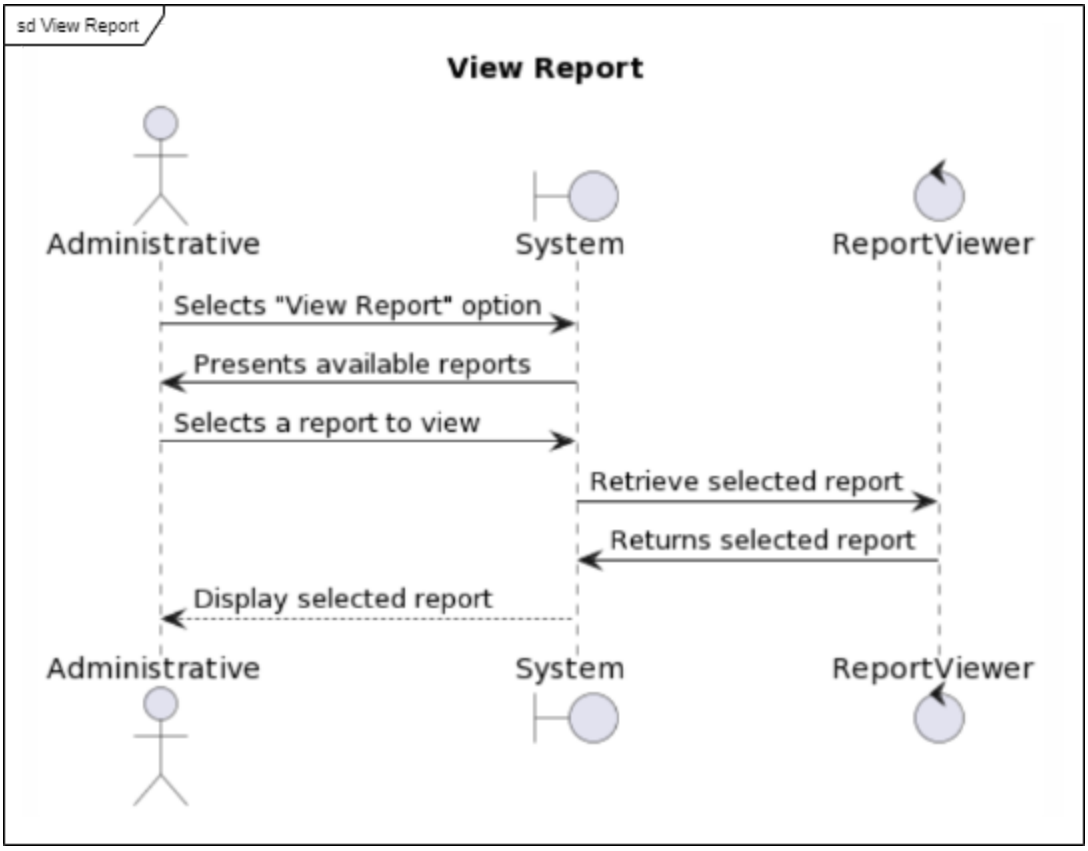

Figure 2.2.12.1: Sequence Diagram for <View Report>

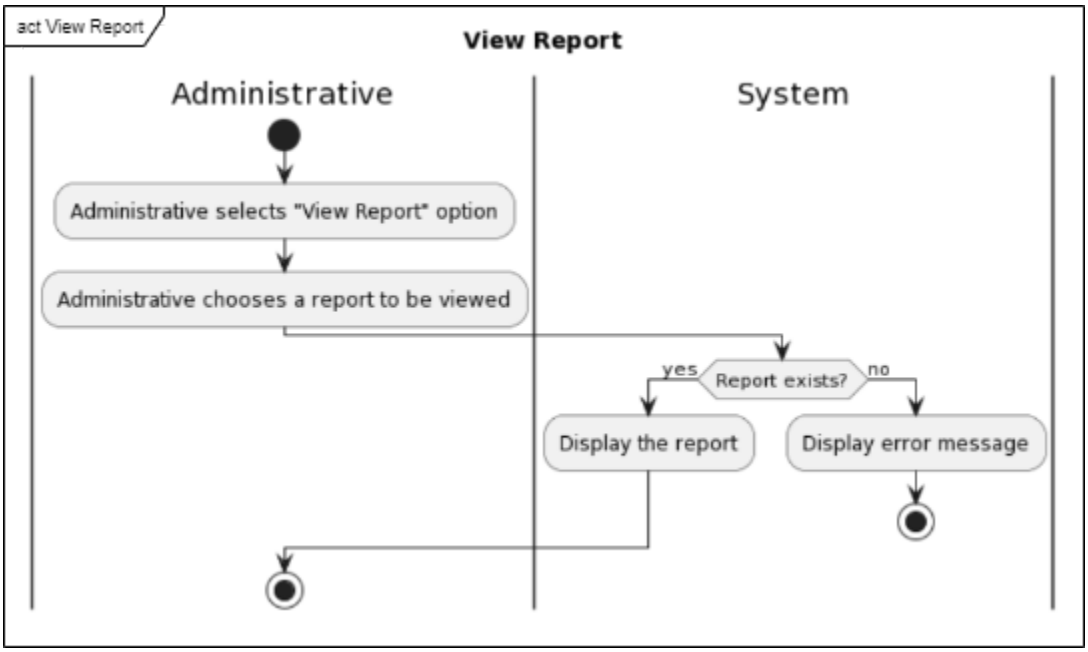

Figure 2.2.12.2: Activity Diagram for <View Report>

2.2.13 **UC013 : Use Case <Manage Report>**

Table 2.2.13: Use Case Description for Manage Report

Use case: Manage Report
Use Case ID: UC013
Use Case Name: Manage Report
Actors: 1. Administrative
Preconditions: 1. The user must be an administrative user with permissions to manage report
Flow of events: 1. The administrative user selects the “Manage Report” option from the administration's menu 2. The system display a list of reports previously generated 3. Administrative can take actions on reports 3.1 Administrative archive report from visible list 3.2 Administrative delete report from visible list 3.3 Administrative view details of the report
Postconditions: 1. Report is archived from list 2. Report is deleted from list
Alternative flow *n*: N/A
Exception flow: 1. Administrative encountered a technical error or system failure 1.1 The system displays an error message indicating the issue 1.2 Administrative need to try again later 1.3 The use case ends without any changes

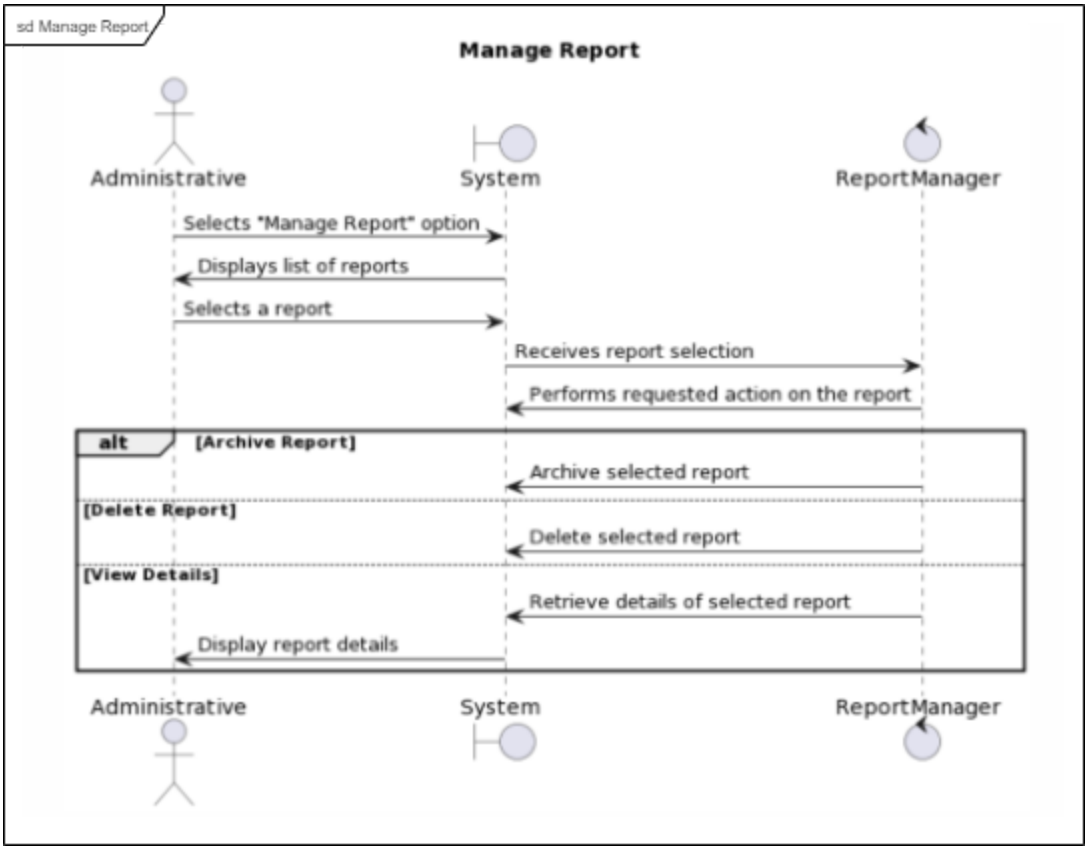

Figure 2.2.13.1: Sequence Diagram for <Manage Report>

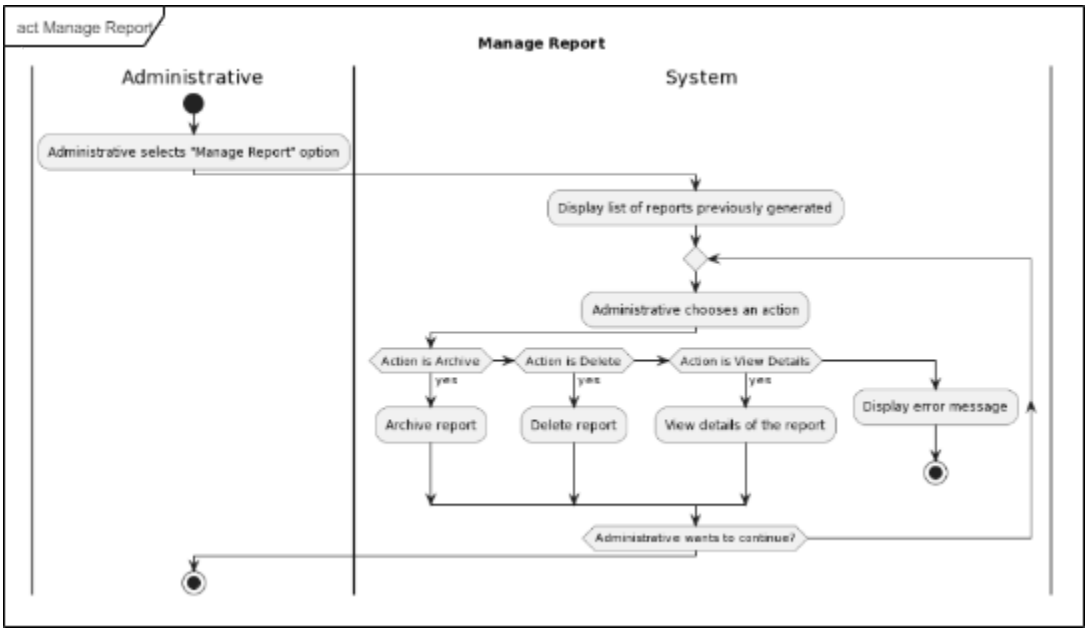

Figure 2.2.13.2: Activity Diagram for <Manage Report>

2.2.14 **UC014 : Use Case <Download Report>**

Table 2.2.14: Use Case Description for Download Report

Use case: Download Report
Use Case ID: UC014
Use Case Name: Download Report
Actors: 1. Administrative
Preconditions: 1. The user must be an administrative user with permissions to download report
Flow of events: 1. The administrative user selects the “Manage Report” option from the administration's menu 2. The system display a list of reports previously generated 3. Administrative choose report from the list 4. Administrative hits download button
Postconditions: 1. Report is downloaded and stored on administrative device
Alternative flow *n*: N/A
Exception flow: 1. Administrative encountered a technical error or system failure 1.1 The system displays an error message indicating the issue 1.2 Administrative need to try again later 1.3 The use case ends without any changes

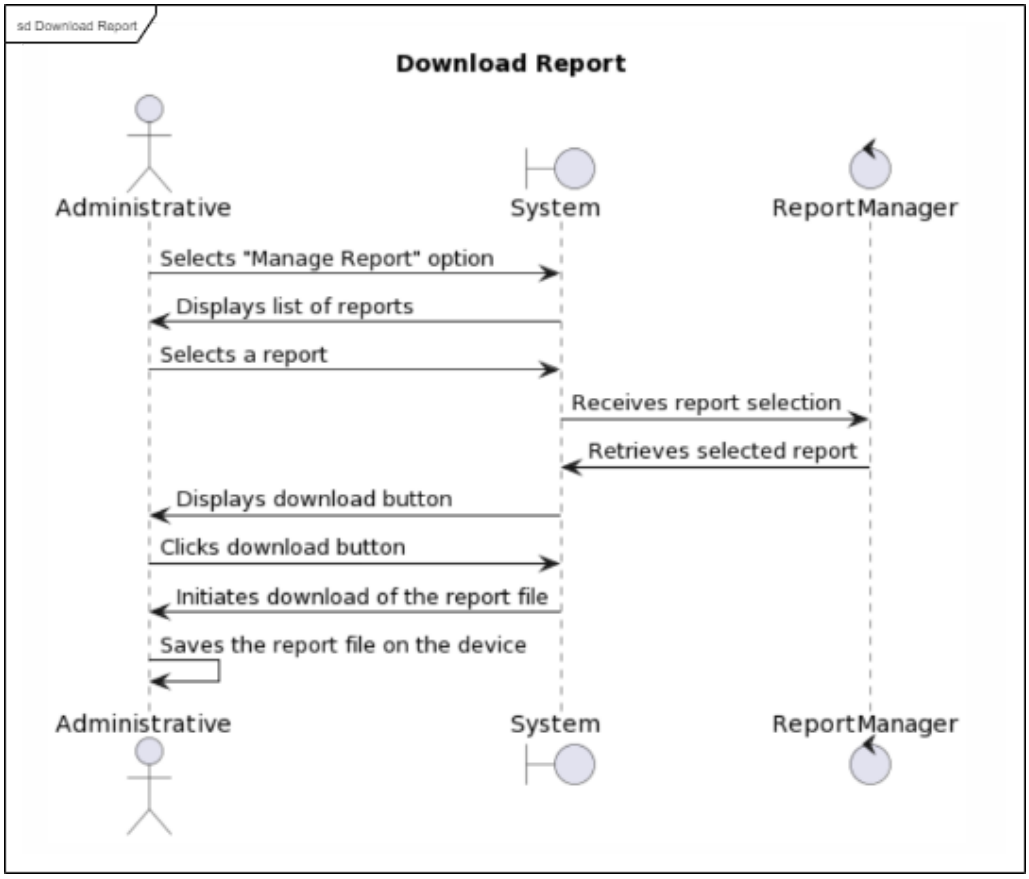

Figure 2.2.14.1: Sequence Diagram for <Download Report>

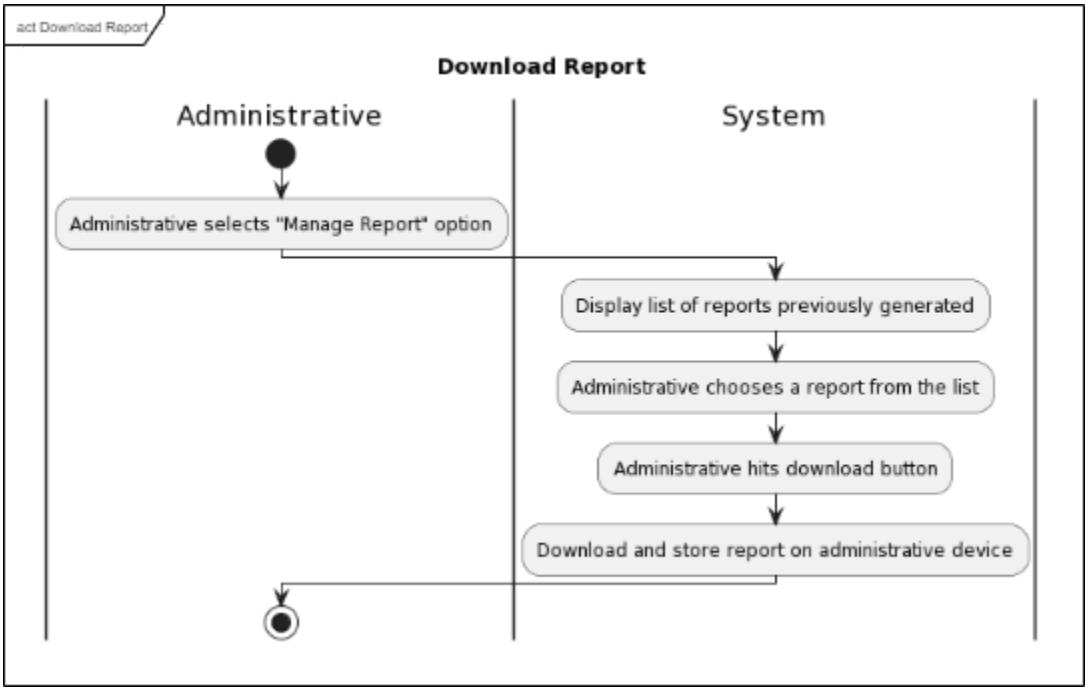

Figure 2.2.14.2: Activity Diagram for <Download Report>

2.2.15 **UC015 : Use Case <Make Post>**

Table 2.2.15: Use Case Description for Make Post

Use case: Make Post
Use Case ID: UC015
Use Case Name: Make Post
Actors: 1.Faculty Computing's Students 2.Faculty Computing's Lecturers
Preconditions: 1.Users have a personal account in the system. 2.Users can access the account's dashboard.
Flow of events: 1.Users login into their personal account. 2.Within the personalized dashboard, users select the create post option. 3.They can either post a text or any pictures or both and choose to add some tags relevant to their posts.. 4.After filling in what they wanted to post, they can just click the post button. 5.The post is successfully posted in their personal dashboard and others can see their posts.
Postconditions: 1.A new post is successfully added to the dashboard. 2.Users receive notifications on the successful adding of posts.
Alternative flow *n*: 1. If the post is not successfully posted, users will receive notifications for them to click on the repost button. 2. If there is no internet connection, the unsuccessfully posted post would save as a draft so that the content will not be missing and users can continue with any actions on the post such as reposting.
Postconditions: 1. A new post is added to the dashboard. 2. Users receive notifications on successfully posting their post.

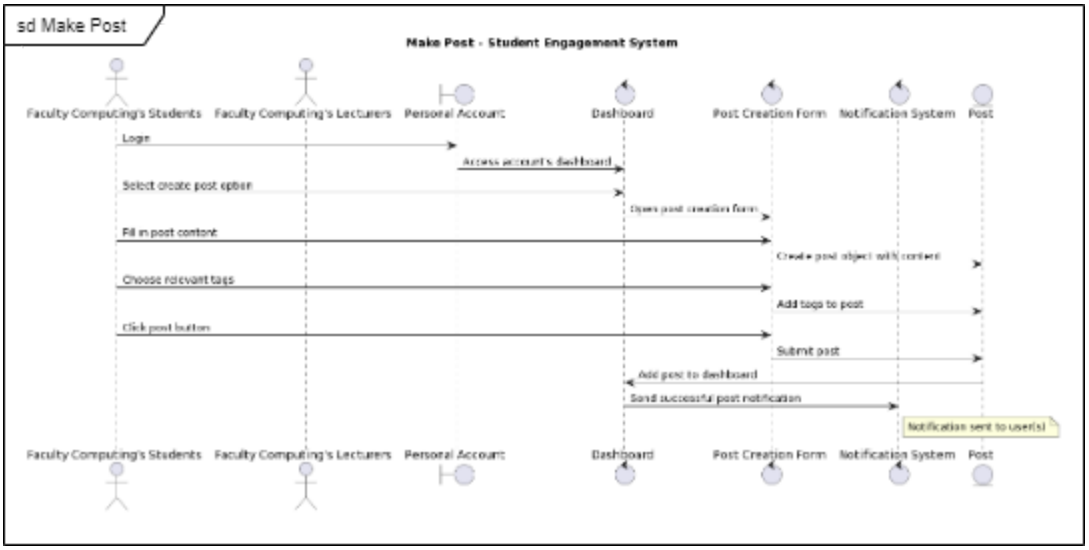

Figure 2.2.15.1: Sequence Diagram for <Make Post>

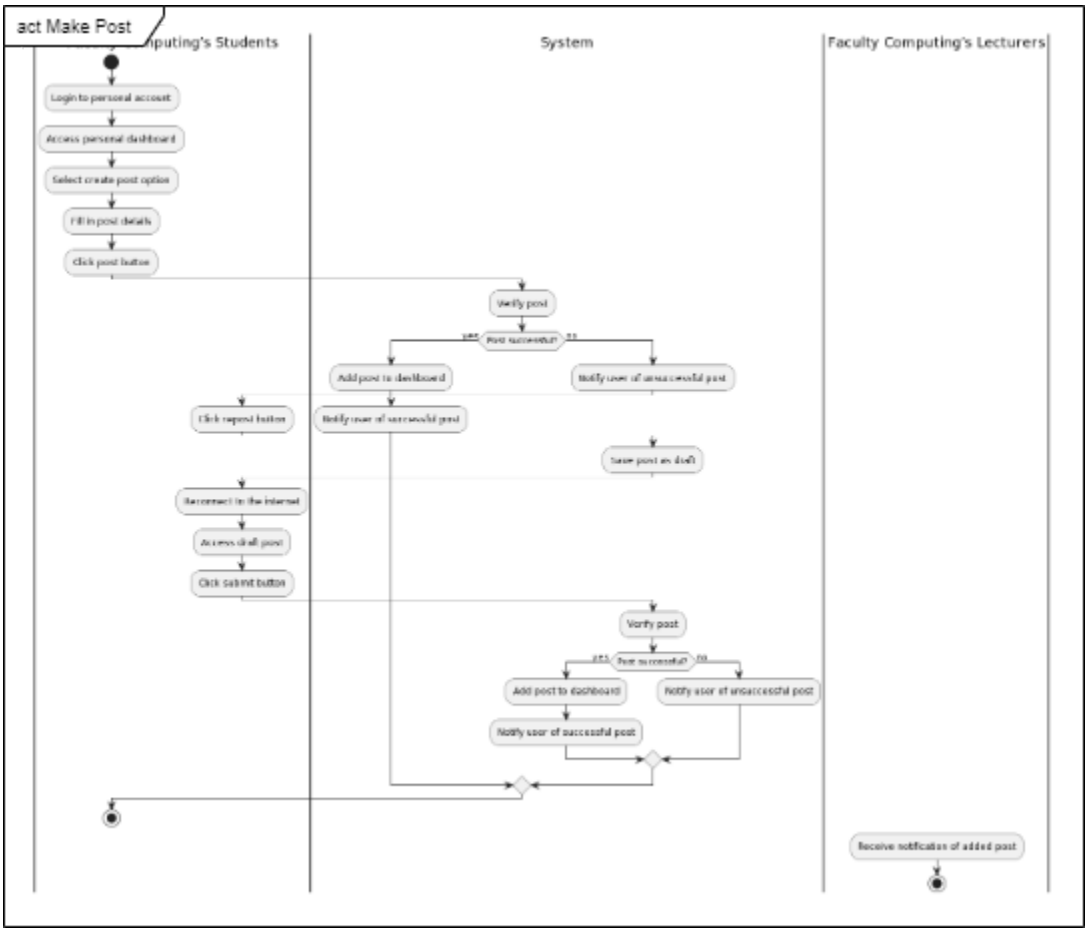

Figure 2.2.15.2: Activity Diagram for <Make Post>

2.2.16 **UC016 : Use Case <Edit Post>**

Table 2.2.16: Use Case Description for Edit Post

Use case: Edit Post
Use Case ID: UC016
Use Case Name: Edit Post
Actors: 1.Faculty Computing's Students 2.Faculty Computing's Lecturers
Preconditions: 1. Users should make sure that they have a stable internet. 2. Users should make sure that the post that they wish to edit is available.
Flow of events: 1. Users log into their personal account of the system. 2. Users access their personal dashboard. 3. Users select the post that they wish to edit. 4. Users edit the existing post by pressing the editing button. 5. Other people will see the updated posts.
Postconditions: 1. The edited post is successfully updated. 2. Users receive a notification indicating that their post has been updated.
Alternative flow *n*: 1. If there is no internet connection, the edited post will not be updated but their edited post will be saved as a draft. 2. When they reconnect to the internet, they can still access the updated post and press the submit button to update their posts.
Postconditions: 1. The edited post is successfully updated. 2. Users receive a notification indicating that their post has been updated.

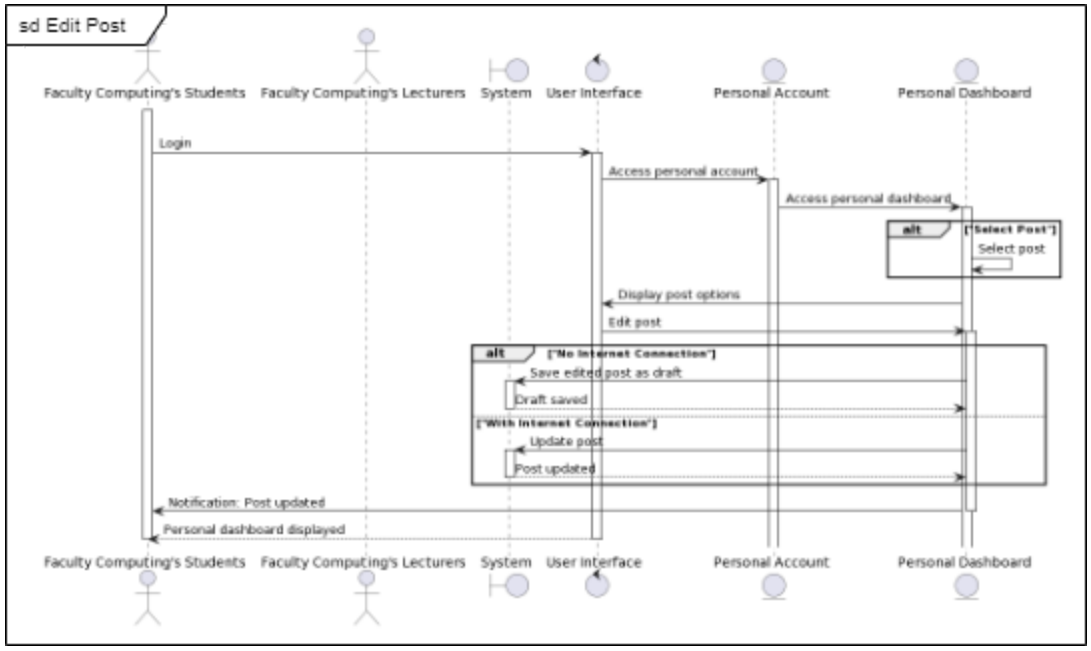

Figure 2.2.16.1: Sequence Diagram for <Edit Post>

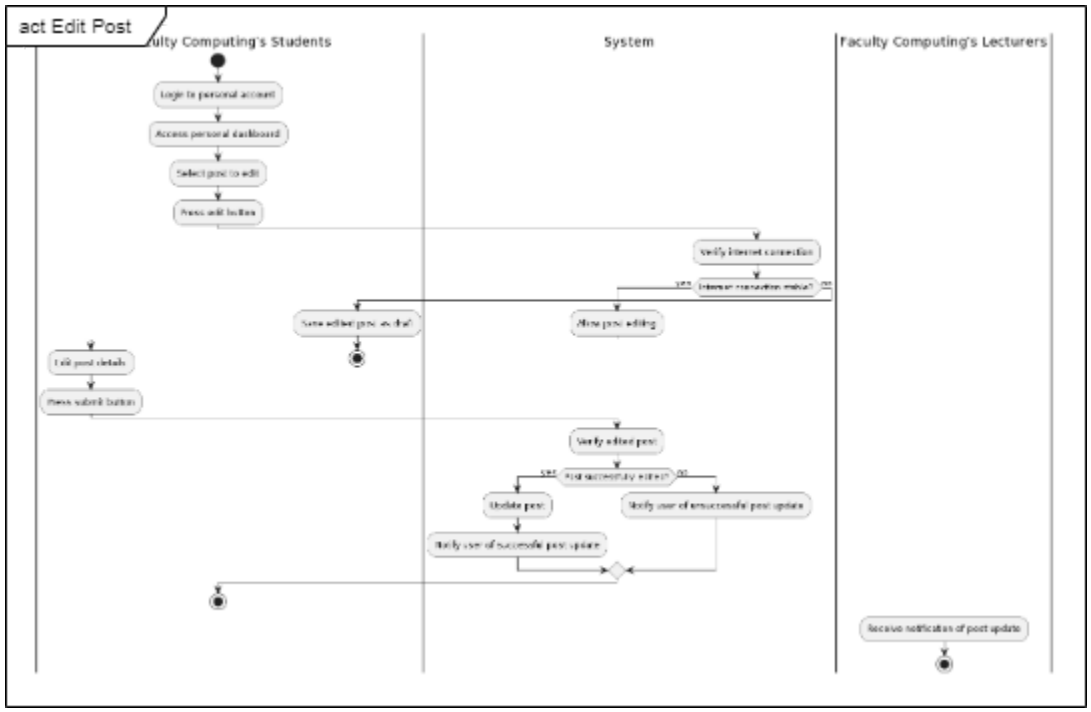

Figure 2.2.16.2: Activity Diagram for <Edit Post>

2.2.17 **UC017 : Use Case <Delete Post>**

Table 2.2.17: Use Case Description for Delete Post

Use case: Delete Post
Use Case ID: UC017
Use Case Name: Delete Post
Actors: 1. Faculty Computing's Students 2. Faculty Computing's Lecturers
Preconditions: 1. Users should make sure that they have a stable internet. 2. Users should make sure that the post that they wish to delete is available.
Flow of events: 1. Users log into their personal account of the system. 2. Users access their personal dashboard. 3. Users select the post that they wish to delete. 4. Users delete the existing post by pressing the delete button. 5. Other people will not see the deleted posts.
Postconditions: 1. The post is successfully deleted. 2. Users receive a notification indicating that their post has been successfully deleted.
Alternative flow *n*: 1. If there is no internet connection, the post will not be deleted until it reconnects to the internet. 2. When they reconnect to the internet, they can still delete the post that they wanted to delete.
Postconditions: 1. The post is successfully deleted. 2. Users receive a notification indicating that their post has been successfully deleted.

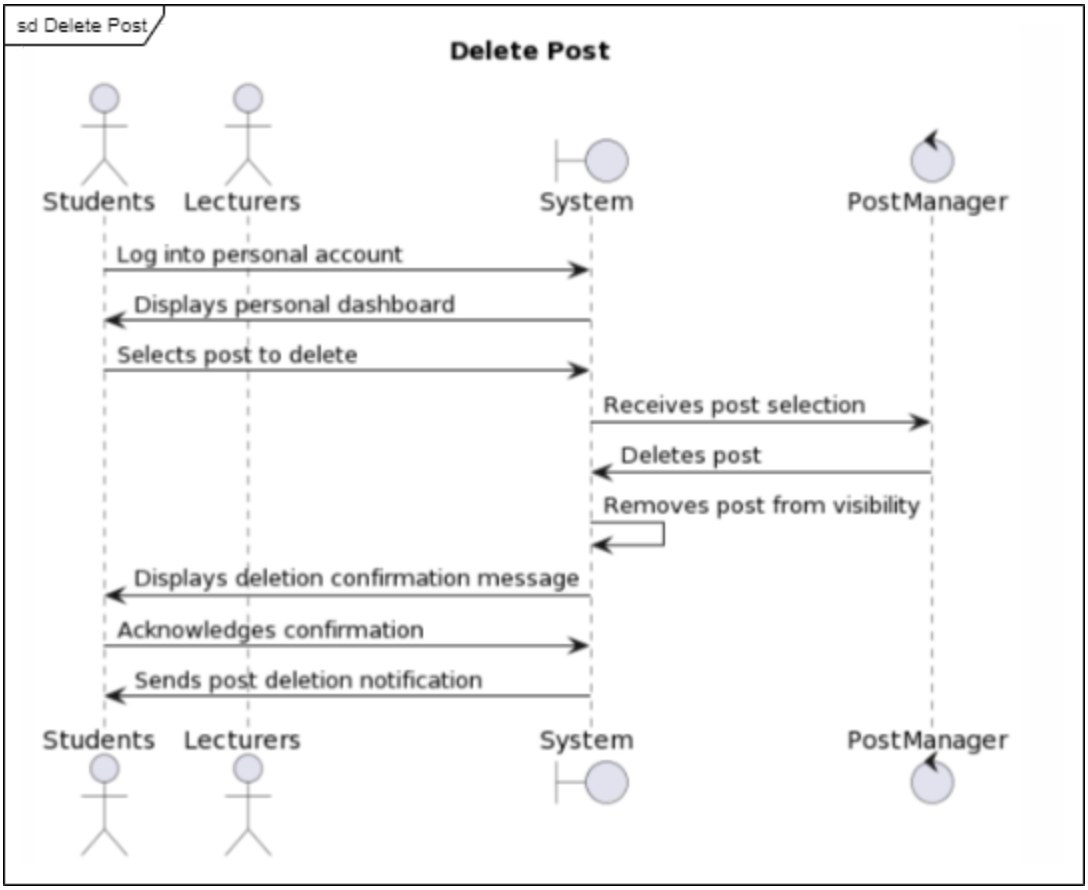

Figure 2.2.17.1: Sequence Diagram for <Delete Post>

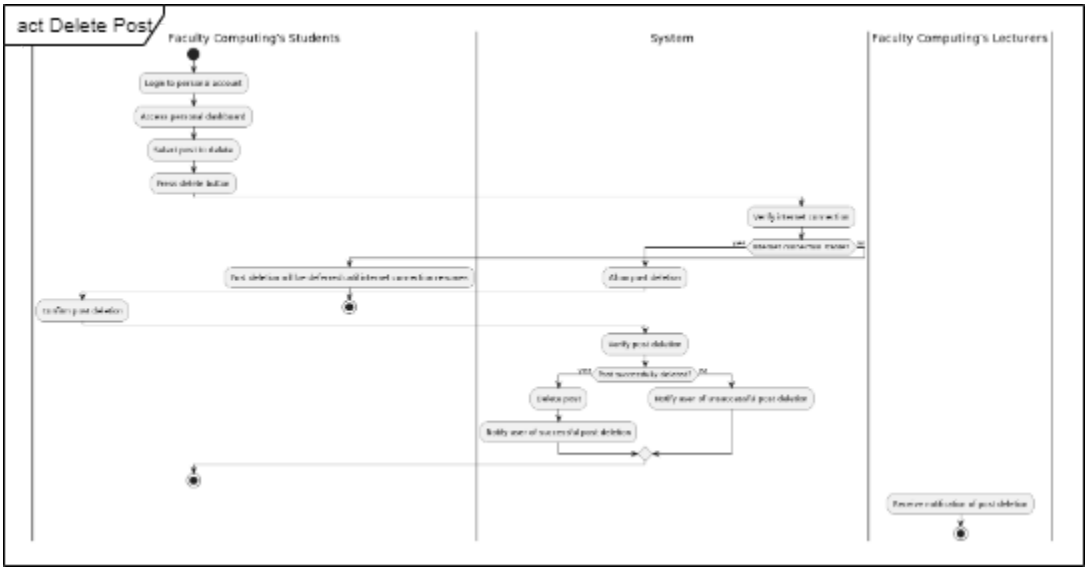

Figure 2.2.17.1: Activity Diagram for <Delete Post>

2.2.18 **UC018 : Use Case <View Feedback>**

Table 2.2.18: Use Case Description for View Feedback

Use case: View Feedback
Use Case ID: UC018
Use Case Name: View Feedback
Actors: 1. Student Representative Council(SRC)
Preconditions: 1. SRC should ensure that they have a stable internet in order to access the feedback module. 2. They should ensure that they have accounts.
Flow of events: 1. SRC log into their personal account. 2. They access the feedback section. 3. They view the posted feedback by pressing the viewing button.
Postconditions: 1. A list of feedback will be shown and they can perform any further actions.
Alternative flow *n*: 1. If there are no internet connections, they cannot load the list of feedback until they reconnect to the internet. 2. They can view the list of feedback by pressing the viewing button.
Postconditions: 1. A list of feedback will be shown and they can perform any further actions.

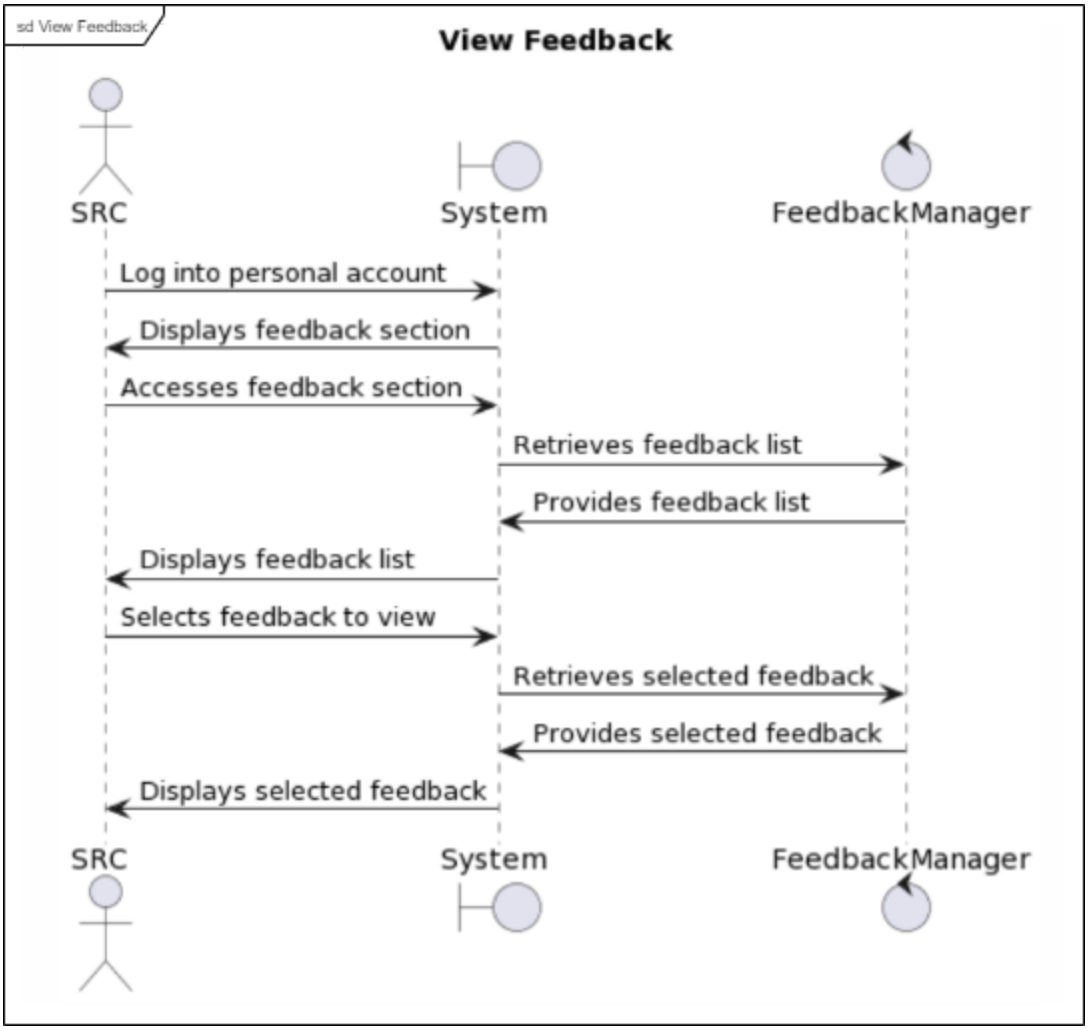

Figure 2.2.18.1: Sequence Diagram for <View Feedback>

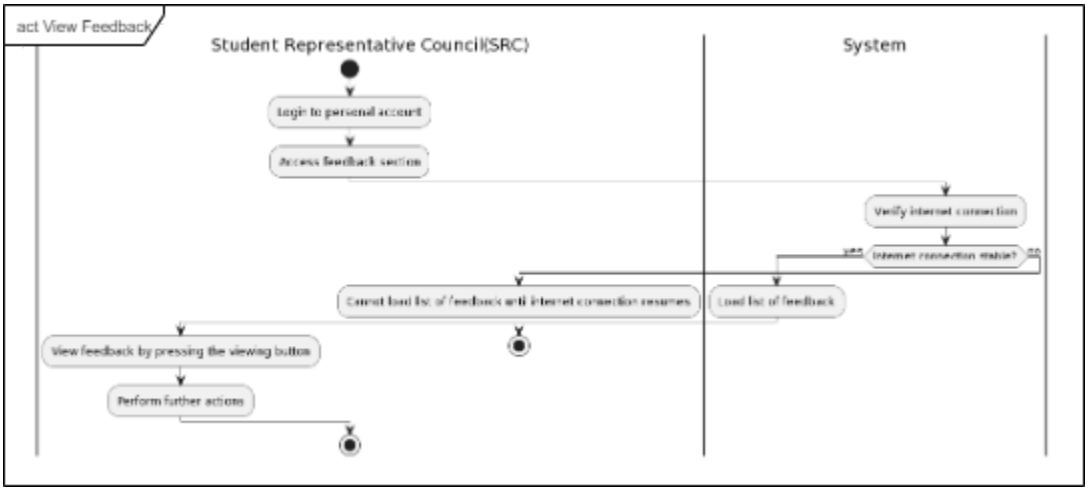

Figure 2.2.18.2: Activity Diagram for <View Feedback>

2.2.19 **UC019 : Use Case <Submit Feedback>**

Table 2.2.19: Use Case Description for Submit Feedback

Use case: Submit Feedback
Use Case ID: UC019
Use Case Name: Submit Feedback
Actors: 1. Faculty Computing's Students 2. Faculty Computing's Lecturers
Preconditions: 1. Users should ensure that they have a stable internet in order to access the feedback module. 2. Users should ensure that they have accounts.
Flow of events: 1. Users log into their accounts. 2. Users access to the feedback section by pressing the feedback button at the tab. 3. Users can write any complaints or feedback in the form provided. 4. After checking, they can submit the feedback by pressing the submit button.
Postconditions: 1. The feedback has been successfully submitted. 2. Users will receive notifications on successfully posting feedback. 3. The feedback will later be viewed by SRC.
Alternative flow *n*: 1. If there are no internet connections, the written feedback will be saved as draft. 2. They can access to the feedback again when there are internet connections. 3. After checking, they can submit the feedback by pressing the submit button.
Postconditions: 1. The feedback has been successfully submitted. 2. Users will receive notifications on successfully posting feedback. 3. The feedback will later be viewed by SRC.

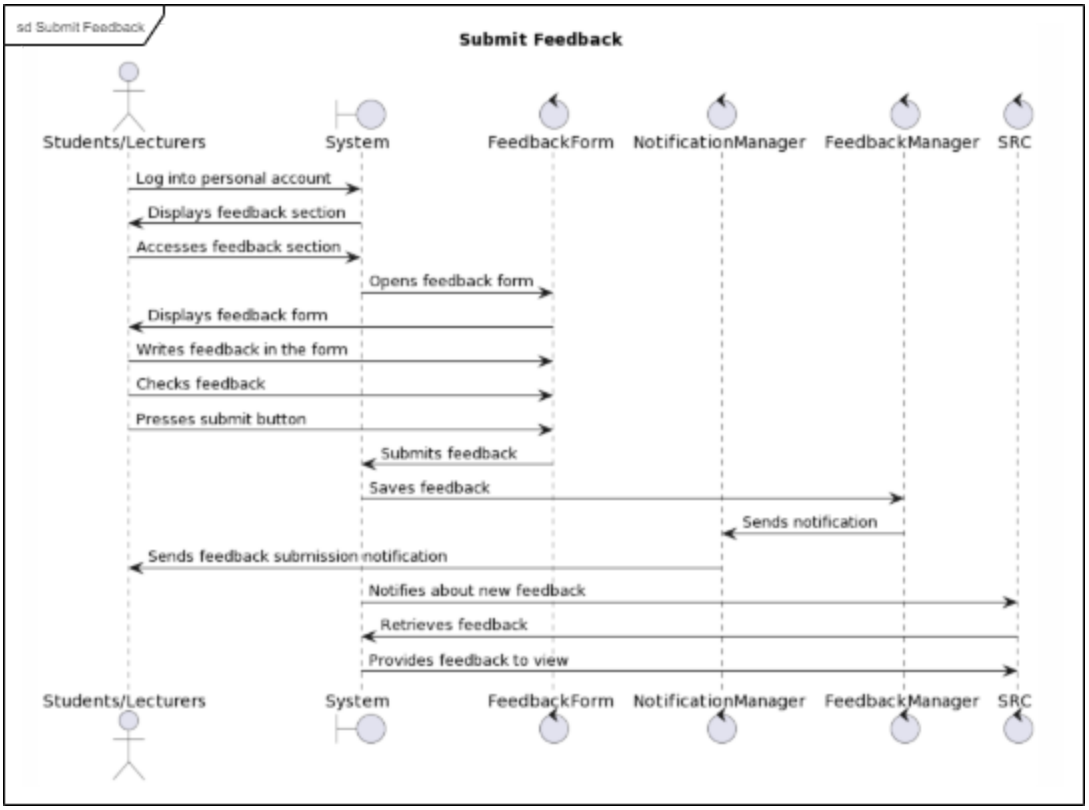

Figure 2.2.19.1: Sequence Diagram for <Submit Feedback>

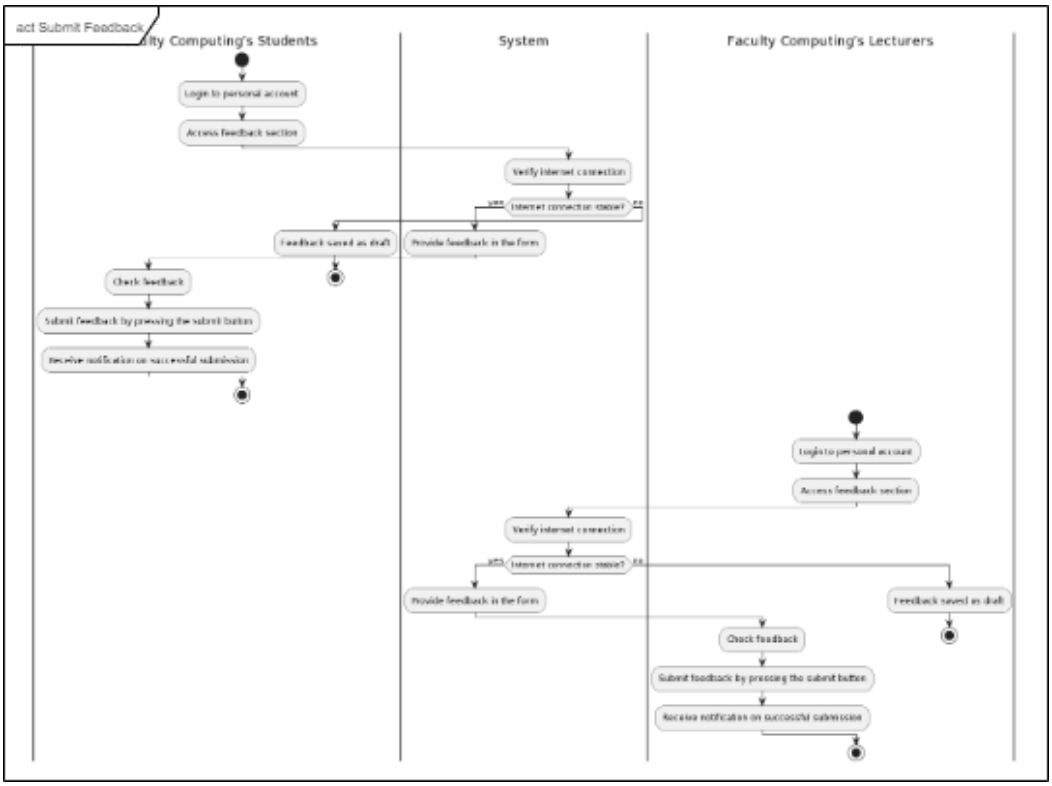

Figure 2.2.19.2: Activity Diagram for <Submit Feedback>

2.2.20 **UC020 : Use Case <Manage Feedback>**

Table 2.2.20: Use Case Description for Manage Feedback

Use case: Manage Feedback
Use Case ID: UC020
Use Case Name: Manage Feedback
Actors: 1. Admins
Preconditions: 1. Admins should ensure that they have a stable internet in order to access the feedback module. 2. Admins should ensure that they have accounts.
Flow of events: 1. Admins log into their own account. 2. They can manage the feedback by filtering them and leaving the effective one. 3. They delete any feedback that is not important. 4. They arrange the feedback based on the priority and put the most important at the top. 5. They commit the changes by pressing the commit button.
Postconditions: 1. Admins will receive a notification indicating the feedback has been successfully filtered. 2. SRC can view the filtered feedback and perform any actions based on the priority.
Alternative flow *n*: 1. If there are no internet connections, the feedback cannot be filtered until they reconnect to the internet. 2. When they get back to the internet, they can still filter the feedback and commit changes so that the filtered feedback can be viewed by SRC.
Postconditions: 1. Admins will receive a notification indicating the feedback has been successfully filtered. 2. SRC can view the filtered feedback and perform any actions based on the priority.

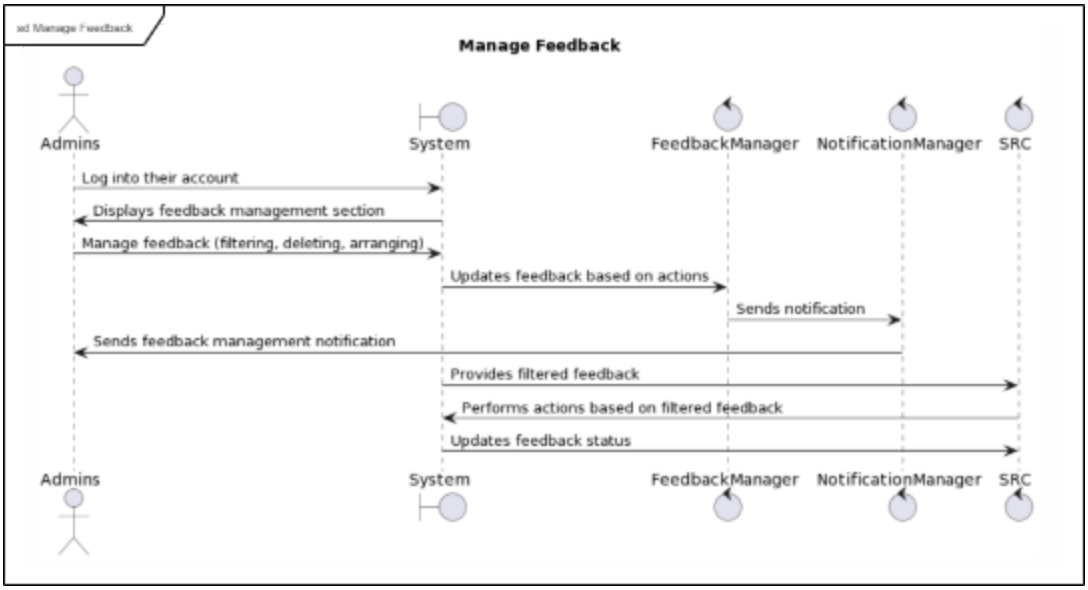

Figure 2.2.20.1: Sequence Diagram for <Manage Feedback>

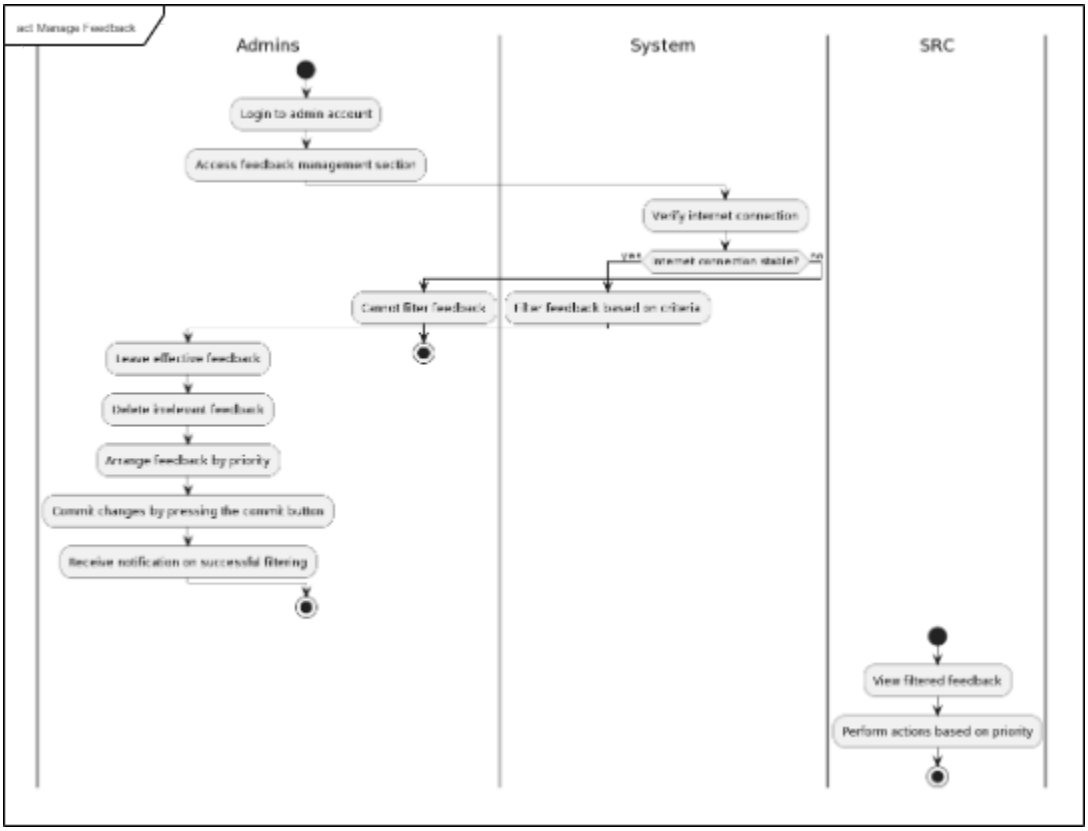

Figure 2.2.20.2: Activity Diagram for <Manage Feedback>

2.2.21 **UC021 : Use Case <Create Forum>**

Table 2.2.21: Use Case Description for Create Forum

Use case: Create Forum
Use Case ID: UC021
Use Case Name: Create Forum
Actors: 1. Faculty Computing's Students 2. Faculty Computing's Lecturers
Preconditions: 1. Has an active student engagement system account 2. User is within a stable internet environment.
Flow of events: 1. The use case starts when a user, either a Faculty Computing's Student or Faculty Computing's Lecturer, initiates the process of creating a new forum. 2. The user provides the necessary information for the new forum, such as the forum name, description, and any additional settings or preferences. 3. The system validates the provided information to ensure it meets any specified requirements or constraints (e.g., unique forum name, length restrictions). 4. If the provided information is valid, the system creates the new forum and assigns it a unique identifier. 5. The system confirms the successful creation of the forum and notifies the user about the forum's creation.
Postconditions: 1. The new forum was successfully created and added to the system. 2. The forum is assigned a unique identifier. 3. The user is notified of the successful creation of the forum. 4. The forum information, including the forum name, description, and any additional settings or preferences, is accurately stored in the system. 5. The forum is ready for user interaction, such as posting content and engaging in discussions. 6. The system is in a state where it can handle further actions and operations related to the newly created forum.
Alternative flow *n*: 1.If the provided information is valid and does not trigger any moderation rules: ● The new forum is successfully created and added to the system. ● The forum is assigned a unique identifier. ● The user is notified of the successful creation of the forum. ● The forum information, including the forum name, description, and any additional settings or preferences, is accurately stored in the system. ● The forum is ready for user interaction, such as posting content and engaging in discussions.
Postconditions:

1. The new forum is successfully created and added to the system.
2. The forum is assigned a unique identifier.
3. The user is notified of the successful creation of the forum.
4. The forum information, including the forum name, description, and any additional settings or preferences, is accurately stored in the system.
5. The forum is in an active state, ready for user interaction.
6. Users can participate in the forum by posting content, commenting, and engaging in discussions.
7. The system is in a state where it can handle further actions and operations related to the created forum, such as managing forum settings and moderating user activities.

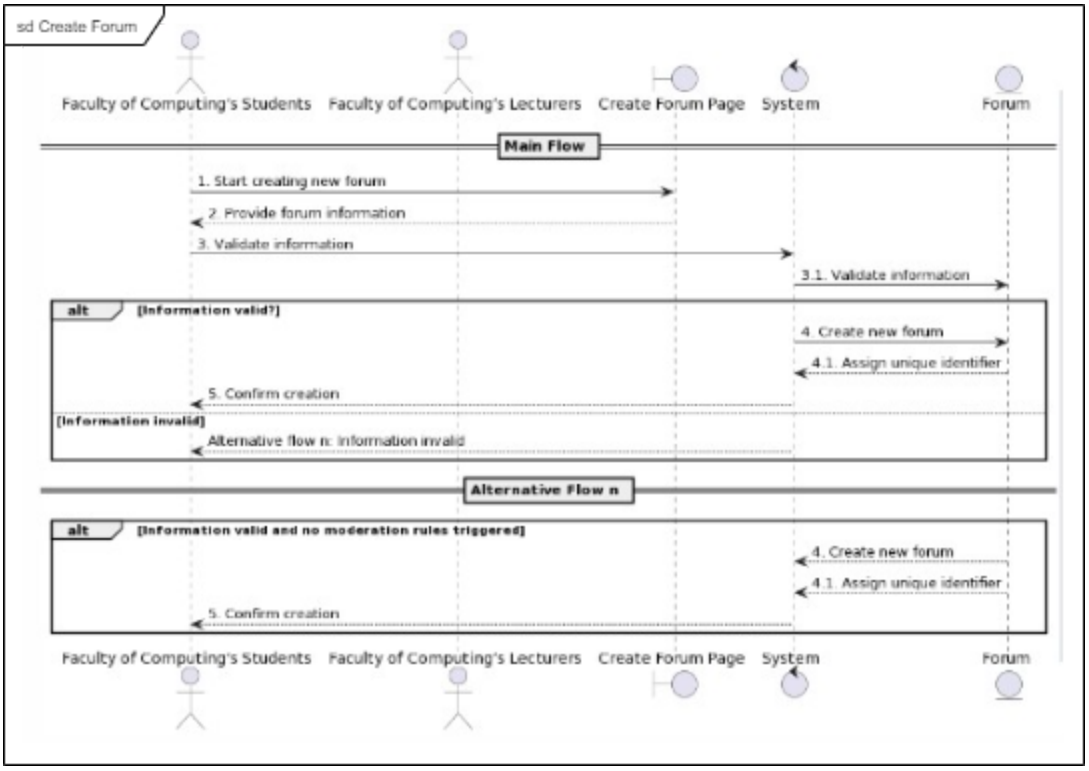

Figure 2.2.21.1: Sequence Diagram for <Create Forum>

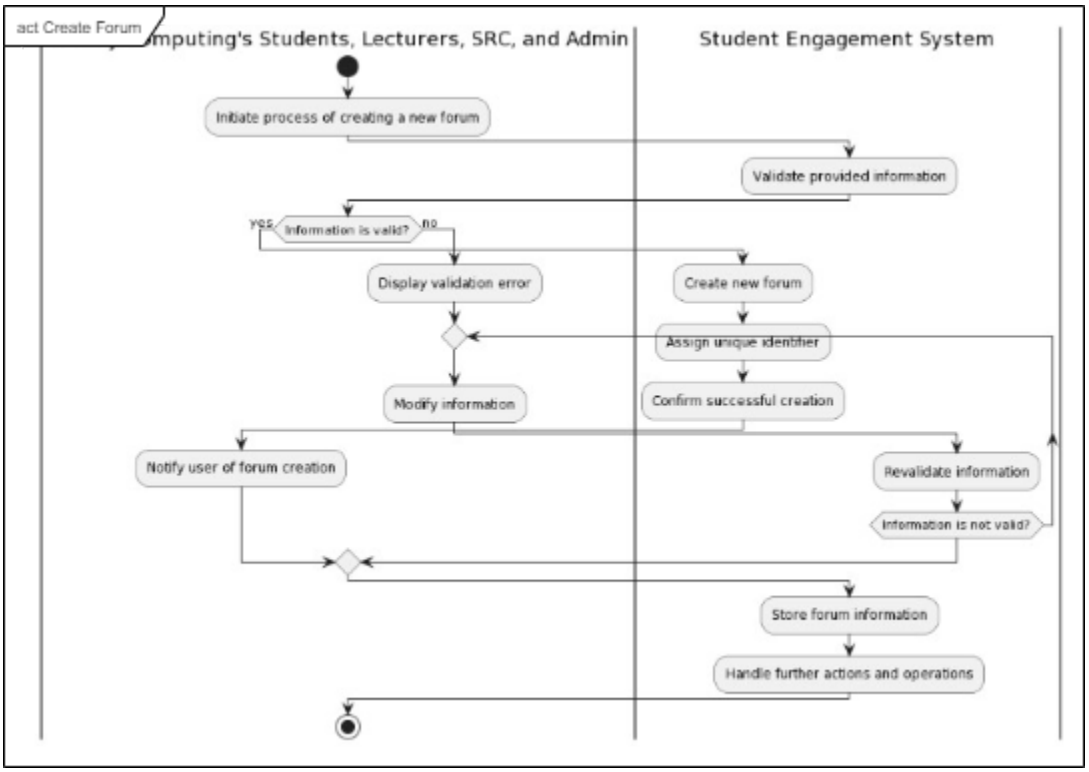

Figure 2.2.21.2: Activity Diagram for <Create Forum>

2.2.22 **UC022 : Use Case <Forum Comments>**

Table 2.2.22: Use Case Description for Forum Comments

Use case: Forum Comments
Use Case ID: UC022
Use Case Name: Forum Comments
Actors: 1. Faculty Computing's Students 2. Faculty Computing's Lecturers
Preconditions: 1. Has an active student engagement system account 2. User is within a stable internet environment.
Flow of events: 1. The use case starts when a user wants to add a comment to a forum and post. 2. The user selects the specific forum and post to which they want to add a comment. 3. The system retrieves the selected forum and post and displays it along with existing comments. 4. The user enters their comment in the designated input field. 5. The system validates the comment to ensure it meets any specified requirements or constraints (e.g., length restrictions). 6. If the comment passes validation, the system creates a new comment and assigns it a unique identifier. 7. The system associates the comment with the selected forum and post, linking it to the appropriate context. 8. The system stores the comment content and metadata, such as the commenter's username and timestamp, in the database. 9. The system confirms the successful creation of the comment and notifies the user about the comment's addition. 10. The comment becomes visible to other users within the forum and post, displaying alongside existing comments. 11. Other users can interact with the comment by replying, liking, or reporting it. 12. The user can view the posted comment and any associated interactions.
Postconditions: 1. The user's comment is successfully created and added to the selected forum and post. 2. The comment is assigned a unique identifier. 3. The comment content, including text and any associated metadata (e.g., commenter's username, timestamp), is accurately stored in the system's database. 4. The comment is associated with the selected forum and post, ensuring it appears within the appropriate context. 5. The system confirms the successful creation of the comment and notifies the user about the comment's addition. 6. The comment becomes visible to other users who can view, interact with, and respond to it within the forum and post.
Alternative flow *n*:

1. After the user enters their comment, the system validates the comment to ensure it meets any specified requirements or constraints. 2. If the comment fails validation (e.g., exceeds length restrictions), the system displays an error message indicating the specific validation failure. 3. The user is notified of the validation failure and provided with instructions on how to resolve the error. 4. The user can choose to either modify the comment to meet the requirements or cancel the comment creation process. 5. If the user decides to modify the comment, they can make the necessary changes and reattempt the comment creation process from step 4 of the main flow. 6. If the user chooses to cancel the comment creation process, they can return to the previous state without creating the comment.
Postconditions: 1. The system displays an error message indicating the specific validation failure. 2. The user is notified of the validation error and provided with instructions on how to resolve it. 3. The user has the option to modify the comment to meet the specified requirements or cancel the comment creation process. 4. If the user chooses to modify the comment: ● They can make the necessary changes to address the validation error. ● The user can reattempt the comment creation process from the step where they encountered the validation error. 5. If the user chooses to cancel the comment creation process: ● The system returns the user to the previous state without creating the comment. 6. The system remains in a state where it can handle further actions and operations related to forum comments.

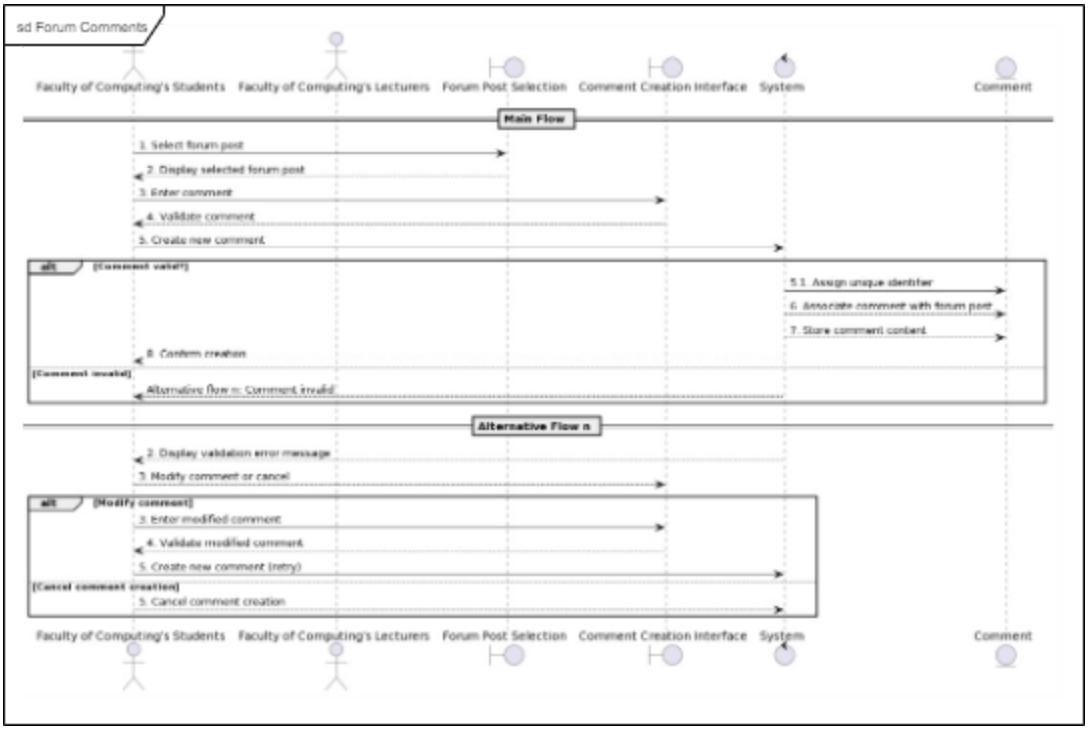

Figure 2.2.22.1: Sequence Diagram for <Forum Comments>

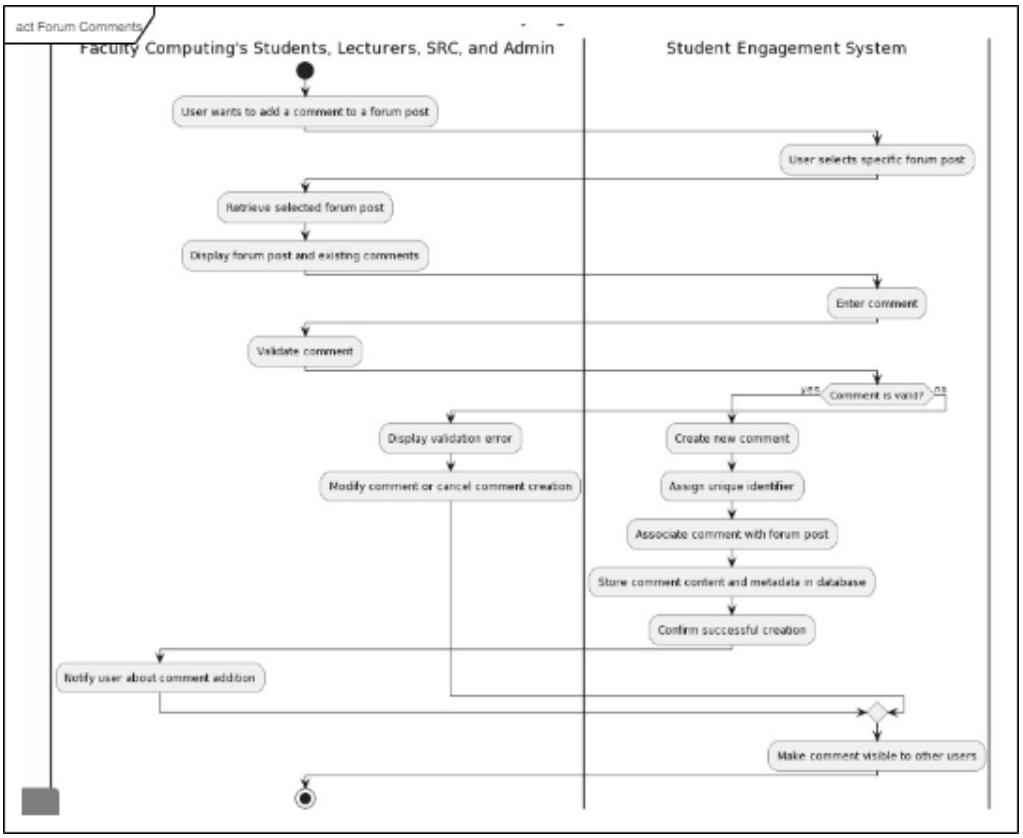

Figure 2.2.22.2: Activity Diagram for <Forum Comments>

2.2.23 **UC023 : Use Case <Edit Forum>**

Table 2.2.23: Use Case Description for Edit Forum

Use case:Edit Forum
Use Case ID: UC023
Use Case Name: Edit Forum
Actors: 1. Faculty Computing's Students 2. Faculty Computing's Lecturers
Preconditions: 1. Has an active student engagement system account 2. User is within a stable internet environment.
Flow of events: 1. The use case starts when a user wants to edit their existing descriptions in the forum. 2. The user selects the specific forum descriptions they want to edit. 3. The system retrieves the selected forum descriptions and presents the user with the editing interface. 4. The user modifies the content of the forum descriptions, making the necessary changes. 5. The system validates the edited forum descriptions to ensure it meets any specified requirements or constraints. 6. If the edited forum descriptions pass validation, the system updates the forum descriptions with the new content. 7. The system stores the updated forum descriptions content in the database. 8. The system confirms the successful editing of the forum descriptions and notifies the user about the update. 9. The updated forum descriptions are displayed to other users, reflecting the changes made by the user. 10. Other users can view the edited forum descriptions and any associated interactions.
Postconditions: 1. The user's forum descriptions are successfully edited with the updated forum descriptions. 2. The edited forum description content is accurately stored in the system's database, replacing the previous forum description. 3. The system confirms the successful editing of the forum descriptions and notifies the user about the update. 4. The edited forum descriptions are displayed to other users within the forum, reflecting the changes made by the user. 5. Other users can view the edited forum descriptions and any associated interactions. 6. The system remains in a state where it can handle further actions and operations related to forum descriptions editing.

Alternative flow *n*:

1. After the user modifies the content of the forum descriptions, the system validates the edited content to ensure it meets any specified requirements or constraints.
2. If the edited forum description fails validation (e.g., exceeds length restrictions, violates moderation rules), the system displays an error message indicating the specific validation failure.
3. The user is notified of the validation failure and provided with instructions on how to resolve the error.
4. The user can choose to either modify the forum description content to meet the requirements or cancel the editing process.
5. If the user decides to modify the post content, they can make the necessary changes and reattempt the editing process from step 4 of the main flow.
6. If the user chooses to cancel the editing process, the system retains the previous content of the forum descriptions without any modifications.

Postconditions:

1. The system displays an error message indicating the specific validation failure or restriction.
2. The user is notified of the validation error or restriction and provided with instructions on how to resolve it.
3. The user has the option to modify the post content to address the validation error or restriction or cancel the editing process.
4. If the user chooses to modify the forum description content:
   - They can make the necessary changes to address the validation error or restriction.
   - The user can reattempt the editing process from the step where they encountered the error or restriction.
5. If the user chooses to cancel the editing process:
   - The system retains the previous content of the forum descriptions without any modifications.
6. The system remains in a state where it can handle further actions and operations related to forum descriptions editing.

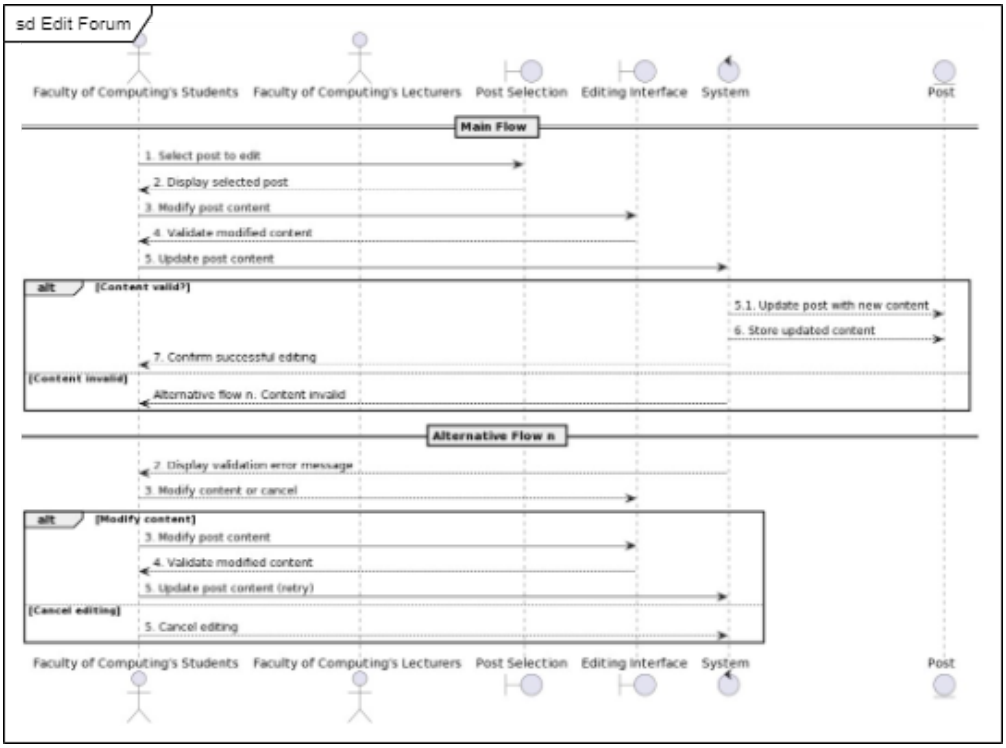

Figure 2.2.23.1: Sequence Diagram for <Edit Forum>

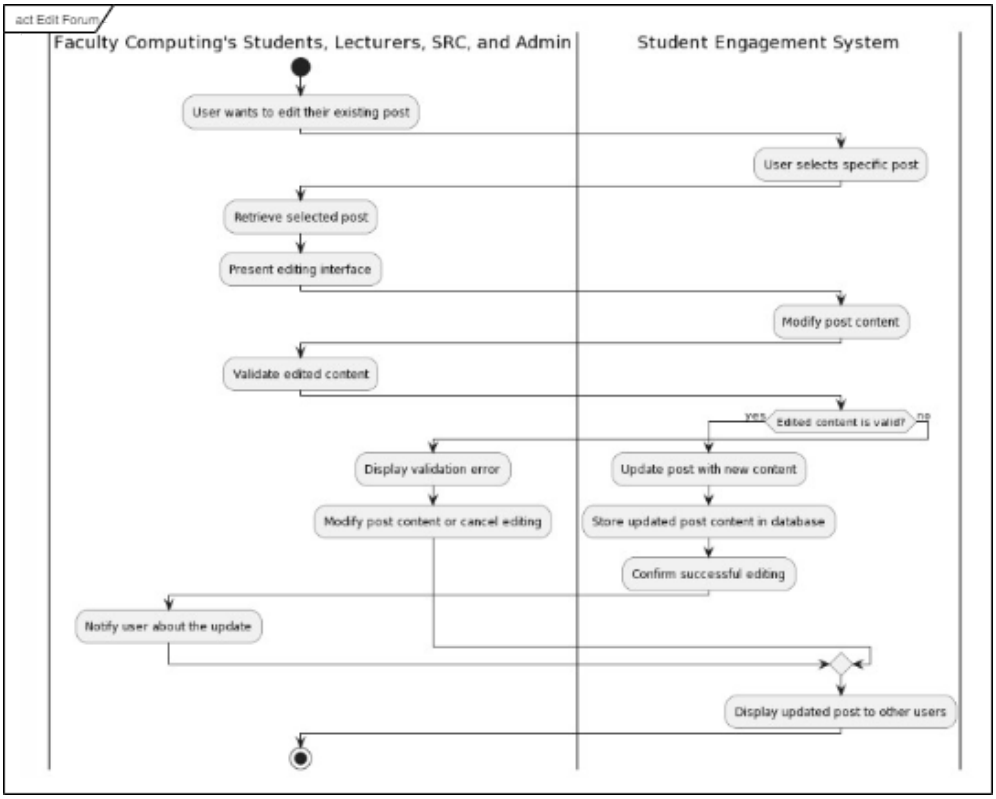

Figure 2.2.23.2: Activity Diagram for <Edit Forum>

2.3 **Software System Attributes, Performance and Other Requirements**

Software System Attributes:

1. Usability
    - The system should have a user-friendly interface that is intuitive and easy to navigate. It should provide a seamless user experience for students, faculty members, and administrators.
2. Reliability
    - The system should be reliable, ensuring that it is available and accessible to users when needed. It should have mechanisms in place to handle failures, errors, and minimize downtime.
3. Maintainability
    - The system should be designed for ease of maintenance and support. It should have proper documentation, error logging, and monitoring capabilities to facilitate troubleshooting and debugging.
4. Portability
    - The system will run on different platforms such as laptops, smartphones, tablets, and desktops with different operating systems.
5. Compatibility
    - The system is compatible with any system that can run a web server.

Performance

1. Response Time
    - The system should aim for low response times to ensure that students, faculty members, and administrators can interact with the system efficiently. Short response times contribute to improved user satisfaction and productivity.
2. Throughput
    - To ensure efficient student engagement, the system should be designed to handle a high throughput, enabling multiple users to interact with the system without experiencing delays or performance degradation.
3. Capacity
    - The system should be designed to have sufficient capacity to accommodate the expected number of users and data volume. Scalability measures, such as horizontal scaling, load balancing, and utilizing cloud infrastructure, can help increase the system's capacity to handle increasing demands.
4. Availability
    - For our system, it is crucial to ensure high availability to prevent disruptions in communication and engagement. This can be achieved through redundancy, fault tolerance, and proper backup and recovery mechanisms. High availability

minimizes downtime and ensures that users can access the system whenever they need it.

Other requirements

1. Security
    - The system should implement secure authentication mechanisms, such as username/password combinations or multi-factor authentication, to verify the identity of users. It should also enforce access controls to limit user access to appropriate resources. Encryption should be used to protect data during transmission and storage. Regular security audits, vulnerability assessments, and penetration testing should be conducted to identify and mitigate potential security risks.
2. Safety
    - Safety requirements ensure that the system operates in a safe and reliable manner, preventing harm to users or damage to infrastructure. For instance, the system should not cause any physical harm to users or compromise the integrity of personal and academic data. It should adhere to industry standards and best practices to minimize risks associated with cybersecurity threats, data breaches, or system failures.
3. Legal and regulatory
    - The system should comply with relevant laws, regulations, and policies. This includes data protection regulations, such as GDPR or local data privacy laws, which govern the collection, storage, and processing of personal information. Intellectual property laws should be respected, ensuring that copyrighted materials are not infringed upon. Additionally, the system should comply with academic policies and guidelines set by the UTM Faculty of Computing.
4. Environment
    - The system should consider environmental factors, such as sustainability and energy efficiency. This can include optimizing server infrastructure, using power-saving measures, and adopting eco-friendly practices in system operations. Minimizing resource consumption and employing green computing practices can contribute to a more environmentally friendly system.

2.4 **Design Constraints**

1. Technology Constraints
    - The system may need to be designed within specific technology constraints defined by the university or the computing faculty. For example, the system may need to be built using a particular programming language, framework, or database management system.
2. Integration Constraints
    - The system may need to integrate with existing systems or platforms used by the UTM Faculty of Computing, such as the university's student information system or learning management system. Integration constraints may dictate the protocols, data formats, or APIs that need to be followed for seamless integration.
3. Performance Constraints
    - The system may have specific performance requirements that must be met. For example, it may need to support a certain number of concurrent users, achieve a specified response time, or handle a particular transaction throughput. Designing the system to meet these performance constraints is crucial.
4. Security Constraints
    - The system may have security constraints that require specific security measures to be implemented. This can include encryption of sensitive data, adherence to security protocols, compliance with data protection regulations, and access control mechanisms.
5. Usability Constraints
    - The system should be designed to meet usability requirements and provide an intuitive and user-friendly interface. Usability constraints may include considerations such as accessibility for users with disabilities, adherence to user experience design guidelines, and support for multiple languages.
6. Time Constraints
    - The system may have time constraints that require the project to be completed within a specific timeframe. Time constraints can influence the development methodology, resource allocation, and prioritization of features and functionalities.
7. Budget Constraints
    - The system may need to be designed and implemented within a predefined budget. Budget constraints can impact technology choices, development resources, and the scope of the system's features and capabilities.
8. Regulatory Constraints
    - The system may need to comply with specific regulatory requirements, such as data protection regulations, privacy laws, or academic policies. Compliance constraints can dictate how data is collected, stored, processed, and shared within the system.

9. Scalability Constraints
    - The system may need to be designed to handle future growth and accommodate an increasing number of users, courses, and resources. Scalability constraints may impact architectural decisions, database design, and the selection of infrastructure and hosting options.
10. Maintainability Constraints
    - The system should be designed with maintainability in mind, making it easier for future updates, bug fixes, and enhancements. Maintainability constraints may include the use of modular design patterns, clear documentation, and well-structured code.