SECJ2203: Software Engineering

System Documentation (SD)

One Touch FYP System

Version 2

16 June 2022

School of Computing, Faculty of Engineering

Prepared by: Group 9 <DRAMA QUEEN>

Felicia Chin Hui Fen	A20EC0037
Goh Yitian	A20EC0038
Gui Yu Xuan	A20EC0039

Revision Page

a. **Overview**

The current document includes an introduction to the One Touch FYP System, a description of the overall functionalities and the specific requirements of the system. The specific requirements include external interface requirements, system features, performance, other requirements, design constraints, and software system attributes of the system.

b. **Target Audience**

The target audiences for this project are UTM's students, coordinators of the School of Computing, industrial supervisor and UTM's lectures.

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**
Goh Yitian	Group Leader	1. Use Case Diagram For All Methods	Completed
		2. Use Case <RegisterUser>	Completed
		3. Use Case <SubmitProject>	Completed
		4. Use Case <ChatBetweenUser>	Completed
		5. Activity Diagram <RegisterUser>	Completed
		6. Activity Diagram <SubmitProject>	Completed
		7. Activity Diagram <ChatBetweenUser>	Completed
		8. Sequence Diagram <RegisterUser>	Completed
		9. Sequence Diagram <SubmitProject>	Completed

		10. Sequence Diagram <ChatBetweenUser>	Completed
		11. Identify the Definitions, Acronyms and Abbreviations	Completed
		12. Overview of the System	Completed
		13. User Interface for Register Users	Completed
		14. User Interface for Submit Project	Completed
		15. User Interface for ChatBetweenUser	Completed
		16. Specify the Hardware Interface	Completed
		17. Determine the Design Constraints	Completed
		18. Complete Package Diagram	Completed
		19. Overview of Interface	Completed
		20. Details Description for Project Subsystem	Completed
		21. Package Diagram for Project Subsystem	Completed
		22. Class Diagram for Project Subsystem	Completed
		23. Table of Methods for Project Subsystem	Completed
		24. Sequence Diagram for <Post and View Forum Scenario>	Completed
		25. Sequence Diagram for <Set up Notification Scenario>	Completed
		26. Sequence Diagram for <Manage and Fill Form Scenario>	Completed
		27. Sequence Diagram for <Submit Project Scenario>	Completed
Felicia Chin Hui Fen	Group Member	1. Draw the general activity diagram	Completed
		2. Use Case < LoginUser>	Completed
		3. Use Case <MeetingBetweenUser>	Completed
		4. Use Case <SetUpNotification>	Completed
		5. Activity Diagram < LoginUser>	Completed
		6. Activity Diagram <MeetingBetweenUser>	Completed
		7. Activity Diagram <SetUpNotification>	Completed
		8. Sequence Diagram < LoginUser>	Completed
		9. Sequence Diagram <MeetingBetweenUser>	Completed
		10. Sequence Diagram <SetUpNotification>	Completed

		11. Check and Correct the Use Case Diagram and Domain Model	Completed
		12. Purpose	Completed
		13. User Interface for Login Users	Completed
		14. User Interface for Calendar	Completed
		15. User Interface for Online Meeting	Completed
		16. Software interface	Completed
		17. Performance and other requirements	Completed
		18. Component Model	Completed
		19. Check Component Model and Complete Package Diagram	Completed
		20. Details Description for Communication Subsystem	Completed
		21. Package Diagram for Communication Subsystem	Completed
		22. Class Diagram for Communication Subsystem	Completed
		23. Table of Methods for Communication Subsystem	Completed
		24. Sequence Diagram for <Online Meeting scenario>	Completed
		25. Sequence Diagram for <Chatting scenario>	Completed
Gui Yu Xuan	Group Member	1. Domain Model For All System	Completed
		2. Use Case <FillingForm>	Completed
		3. Use Case <EvaluateProject>	Completed
		4. Use Case <PostingForum>	Completed
		5. Activity Diagram <FillingForm>	Completed
		6. Activity Diagram <EvaluateProject>	Completed
		7. Activity Diagram <PostingForum>	Completed
		8. Sequence Diagram <FillingForm>	Completed
		9. Sequence Diagram <EvaluateProject>	Completed
		10. Sequence Diagram <PostingForum>	Completed
		11. Identify Scope of the System	Completed
		12. User Interface for Filling Form	Completed

		13. User Interface for Evaluate Project	Completed
		14. User Interface for Posting Forum	Completed
		15. Specify the Communication Interface	Completed
		16. Determine Software System Attribute	Completed
		17. Check Component Model and Complete Package Diagram	Completed
		18. Architecture Style and Rationale	Completed
		19. Details Description for Project Evaluation Subsystem	Completed
		20. Package Diagram for Project Evaluation Subsystem	Completed
		21. Class Diagram for Project Evaluation Subsystem	Completed
		22. Table of Methods for Project Evaluation Subsystem	Completed
		23. Sequence Diagram for <Evaluate Project Scenario>	Completed
		24. Table of Methods for another method used in the system (Login and Register)	Completed
		25. Sequence Diagram for <Login Scenario>	Completed
		26. Sequence Diagram for <Register Scenario>	Completed

d. **Version Control History**

Version	**Primary Author(s)**	**Description of Version**	**Date Completed**
1.0	All team members	1. Completed Chapter 1 and Chapter 2, Section 1. 2. All team members discuss together to discuss the details of this system. 3. All team members work together to produce the requirements.	5 June 2022

2.0	All team members	1. Completed Chapter 3, Chapter 4, Chapter 5 and Chapter 6, Section 1. 2. All team members discuss together the details of the component models and the interface. 3. Make corrections in Chapter 1 and Chapter 2, Section 1	16 June 2022

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. 1016-1998 1), and Software Test Documentation (IEEE Std. 829-2008) that are simplified and customized to meet the need of SECJ2203 course at School of Computing, UTM. Examples of models are from Arlow and Neustadt (2002) and other sources stated accordingly.

Table of Contents

1 Introduction 3-7

1.1 Purpose 3

1.2 Scope 3-5

1.3 Definitions, Acronyms and Abbreviations 5

1.4 References 6

1.5 Overview 7

2 Specific Requirements **8-56**

2.1 External Interface Requirements

2.1.1 User Interfaces 8-11

2.1.2 Hardware Interfaces 11-12

2.1.3 Software Interfaces 12-14

2.1.4 Communication Interfaces 14

2.2 System Features 15-25

2.2.1 UC001: Use Case <RegisterUser> 26-28

2.2.2 UC002: Use Case <LoginUser> 29-31

2.2.3 UC003: Use Case <SetUpNotification> 32-34

2.2.4 UC004: Use Case <PostingForum> 35-37

2.2.5 UC005: Use Case <EvaluateProject> 38-40

2.2.6 UC006: Use Case <FillingForm> 41-44

2.2.7 UC007: Use Case <MeetingBetweenUser> 45-48

2.2.8 UC008: Use Case <SubmitProject> 49-51

2.2.9 UC009: Use Case <ChatBetweenUser> 52-54

2.3 Performance and Other Requirements 55

2.4 Design Constraints 56

2.5 Software System Attributes 56

3 System Architectural Design **57-59**

3.1 Architectural Style and Rationale 57

3.2 Component Model 58-59

4 Detailed Description of Components **60-96**

4.1 Complete Package Diagram 60

4.2 Detailed Description

4.2.1 P001: <Project> Subsystem 61-71

4.2.2 P002: <Project Evaluation> Subsystem 72-76

4.2.3 P003: <Communication> Subsystem 77-96

5 Data Design **97-101**

5.1 Data Description 97

5.2 Data Dictionary 98-101

6 Interface Design **102-103**

6.1 Overview of Interface 102-103

1. Introduction

1.1 Purpose

This system documentation (SD) describes how the system, One Touch FYP System works which aims to help the users to complete the final year project's task. The system documentation will provide an overview for the users to understand how the platform in the system interacts with each other. The intended audience for the system documentation (SD) are students, supervisor, coordinator and the evaluator.

Besides that, the software requirement specification (SRS) will help to show and describe how the system will do and how it will be expected to perform for the users. It also describes the functionality that the system will perform to help in the final year project process.

Furthermore, the purpose of the software design document (SDD) is to show the overall design and architecture of the system and explain how the system will be built to meet a set of technical requirements.

Lastly, the software test description (STD) will show the test preparation, test cases and test procedures to be used to test the system, One Touch FYP System.

1.2 Scope

The software product that our group will be producing is One Touch FYP System. It is a platform that allows different users, such as undergraduate students, supervisors and committees of the School of Computer to communicate and share tasks or information about final year projects. By using this system, every activity that should be done will be more efficient and effective without wasting materials, time, or energy. Students able to complete the form that are needed in completing their final year project directly in the system, supervisor and evaluators able to view the students' final year projects and evaluate it directly in the system, communication and online meeting become easier as it can be done in the system and coordinators able to see all the progress via the system.
The scope of the system includes:

a. Forum
    i. For coordinators
    ii. To share the news and information such as news, PSM calendar, PSM presentation schedule and list of lecturers and their area of expertise for students.

b. Chat Box
    i. For students, supervisors, coordinators and evaluators.
    ii. Coordinators act as customer services to answer student's questions about the system.
    iii. Enable students to communicate with supervisors and evaluators.

c. Meeting Platform
    i. For coordinators, students, supervisors and evaluators.
    ii. Coordinator brief on suitable topics, lecturer and his/her area
    iii. Students meeting with supervisor to discuss the project
    iv. Students make presentations on this platform.

d. Submission platform
    i. For students.
    ii. Students submit their project.

e. Form Platform
    i. For students.
    ii. Students fill the forms that are needed to be submitted.
    iii. Project Proposal Form, Meeting logbook, Project Evaluation Form, Draft Report Submission Form and Changing Project Title Form are the forms provided in the system.

f. Evaluation Platform
    i. For supervisors and evaluators.
    ii. Students' PSM report will be received and able to make the evaluation

g. Calendar
    i. For students and coordinators.
    ii. Coordinators set the important dates to remind students.

iii. Students are able to set the date on their own for alert purposes.

The main goal of the proposed system, the One Touch FYP System is to simplify the communication and interaction between users. It is different from the current system where third parties must be used for other purposes. Every activity is able to be done directly via the system which is able to save materials, users time and energy. In addition, a calendar created for this system will be able to notify and remind users by preventing them from forgetting important dates. To conclude that, the proposed system is able to ease the use of users, simplify the interaction between different users and increase the efficiency and effectiveness of the system.

1.3 Definitions, Acronyms and Abbreviation

Definitions/ Acronyms/ Abbreviation	**Meaning**
PSM	Project Sarjana Muda
FYP	Final Year Project
Coordinator	The person who manages the FYP program and students in this system
Supervisor	The user who will supervise the student's progress
Evaluator	The user who will evaluate the students' project

1.4 References

i. ManageEngine. (n.d.). Network Protocols. Retrieved on 31 May 2022, from https://www.manageengine.com/network-monitoring/network-protocols.html.

ii. Sommerville, I., Fowler, M., Beck, K., Brant, J., Opdyke, W., & Roberts, D. (2019). Edition: Software Engineering. *Instructor*.

iii. How to write a software design document (SDD). (n.d.). Retrieved 1 June 2022, from https://www.nuclino.com/articles/software-design-document

iv. Lutkevich, B. (2022, February 25). *What is software documentation? definition, types and examples*. SearchSoftwareQuality. Retrieved 1 June 2022, from https://www.techtarget.com/searchsoftwarequality/definition/documentation

v. Editor. (2019, December 9). *Technical documentation in software development: Types, best practices, and Tools*. AltexSoft. Retrieved 1 June 2022, from https://www.altexsoft.com/blog/business/technical-documentation-in-software-development-types-best-practices-and-tools/

vi. Rsameser. (n.d.). *Getting started with Windows IOT Enterprise*. Microsoft Docs. Retrieved 1 June 2022, from https://docs.microsoft.com/en-us/windows/iot/iot-enterprise/getting_started

vii. *What is software testing? definition of software testing, software testing meaning*. The Economic Times. (n.d.). Retrieved 1 June 2022, from https://economictimes.indiatimes.com/definition/software-testing

viii. *The Essential Tool for Mathematics*. Maple. (n.d.). Retrieved 1 June 2022, from https://www.maplesoft.com/products/Maple/

1.5 **Overview**

This system documentation consists of specific requirements for the One Touch FYP System which are the external interface, hardware interface, software interface, user interface, and communication interface. In the system features, domain model, use case, use case description, sequence diagram, and activity diagram of each feature in the system will be described.

After that, performance and other requirements, as well as design constraints, will be specified in Chapters 2.3 and 2.4. Lastly, the software system attributes explain the attributes that will be used in this system.

This system documentation is being organized by all the team members of the One Touch FYP System. All the system features and the chapters in this documentation are discussed and done together in order to produce a system software that can ease the process while undergoing PSM.

2. Specific Requirements

2.1 External Interface Requirements

2.1.1 User Interfaces

RegisterUser Interface

The registration interface has a form for users to fill in their information details such as name, email address, gender, and contact number. A unique userID is also required. The system will compare the userID entered by the user with the UTM database. The user can only register successfully if the userID exists in the UTM database. This is to ensure only UTM's students and staff as well as the company that has UTM students interning in it can use the system. If the userID is matched, the system will add the new user information into the system. After registering successfully, the system will display a registration successful message and the user can now use the username and password created to log in to the system. If the userID does not exist, the system will display an error message and the user is unable to register and login into the system.

Login Interface

The login interface provides an option for users to login in the system. It provides a place for users to enter username and password in the system. The user chooses the "login" option in the login interface and fills in their username and password. The system will check the validity of the username and password. If the username and password is valid, the user will redirect to the system home page. Else, the system will display an error message to the user. The user needs to re-enter the username and password if they want to access the system. The user who is able to enter the system can view their personal information in the system. If the user does not have an account in the system, they can choose the "Register" option in the login interface to register in the system.

Calendar Interface

The calendar interface provides a notification setting for coordinator or students to remind the student or themselves of the upcoming students activity in the system. Firstly, the coordinator needs to set up the student's activity in the system. Then, the students need to fills their email

address when they register in the system. In order to set up the notification in the system, users need to click on the “settings option” in the calendar interface. If the user is a student, the system will display the event that was set by the coordinator. The student selects the activity and sets the date and time they wish to notify for themselves. If the user is the coordinator, the system will display a list of student events for the coordinator. The coordinator selects the student’s activities and sets the date and time to remind the student. After the settings, the user clicks the save option to save it in the system. The system will notify the student based on the time and date via the student’s email.

PostingForum Interface

First, the user must log in to the system. Then, on the user profile page, there will be a navigation bar that the user can click to select different functions. The user clicks the "Forum" button. After the user clicks, they will be directed to the forum interface. In the forum interface, there will be a "Post" button that allows users to post new posts, with previously posted content displayed below the button. If the user wishes to publish a new post, click the "Post" button. After that, a form will be displayed. Users can type information into the form. If the user wishes to upload a file, the user can click the "Upload File" button and upload the file. Otherwise, the user can click the "Submit" button to submit the post.

EvaluateProject Interface

First, the user must log in to the system. Then, on the user profile page, there will be a navigation bar that the user can click to select different functions. The user clicks the "Evaluation" button. After the user clicks, it will jump to the evaluation interface. In the Evaluation interface, there will be a list of students' projects. Each project will have a "View" button that allows the user to view the project and an "Evaluate" button to enter grades. If the user chooses to view the student's project, the project file will be displayed on the screen. If the user chooses to evaluate the item, an evaluation form is displayed. Then, the user can enter the marks in the evaluation form.

FillingForm Interface

First, the user must log in to the system. Then, on the user profile page, there will be a navigation bar that the user can click to select different functions. The user clicks the "Form" button. After the user clicks, it will be directed to the filling form interface. In the filling form interface, there will be a navigation bar showing the different forms available in the system.

Users can choose whether to fill in the Project Proposal Form, Meeting Log, Evaluation Form, Draft Report Submission Form, and Change Project Title form. After clicking on any form, the relevant form is displayed on the screen, and the user can then enter information and submit. If the user wishes to submit the file while filling out the form, the user can click the "Upload" button to upload the file.

MeetingBetweenUser Interface

Online meeting interfaces provide a platform for users to meet together virtually. The user clicks on the “Meeting” button to access the meeting room interface. Then, users need to select the way to start the meeting. If the user has the meeting room id, user select the “Enter meeting room ID” option to enter the meeting room id. Then the system will find the matched meeting room based on the meeting room id. If the meeting room id entered is invalid, the system will display an error message to the user. Otherwise, the system will redirect the user to the meeting room.

For the user who does not have a meeting room id and wants to meet online, they can choose the “open a new room” option in the meeting room interface. The system will assign the user to an empty room with a meeting room id. The user can share the meeting room id by clicking on the settings option and copy the id. In the virtual meeting room, the user can use the meeting room’s functions such as microphone, slideshare and “start video”. If the user wants to record the meeting, they can click the “record” button to start the recording. The system will record the meeting and save it. After the meeting ends, the user can click the “leave” button to leave the meeting room. Then, the system will automatically generate and save it in the system database. A URL link that links to the recorded video will be sent to the user’s email.

SubmitProject Interface

Firstly, the user must log in to the system. Then, on the user profile page, there will be a navigation bar that the user can click to select different functions. The user clicks the “Submission” button. After clicking, the user will be directed to the project submission interface. On this page, there is a form that allows the user to upload files. In the form, an “upload” button is present on the left side, after the user clicks the “upload” button, a “choose file” button will be shown. After clicking it, the user’s file explorer will pop up and the user can choose the file he/ she wants to upload. After clicking on the file he/ she chooses to upload, the system will upload the file. If the file uploaded successfully, the system will

display the fileName and fileID that user uploaded. Else if upload fails, the system will display an error message and the user needs to upload again. After that, the user can click the "save changes and submit" button and the file will be submitted and saved to the system.

ChatBetweenUser Interface

Firstly, the user must log in to the system. The "Conversation" button will be displayed on the bottom right side of the page. The user can click on it to start the conversation. After clicking the "conversation" button, a list of the name will pop up and the user can choose the name he/ she wishes to chat with. After choosing the name, the user can start the conversation by typing the message and clicking the "send" button on the right side of the textbox after finishing typing.

If the user received a message, the chat box will pop up and display the message received. The user can reply by typing the message in the textbox and clicking the "send" button to send it.

If the user wishes to find the customer service, the user clicks on the "Conversation button" and the "Student Care" will be the first in the name list. The user can choose "Student Care" and send the message with the question he/ she wishes to get help.

2.1.2 Hardware Interfaces

Our proposed system should be accessed through a personal computer. Some input and output devices supported by the system are:

> Mouse - to click on the function button such as register, submit, etc

> Keyboard - to input data such as personal information when register and type message for conversation

> 4GB HD space required for a typical live system with 1000-2000 events.

> Recommended minimum CPU - Pentium 4.3.2GHz

> Recommended 1GB RAM for a Central Server with 3 nodes.

> Network Card

Our system will be running on a website, therefore, using the standard port numbers 80 for HTTP. Besides that, as the user needs to upload a file to submit their project, File Transfer

Protocol (FTP) and port 21 will be used. Moreover, Transmission Control Protocol (TCP) and port 3306 will be used for the MySQL database.

Our system supported most operating systems including Windows, Mac OS, and Linux. Therefore, most of the devices could access our system. The users are required to use a modern web browser such as Mozilla Firefox 1.5, Internet Explorer 6 and Google Chrome to use our web-based system.

2.1.3 Software Interfaces

a. Data management system

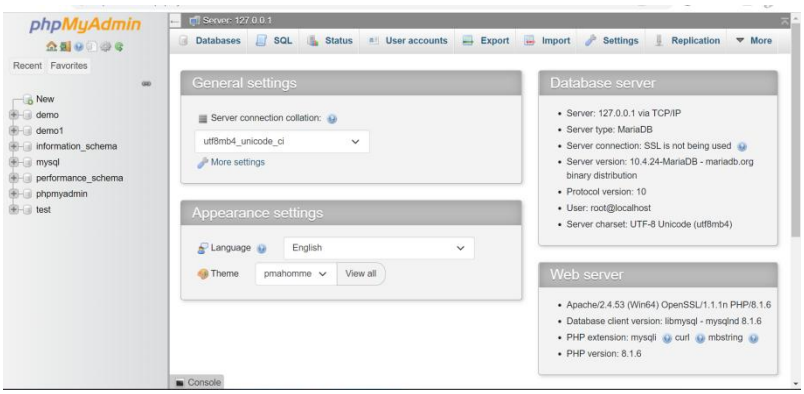

Figure 1: phpMyAdmin dashboard

b. Name: phpMyAdmin

i. Mnemonic: phpMyAdmin

ii. Specification number: N/A

iii. Version number: phpMyAdmin 5.2.0

iv. Source: https://www.javatpoint.com/phpmyadmin

In our system, we use phpMyAdmin for database management. This is because phpMyAdmin can run on any server or operating system. It can easily manage the database, relation, tables, columns, users, and so on in MySQL. It can also perform administrative functions like database creation and query execution.

c. Operating system

i. Name: Windows 10 Internet of Things Enterprise
ii. Mnemonic: Windows 10 IoT Enterprise
iii. Specification number: N/A
iv. Version number: LTSC (Long-Term Servicing Channel) 2021
v. Source:https://docs.microsoft.com/en-us/windows/iot/iot-enterprise/getting_started

We will use Windows Internet of Things Enterprise as our operating system. This is due to the fact that it is a productive system that is capable of building and managing Windows IoT Enterprise devices using powerful tools and technologies to unlock data and drive digital transformation. Apart from that, it is saved to protect our devices, data, and users' identities. Last but not least, it can connect devices to a network and the cloud.

d. Mathematical package

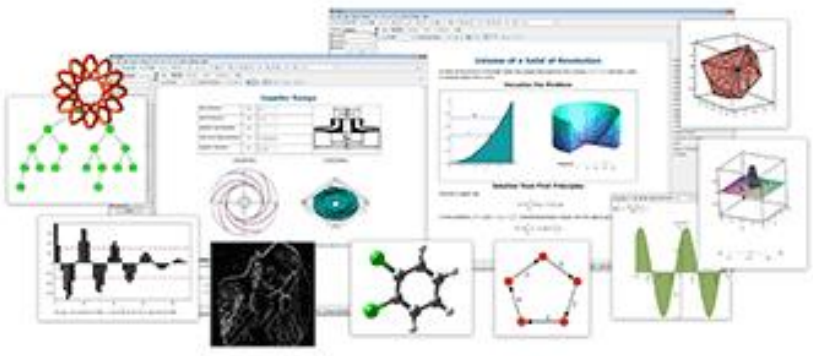

Figure 2: A Google image of what Maple is capable of.

i. Name: Maple
ii. Mnemonic: Maple
iii. Specification number: N/A

iv. Version number: Maple 2022

v. Source:https://www.maplesoft.com/products/Maple/

The system will use Maple for the mathematical package. Maple is a math programme that combines the world's most powerful math engine with an interface that makes analysing, exploring, visualizing, and solving mathematical problems easier.

2.1.4 Communication Interfaces

The network protocols used are based on OSI models in which the communication process will be splitted into 7 layers. Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are used for data transmission. Both protocols are useful as they will provide error correction while transmitting data and control the flow of the transmission. TCP is mainly used for transmitting large datagram messages while UDP is used for transmitting short datagram messages.

File Transfer Protocol (FTP) also used in the system as it enables the transferring of files. It is important in this system as users need to upload their project report into the system. Besides that, HyperText Transfer Protocol (HTTP) is also used in this system. HTTP is an application layer protocol designed to transfer information between networked devices. This means that the information entered by the user is saved on the server and can be shared with other users. Hence, the communication between students, coordinators, supervisors and evaluators can be easier as they will be able to share over the web.

2.2 System Features

The system features include Use Case Diagram, Activity Diagram of System, Domain Model, Use Case Description, Activiti Diagram of every use case and sequence diagram. The features include in Use Case Description are <RegisterUser> for students and supervisors, <LoginUser>, <SetUpNotification>, <MeetingBetweenUser> and <ChatBetweenUser> for coordinators, evaluators, students and supervisors, <PostingForum> for coordinators, <EvaluateProject> for supervisors and evaluators, <FillingForm> , <SubmitProject> for students.

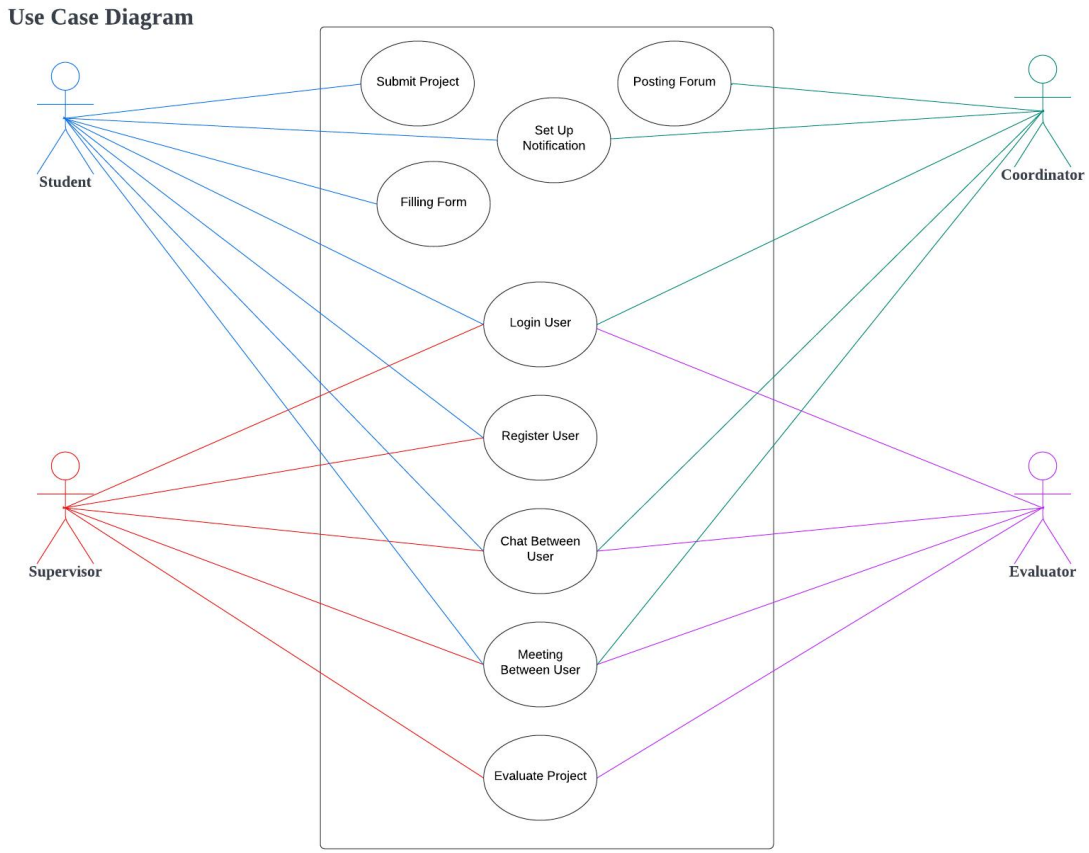

Figure 2.1: Use Case Diagram for <One Touch FYP System>

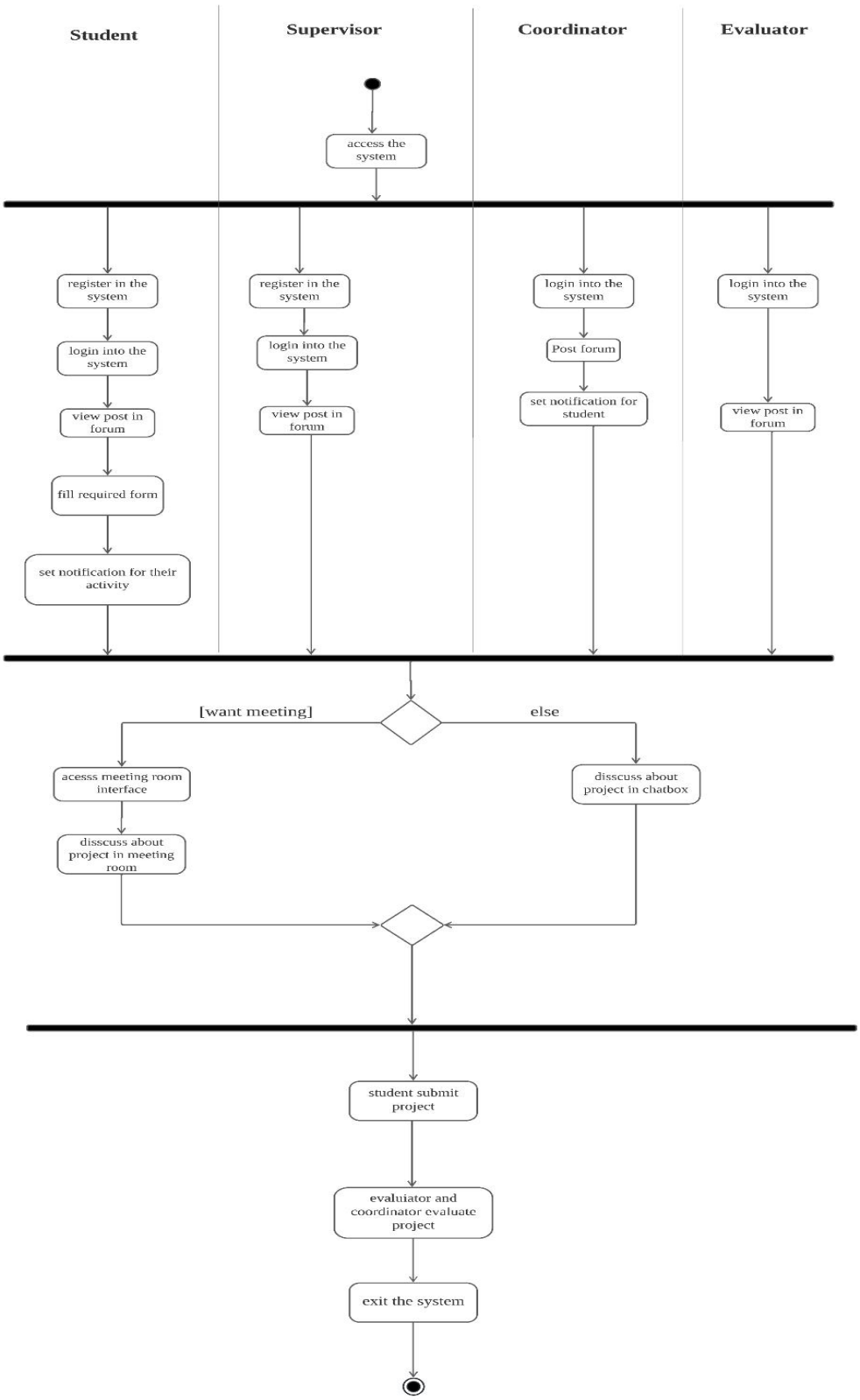

Figure 2.2: Activity Diagram for <One Touch FYP System>

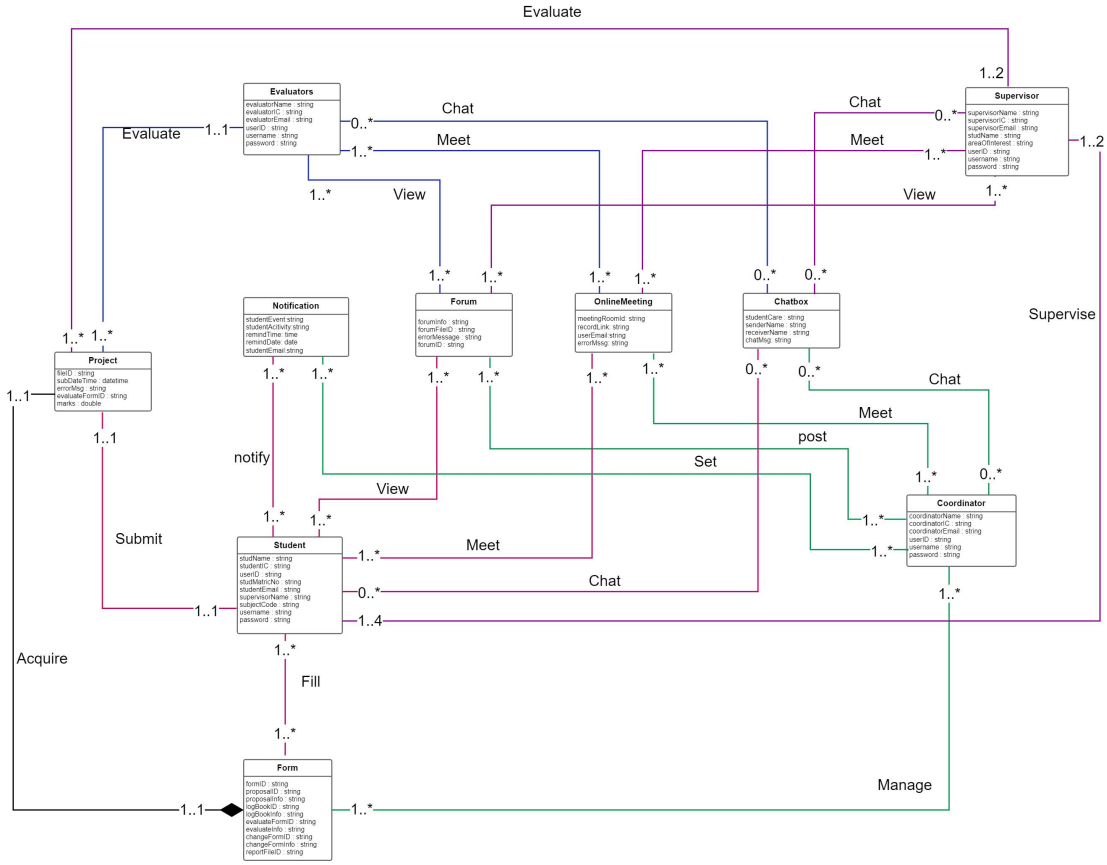

Figure 2.3: Domain Model for <One Touch FYP System>

Classes and Relationship explanation

a. Student

The attributes involved are:

i. studName : Name of student

ii. studentIC : Unique IC number of student

iii. userID : Unique identification of student in the system

iv. studMatricNo : Matric number of student

v. studentEmail : Email of student

vi. supervisorName : Supervisor name of the student

vii. subjectCode : Subject taken by the student

viii. username : Unique username of student for login in the system

ix. password : Unique password of student for login in the system

Relationship of the class

i. Student and Project: Student submits the project that has the mandatory relationship with the multiplicity one to one (1..1 : 1..1)
ii. Student and Notification: Student sets the notification of their activity that has the mandatory relationship with the multiplicity of many to many (1..* : 1..*)
iii. Student and Forum: Student views the post in the forum that has the mandatory relationship with the multiplicity of many to many (1..* : 1..*)
iv. Student and OnlineMeeting: Student meets with other users with mandatory relationships with the multiplicity of many to many (1..* : 1..*)
v. Student and Chatbox: Student chats with other users by using the chat box with the optional relationship with the multiplicity of many to many (0..* : 0..*)
vi. Student and Supervisor: Student that are supervised by their supervisor that they have a mandatory relationship with the multiplicity of minimum of 1 up to maximum of 4 students and minimum of 1 up to maximum of 2 supervisors (1..4 : 1..2)
vii. Student and Form: Student fills the form that is provided in the system that has the mandatory relationship with the multiplicity of many to many (1..* : 1..*)

b. Supervisor

The attributes involved are:

i. supervisorName : Name of supervisor
ii. supervisorIC : Unique IC number of supervisor
iii. supervisorEmail : Email of supervisor
iv. studName : Supervised Student Name
v. areaOfInterest : The area of interest of supervisor
vi. userID : Unique identification of supervisor in the system
vii. username : Unique username used for login in the system
viii. password : Unique password used for login in the system

Relationship of the class

i. Supervisor and Project : Supervisor evaluates the project with the mandatory relationship with the multiplicity of one to many (1..1 : 1..*)
ii. Supervisor and Chatbox : Supervisor chats by using the chat box with an optional relationship with the multiplicity of many to many (0..* : 0..*)
iii. Supervisor and OnlineMeeting : Supervisor meets with other users with mandatory relationships with the multiplicity of many to many (1..* : 1..*)
iv. Supervisor and Forum : Supervisor views the forum with a mandatory relationship with the multiplicity of many to many (1..* : 1..*)
v. Supervisor and Student : Supervisor supervises students with a mandatory relationship with the multiplicity of minimum of 1 up to maximum of 2 supervisors and minimum of 1 up to maximum of 4 students (1..2 : 1..4)

c. Coordinator

The attributes involved are

i. coordinatorName : Name of coordinator
ii. coordinatorIC : Unique IC number of coordinator
iii. coordinatorEmail : Email of coordinator
iv. userID : Unique identification of coordinator in the system
v. username : Unique username used for log in
vi. password : Unique password used for log in

Relationship of the class

i. Coordinator and Chatbox : Coordinator chats by using the chat box with an optional relationship with the multiplicity of many to many (0..* : 0..*)
ii. Coordinator and OnlineMeeting : Coordinator meets with other users with mandatory relationships with multiplicity of many to many (1..* : 1..*)
iii. Coordinator and Notification : Coordinator sets the notification of important events with a mandatory relationship with the multiplicity of many to many (1..* : 1..*)

iv. Coordinator and Forum : Coordinator posts the forum with a mandatory relationship with the multiplicity of many to many (1..* : 1..*)

v. Coordinator and Form : Coordinator manages forms with a mandatory relationship with the multiplicity of many to many (1..* : 1..*)

d. Evaluator

The attributes involved are:

i. evaluatorName : Name of evaluators

ii. evaluatorIC : Unique IC number of evaluators

iii. evaluatorEmail : Email of evaluators

iv. userID : Unique identification of evaluator in the system

v. username : Unique username used for log in

vi. password : Unique password used for log in

Relationship of the class

i. Evaluator and Chatbox: Evaluator chats by using the chat box with an optional relationship with the multiplicity of many to many (0..* : 0..*)

ii. Evaluator and OnlineMeeting: Evaluator meets with other users in the online meeting with a mandatory relationship with the multiplicity of many to many (1..* : 1..*)

iii. Evaluator and Forum: Evaluator views the forum with the mandatory relationship with the multiplicity of many to many (1..* : 1..*)

iv. Evaluator and Project: Evaluator evaluates the project with a mandatory relationship with the multiplicity of one to many (1..1 : 1..*)

e. Notification

The attributes involved are:

i. userEvent: The list of event of the student in the system

ii. userAcitivity: The selected student activity to be notified, it consists of the detailed information of the activity of the student.

iii. remindTime: The time to be notify the student

iv. remindDate: The date to be notify the student

v. userEmail: Student email

Relationship of the class

i. Notification and Student: The notification will notify students of their activity with a mandatory relationship with multiplicity of many to many (1..* : 1..*)

ii. Notification and Coordinator: The notification will be set up by the coordinator with a mandatory relationship with multiplicity of many to many (1..* : 1..*)

f. Forum

The attributes involved are:

i. forumInfo : The content of the forum

ii. forumFileID : The file uploaded from computer devices

iii. errorMessage : Error messages return by the system if the uploading of file and submitting the forum is failed

iv. forumID : The forum which contains the content and with or without files that are submitted by coordinators.

Relationship of the class

i. Forum and Evaluator: Forum can be viewed by evaluators with the mandatory relationship with multiplicity of many to many (1..* : 1..*)
ii. Forum and Supervisor: Forum can be viewed by supervisors with the mandatory relationship with the multiplicity of many to many (1..* : 1..*)
iii. Forum and Coordinator: Forum can be posted by coordinators with a mandatory relationship with the multiplicity of many to many (1..* : 1..*)
iv. Forum and Student : Forum can be viewed by students with the mandatory relationship with multiplicity of many to many (1..* : 1..*)

g. Online Meeting

The attributes involved are:

i. meetingRoomId: unique Meeting room Id for the user
ii. recordLink: recorded meeting video URL link
iii. userEmail: Email of the user that use the meeting room
iv. errorMssg: error message displayed to the user when the user enters the wrong meeting room Id.

Relationship of the class

i. Online meeting and Evaluator: Online meetings where evaluators and other users meet have the mandatory relationship with multiplicity many to many (1..* : 1..*)
ii. Online meeting and Supervisor: Online meetings where the supervisor and other users meet have the mandatory relationship with multiplicity many to many (1..* : 1..*)
iii. Online meeting and Coordinator: Online meetings where the coordinator and other users meet have the mandatory relationship with multiplicity many to many (1..* : 1..*)
iv. Online meeting and Student: Online meetings where student and other users meet have the mandatory relationship with multiplicity many to many (1..* : 1..*)

h. Chatbox

The attributes involved are:

i. studentCare : The customer service of the system

ii. senderName : The user who send the message

iii. receiverName : The user who received the message

iv. chatMsg: The content of the message

Relationship of the class

i. Chatbox and Evaluator: Chatbox where evaluators can chat with other users or find Student Care has the optional relationship with multiplicity many to many (0..* : 0..*)
ii. Chatbox and Supervisor: Chatbox where supervisor can chat with other users or find Student Care has the optional relationship with multiplicity many to many (0..* : 0..*)
iii. Chatbox and Coordinator: Chatbox where coordinator can chat with other users or find Student Care has the optional relationship with multiplicity many to many (0..* : 0..*)
iv. Chatbox and Student: Chatbox where students can chat with other users or find Student Care have the optional relationship with multiplicity many to many (0..* : 0..*)

i. Project

The attributes involved are:

i. fileID : Project report of students

ii. subDateTime : project submission date and time

iii. errorMsg : Error messages return by the system if the uploading of file and submitting the project is failed

iv. evaluateFormID : Evaluation form that filled by students will be sent to evaluators and supervisors to enter marks

v. marks : Marks given by supervisors and evaluators

Relationship of the class

i. Project and Supervisor : Project evaluated by supervisors have the mandatory relationship with multiplicity of one to many (1..1 : 1..*)
ii. Project and Evaluator : Project evaluated by evaluators have the mandatory relationship with multiplicity of one to many (1..1 : 1..*)
iii. Project and Student : Project submitted by the student have the mandatory relationship with multiplicity of one to one (1..1 : 1..1)
iv. Project and Form : Project is a composition of Form having a relationship with multiplicity of one to one (1..1 : 1..1)

j. Form

The attributes involved are:

i. formID : Unique identification of the forms that had been filled by students
ii. proposalID : Unique identification of the proposal form
iii. proposalInfo : The content of the proposal form
iv. logBookID : Unique identification of the meeting log book
v. logBookInfo : The content of the meeting log book
vi. evaluateFormID : Unique identification of the evaluation form
vii. evaluateInfo : The content of the evaluation form
viii. changeFormID : Unique identification of the change project title form
ix. changeFormInfo : The content of the change project title form
x. draftID : Unique identification of the draft submission report form
xi. draftInfo : The content of the draft submission report form
xii. reportFileID : Unique identification of the students' project

Relationship of the class

i. Form and Project : Evaluation form will be acquired by Project is a composition relationship with multiplicity of one to one (1..1 : 1..1)
ii. Form and Student : Form filled and submitted by student have the mandatory relationship with multiplicity of many to many (1..* : 1..*)
iii. Form and Coordinator : Forms managed by coordinators with mandatory relationship with multiplicity of many to many (1..* : 1..*)

2.2.1 UC001: Use Case <RegisterUsers>

Table 2.1: Use Case Description for <RegisterUser>

Use case: RegisterUser
ID: UC001
Actors: 1. Student 2. Supervisor
Preconditions: 1. The users had accessed the system. 2. Coordinator and evaluator had registered into the system.
Flows of Events: 1. The user clicks on the "Register" option in the login interface. 2. The user fills in the required information for registration in the register interface. 3. The system verifies the user id by comparing it with UTM's user database. 3.1. If the user id is valid, the system displays a success message to the user. 3.2. The system accepts the information and saves the username and password of the user. 4. Else, the system will display an error message and the user may need to register again. 5. The user goes back into the login interface.
Postconditions: The registered user in the system is able to login into the system.

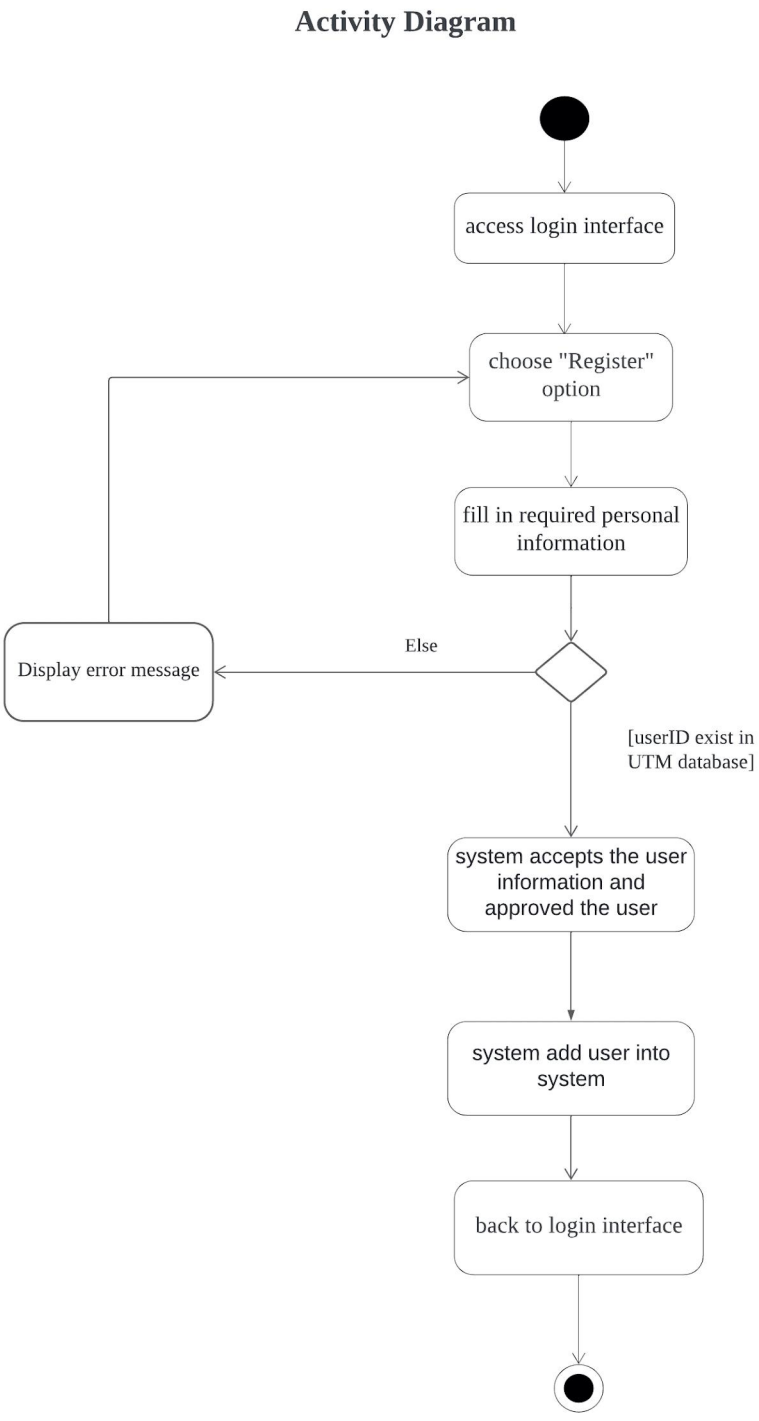

Figure 2.4: Activity diagram for RegisterUser use case

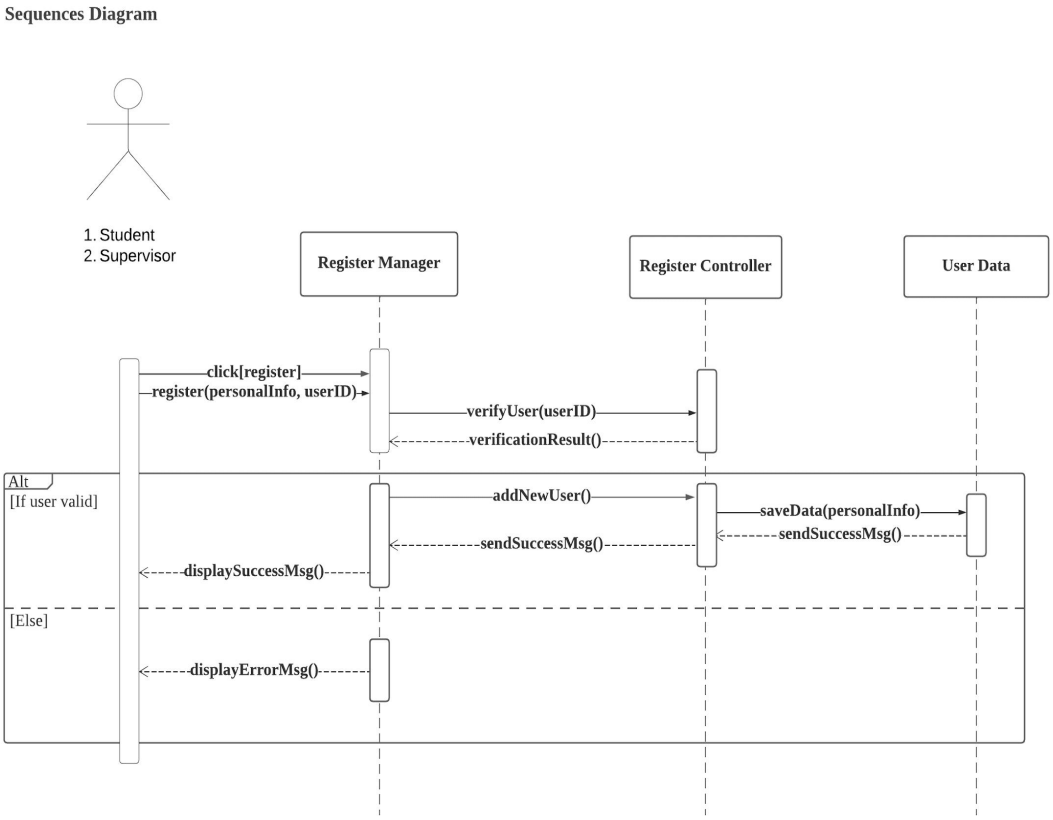

Figure 2.5: Sequence diagram for RegisterUser use case

2.2.2 UC002: Use Case < LoginUser>

Table 2.2: Use case description for LoginUser

Use case: LoginUser
ID: UC002
Actors: 1. Student 2. Supervisor 3. Coordinator 4. Evaluator
Preconditions: 1. The users had accessed the login interface of the system. 2. The users had registered in the system.
Flows of Events: 1. The user clicks on the “login” option in the login interface. 2. The user fills in the username and password. 3. The system checks the validity of the username and password of the user. 3.1. If the username and password is correct, the user enters the system home pages. 3.2. Else, error messages are displayed and the user re-keyIn the username and password again. 4. The system displays the user’s personal information.
Postconditions: 1. The users are able to access and view their personal information in the home page of the system.

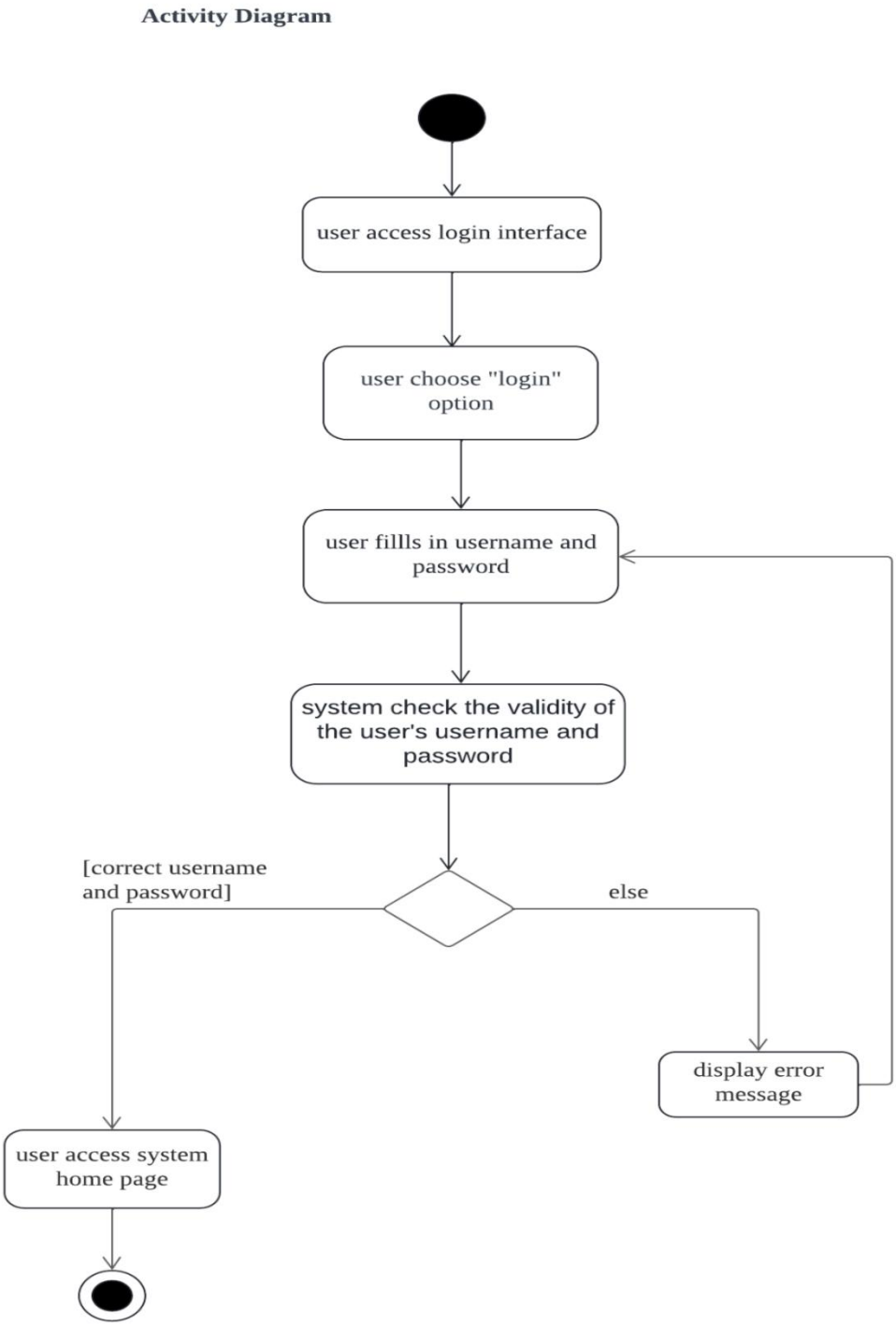

Figure 2.6: Activity diagram for LoginUser use case

Sequences Diagram

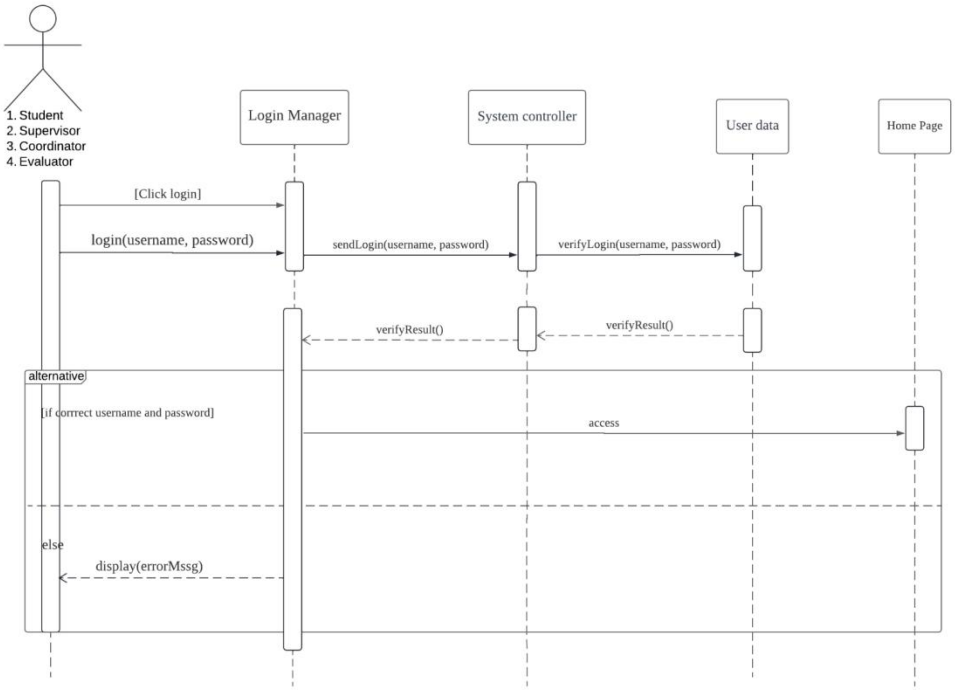

Figure 2.7: Sequence diagram for LoginUser use case

2.2.3 UC003: Use Case <SetUpNotification>

Table 2.3: Use case description for SetUpNotification

Use case: SetUpNotification
ID: UC003
Actors: 1. Student 2. Coordinator
Preconditions: 1. The users had accessed the ”calendar” interface of the system. 2. The students had filled in his/her email account in the system. 3. The coordinator had set the student’s activity. 4. The coordinator had set a due date for the student’s activity.
Flows of Events: 1. The user clicks on the “settings” option in the calendar interface. 2. If the user is a student, the system displays a list of events the student has. 2.1. The student selects the activities they wish to be notified of. 2.2. The student sets the date and time they wish to be notified. 3. Else, the system will display a list of student events for the coordinator. 3.1. The coordinator selects the student’s activities they wish to notify the students. 3.2. The coordinator sets the date and time they wish to notify the students. 4. The user clicks the “save” option to save the notification settings. 5. The system saves the notification settings of the user.
Postconditions: 1. The system will remind the user about the specific activity based on the time date that the user set up via email.

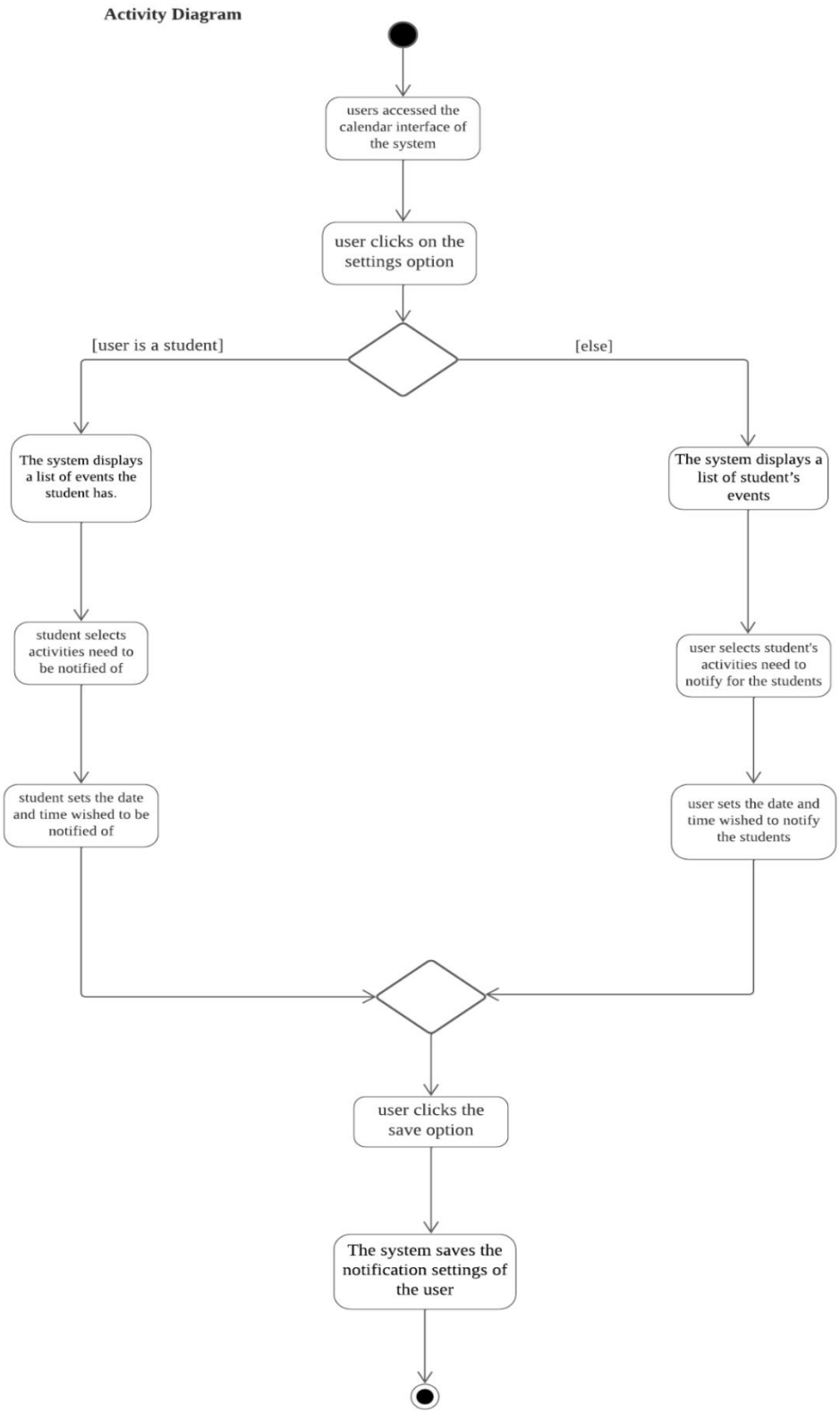

Figure 2.8: Activity diagram for SetUpNotification use case

Sequences Diagram

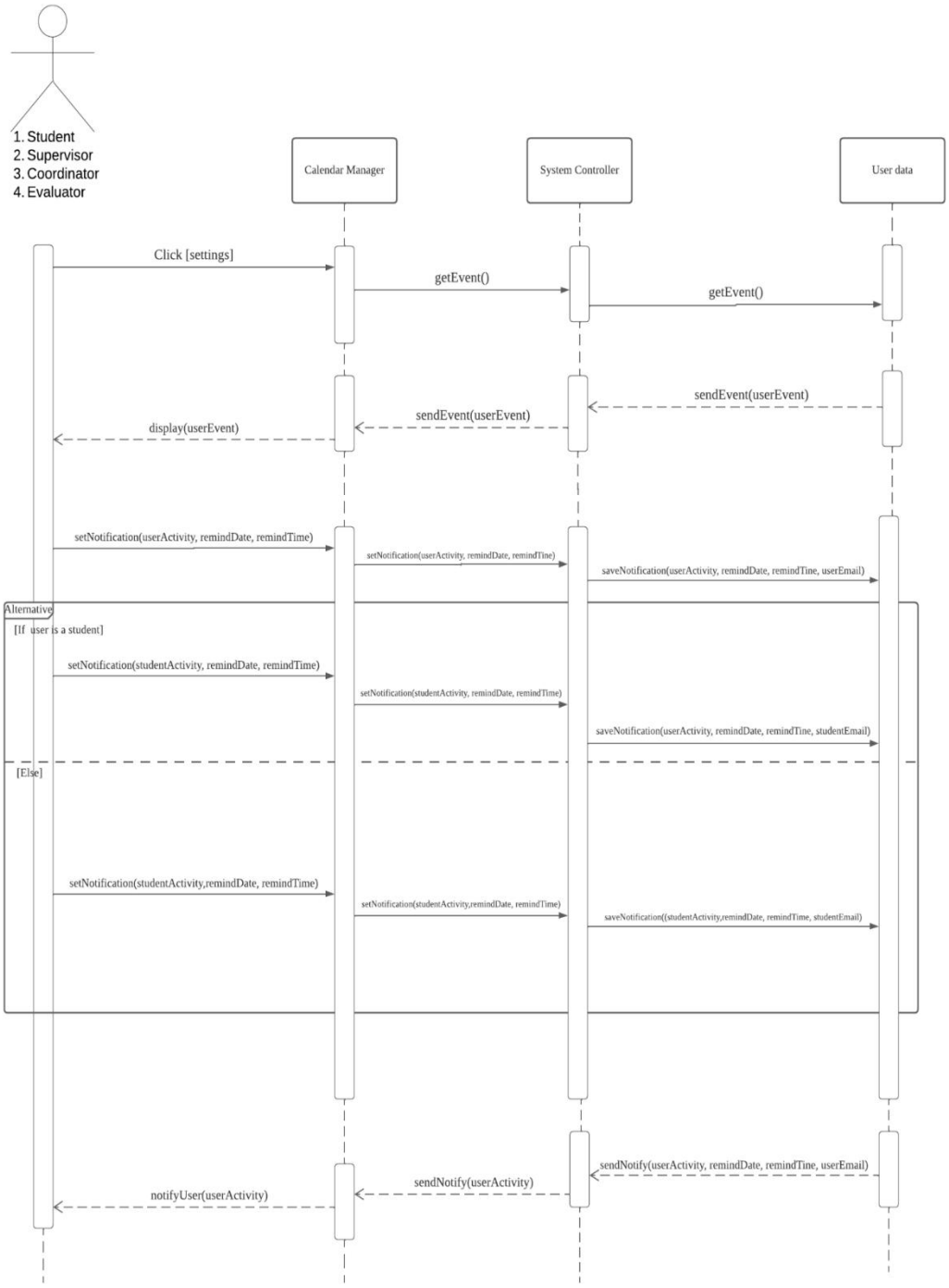

Figure 2.9: Sequence diagram for SetUpNotification use case

2.2.4 UC004: Use Case <PostingForum>

Table 2.4: Use case description for PostingForum

Use case: PostingForum
ID: UC004
Actors: 1. Coordinator
Preconditions: 1. User has logged into the system.
Flows of Events: 1. Users click on the “Forum” button. 2. Users click on the “Post” button. 3. Users typing the information that wish to be shown in the forum. 4. If users want to attach files 4.1. Users select “Upload File” options to upload files. 4.2. If the uploading files is success 4.2.1. System will display it on the screen. 4.3. Else, the system displayed an error message. 4.4. Users upload the files again. 5. Users click on the “Submit” button. 6. If the posting is success 6.1. System saves the information. 7. Else, the system displays an error message. 7.1. Users need to check for the error and correct the content. 7.2. Users click on the “Submit” button.
Postconditions: 1. The post will be uploaded to the Forum template.

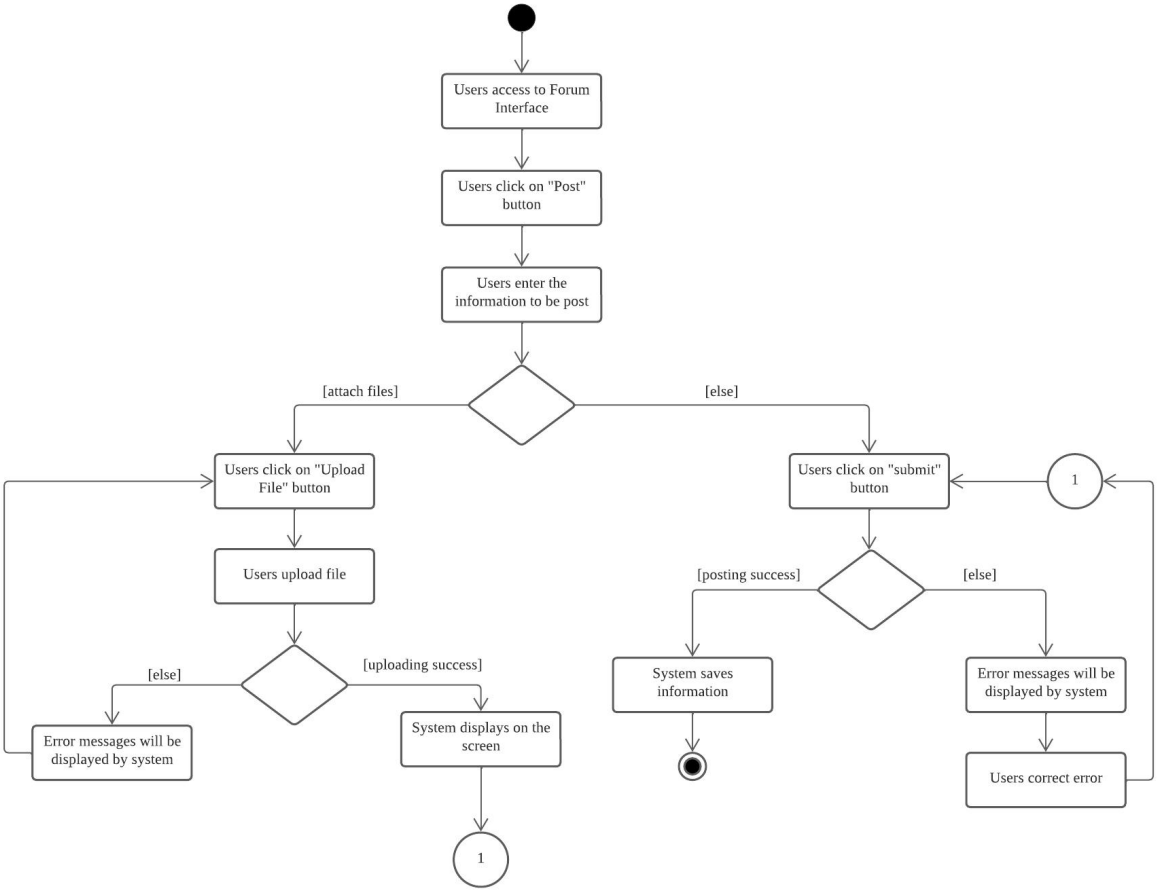

Figure 2.10: Activity diagram for PostingForum use case

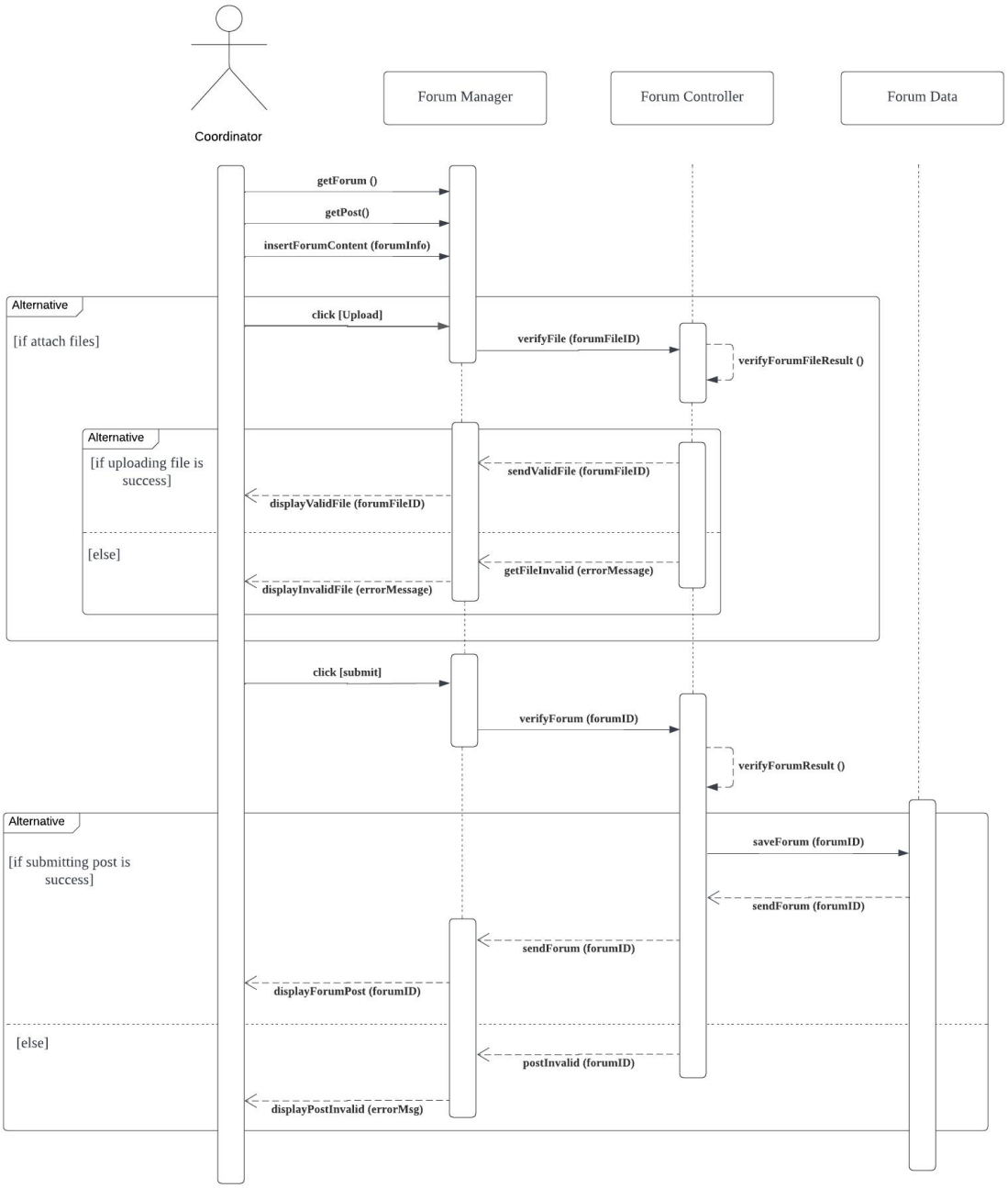

Figure 2.11: Sequence diagram for PostingForum use case

2.2.5 UC005: Use Case <EvaluateProject>

Table 2.5: Use case description for EvaluateProject

Use case: EvaluateProject
ID: UC005
Actors: 1. Supervisor 2. Evaluators
Preconditions: 1. Students had submitted their project report and evaluation form.
Flows of Events: 1. Users click on the “Evaluation” button. 2. Lists of the student’s projects will be displayed by system. 3. Users choose one of the projects. 4. Users click on the “View” button. 5. The content of the project will be displayed by system. 6. Users click on the “Evaluate” button. 7. The evaluation form entered by the users will be displayed by the system. 8. Users can enter the marks in the “Marks” section. 9. If the entering the marks is success, 9.1. System will display the marks on the screen. 10. Else, system display error message 10.1. Users reenter the marks. 11. Users click on the “submit” button. 12. If submit is success 12.1. System saves the information. 13. Else, the system displays an error message. 13.1. Users correct the error and submit again.
Postconditions: 1. The evaluation form will be sent and displayed at the students’ page.

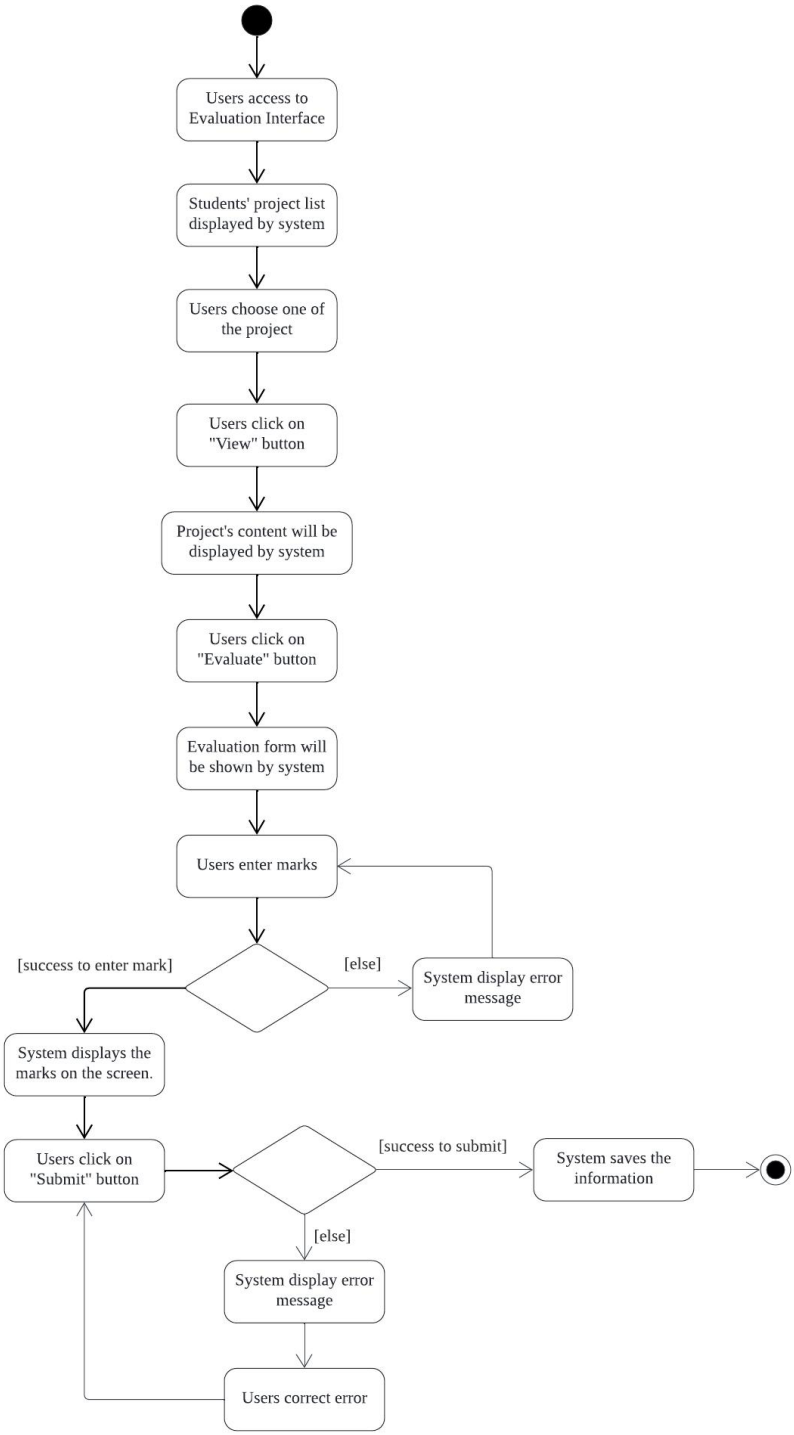

Figure 2.12: Activity diagram for EvaluateProject use case

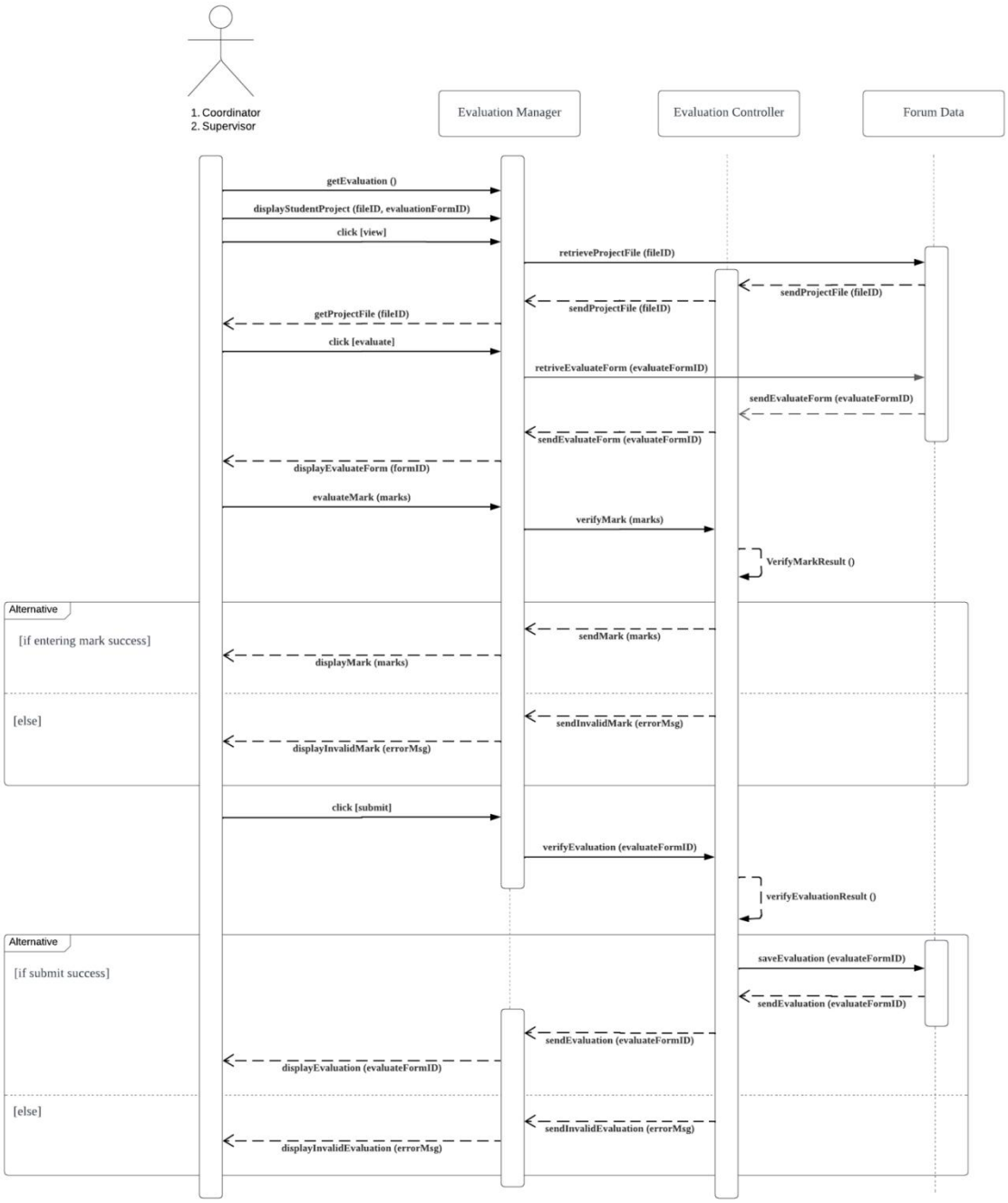

Figure 2.13: Sequence diagram for EvaluateProject use case

2.2.6 UC006: Use Case <FillingForm>

Table 2.6: Use case description for FillingForm

Use case: FillingForm
ID: UC006
Actors: 1. Student
Preconditions: 1. Students had discussed the project with supervisors. 2. Supervisors approved the basic proposal done by students.
Flows of Events: 1. Users click on the “Form” button. 2. Users choose which form they want to fill in. 3. If users wish to fill in the project proposal form, 3.1. Users click on the “Project Proposal Form” button. 3.2. Users fill in the information needed in the form. 4. Else if users wish to fill in the meeting log book. 4.1. Users click on the “Meeting Log Book” button. 4.2. Users fill in the information needed in the form. 5. Else if users wish to fill in the evaluation form. 5.1. Users click on the “Evaluation Form” button. 5.2. Users fill in the information needed in the form. 6. Else if users wish to fill in the draft report submission form. 6.1. Users click on the “Draft Report Submission Form” button. 6.2. Users fill in the information needed in the form. 6.3. Users click on the “Upload” button to upload the draft report. 6.4. If the uploading is success, 6.4.1. The system displays it on the screen. 6.5. Else, the system displays error messages. 6.5.1. Users check for the error and correct it.

7. Else
    7.1. Users click on the “Change Project Title Form” button.
    7.2. Users fill in the information needed in the form.
8. Users click on the “submit” button.
    8.1. If the submission is successful,
        8.1.1. The system will save the information.
    8.2. Else, the system displayed an error message.
        8.2.1. Users check for the error and correct it.
        8.2.2. Users click on the “submit” button.

Postconditions:

1. Project Proposal Form, Meeting Log Book, and Change Project Title Form will be sent to the coordinator for further purposes.
2. Evaluation Form and Draft Report Submission Form will be sent to the evaluators and supervisors for evaluation and sent to the coordinator for further purposes.

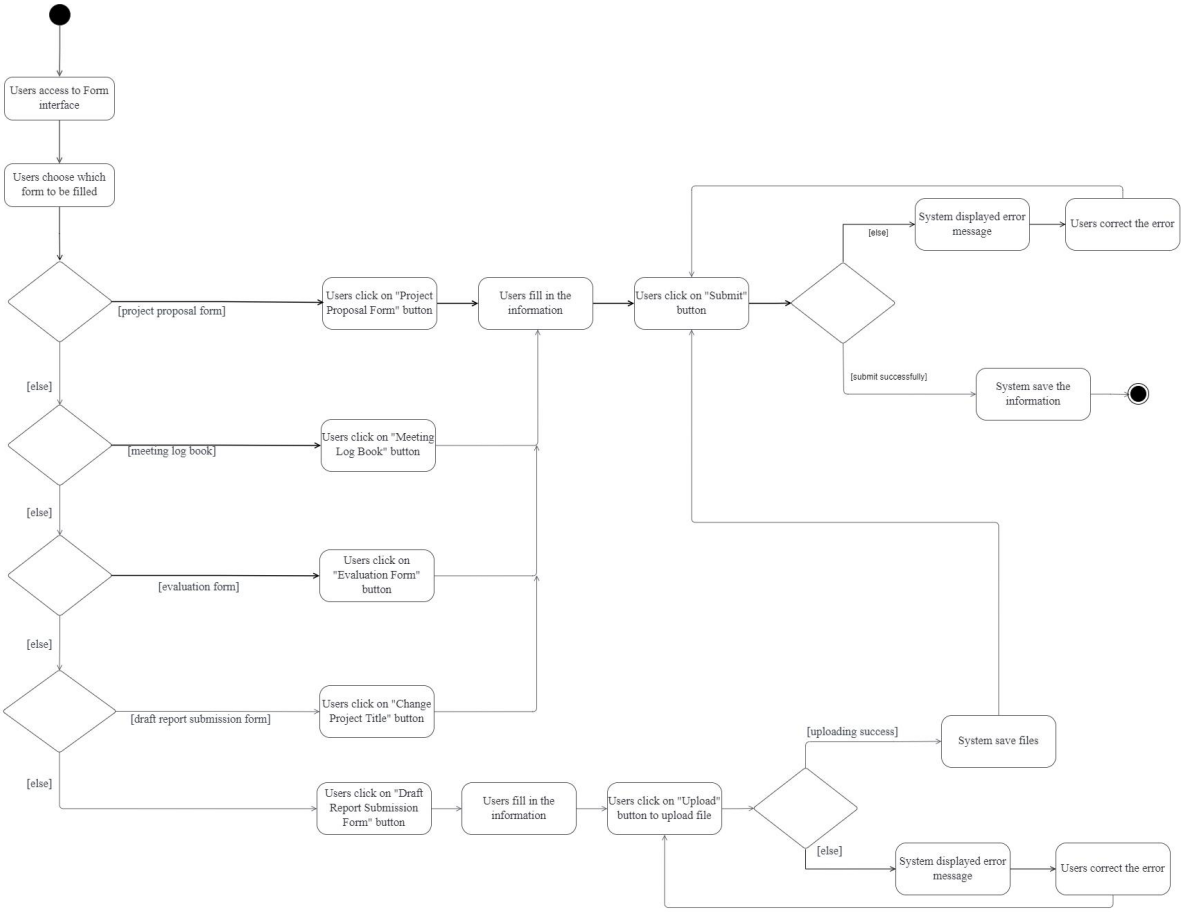

Figure 2.14: Activity diagram for FillingForm use case

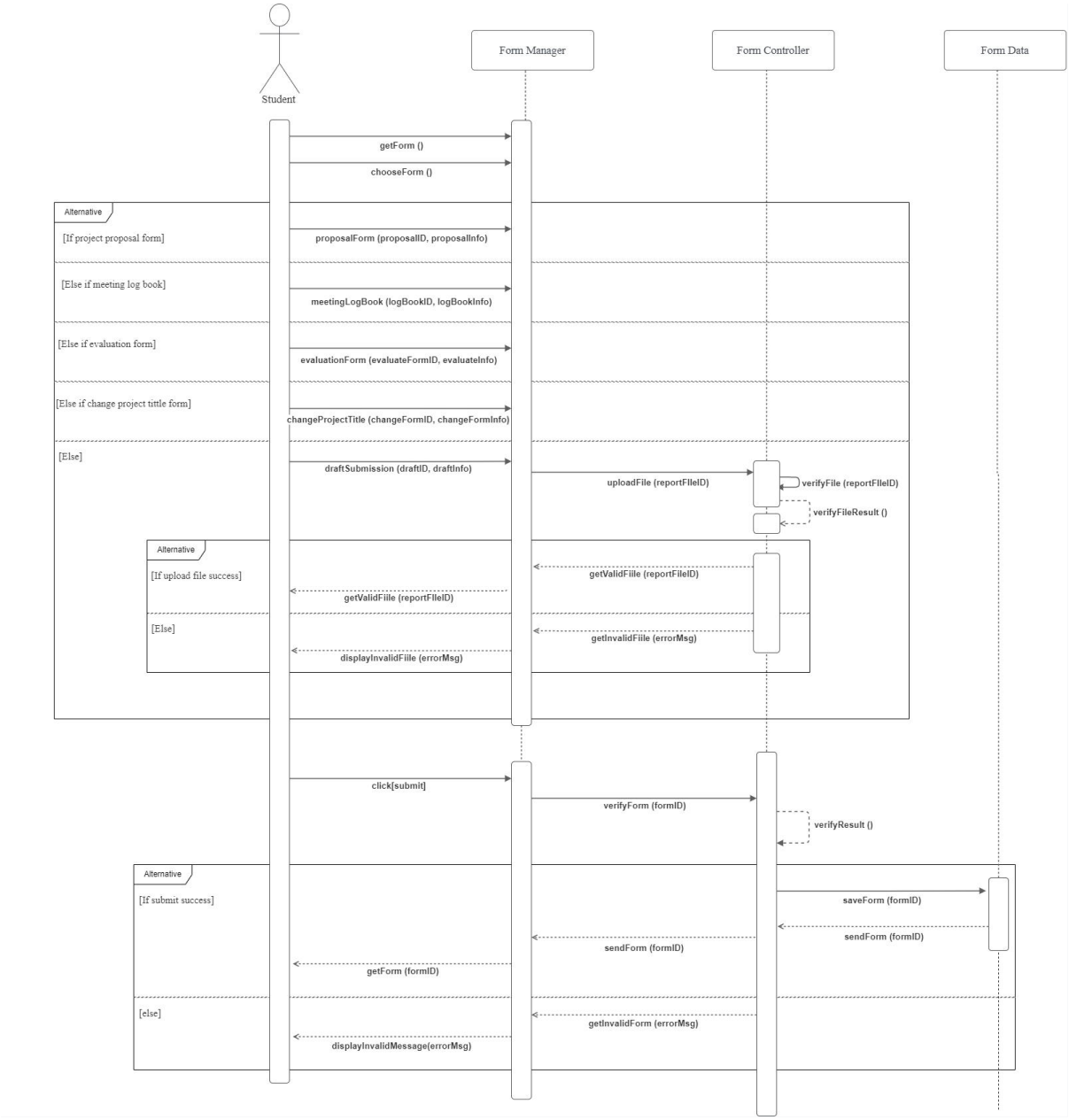

Figure 2.15: Activity diagram for FillingForm use case

2.2.7 UC007: Use Case <MeetingBetweenUser>

Table 2.7: Use case description for MeetingBetweenUser

Use case: MeetingBetweenUser
ID: UC007
Actors: 1. Student 2. Supervisor 3. Coordinator 4. Evaluator
Preconditions: 1. Users have logged into the system. 2. Users have filled in the email information in the system.
Flows of Events: 1. User clicks on the “Meeting” button. 2. User access meeting room interface, user choose a way to start the meeting. 3. If the user has the meeting room id 3.1 User chooses the “Enter Meeting Room ID” option. 3.2 User enters the meeting room id in the “meeting room id box”. 3.3 System tries to find the matched meeting room id. 3.3.1 If the meeting room id invalid, the system displays an error message to the user. 3.3.1.1 User direct to the meeting room interface. 3.3.2 Else, the system will direct the user into a virtual meeting room compatible with the room id. 4. Else, the user does not have a meeting room id. 4.1 User chooses “open a new room”. 4.2 System will assign the user to an empty room with a meeting room id.

4.3 User clicks on the settings icon, and then chooses the share option.

4.4 User copies the meeting room id and then can share it with other users.

5. User starts the meeting in the meeting room.
6. Users can use the microphone, “start video” and the slideshare function by clicking the icon button in the meeting room.
7. If users want to record the meeting
   7.1 user clicks the “Record“ button.
   7.2 System will record the meeting and will save it.
8. After the meeting ends, the user clicks the “leave” button.
9. Users back to the home page.

Postconditions:

1. User left the meeting room.
2. Recorded video will be auto-generated and saved in the system database.
3. A URL link of the recorded video will send the link to the user’s email.
4. Users can view the meeting back when they received the recorded meeting via their email.

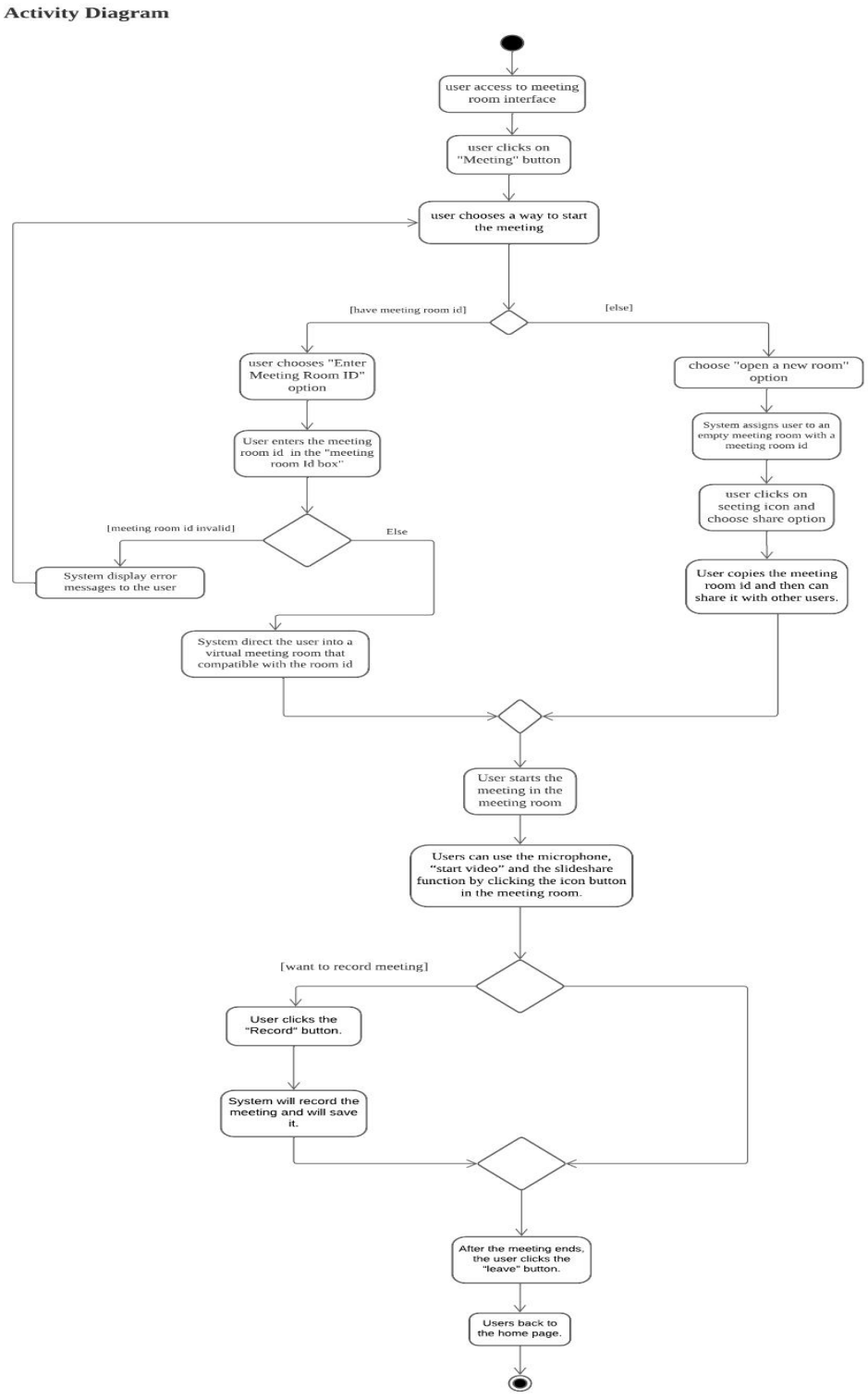

Figure 2.16: Activity diagram for MeetingBetweenUser use case

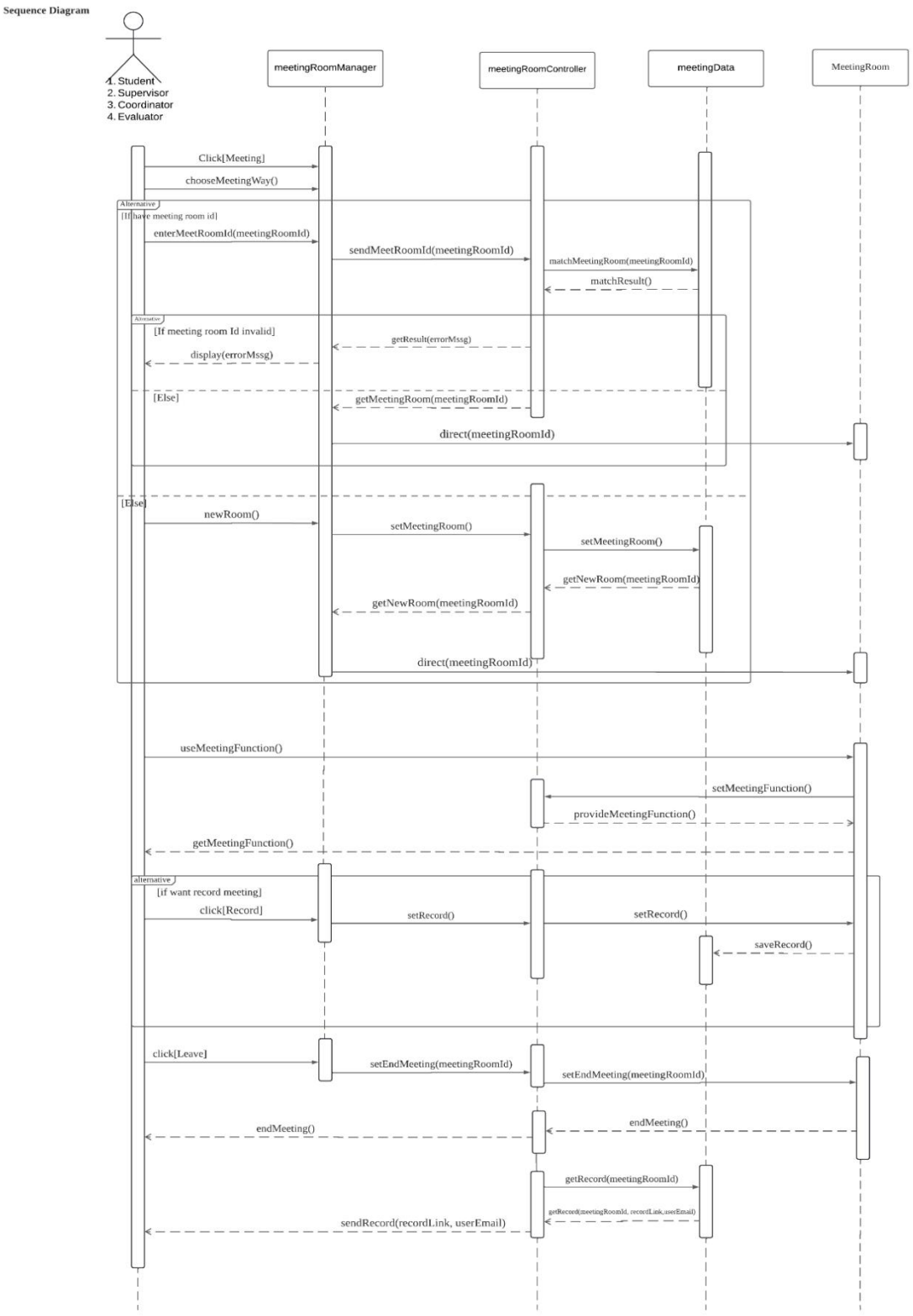

Figure 2.17: Sequence diagram for MeetingBetweenUser use case

2.2.8 UC008: Use Case <SubmitProject>

Table 2.8: Use case description for SubmitProject

Use case: SubmitProject
ID: UC008
Actors: 1. Student
Preconditions: 1. Students have logged into the system. 2. Students have done the project given.
Flows of Events: 1. Users click on the “submission” button to go to the submission interface. 2. Users click the “upload” button. 3. Users choose the file to upload from the device. 4. If upload success, 4.1. The system will display the fileName and fileID that uploaded successfully 5. Else, if the upload failed 5.1. System will display error message 5.2. User repeat steps 2-4. 6. Users click the “submit” button. 7. If the submission success 7.1. System will save the submitted file. 7.2. System will show the time and date of submission. 8. Else, if the submission failed 8.1. System displays an error message “Sorry! Your submission is not successful! Please submit again.” 8.2. Users repeat flows 2 to 5.

Postconditions:
1. Users submitted the file successfully.
2. The evaluator and supervisor can view it from the system.

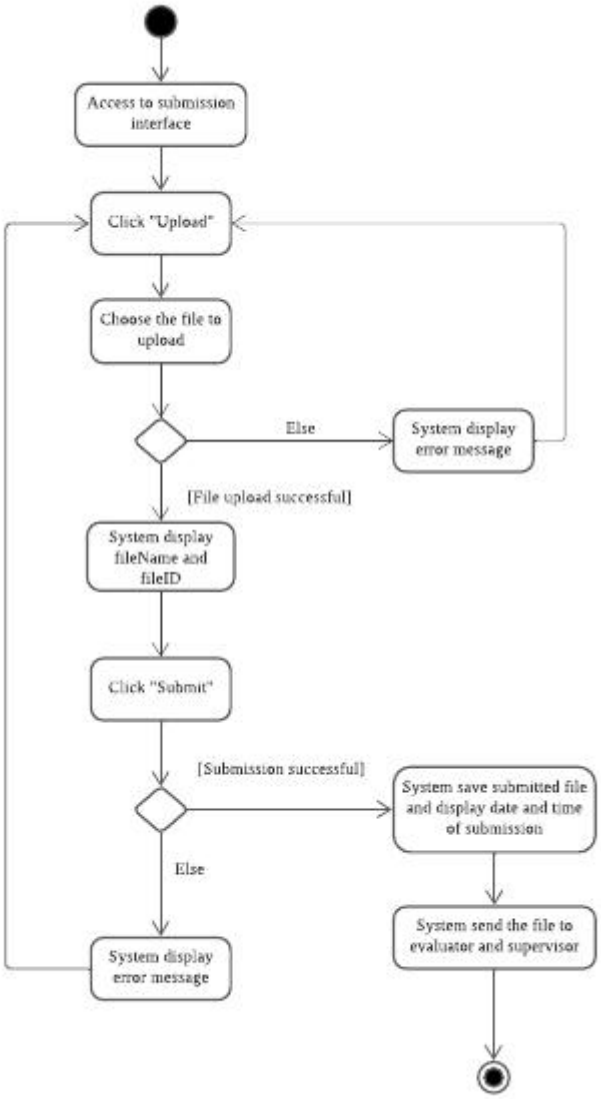

Figure 2.18: Activity diagram for SubmitProject use case

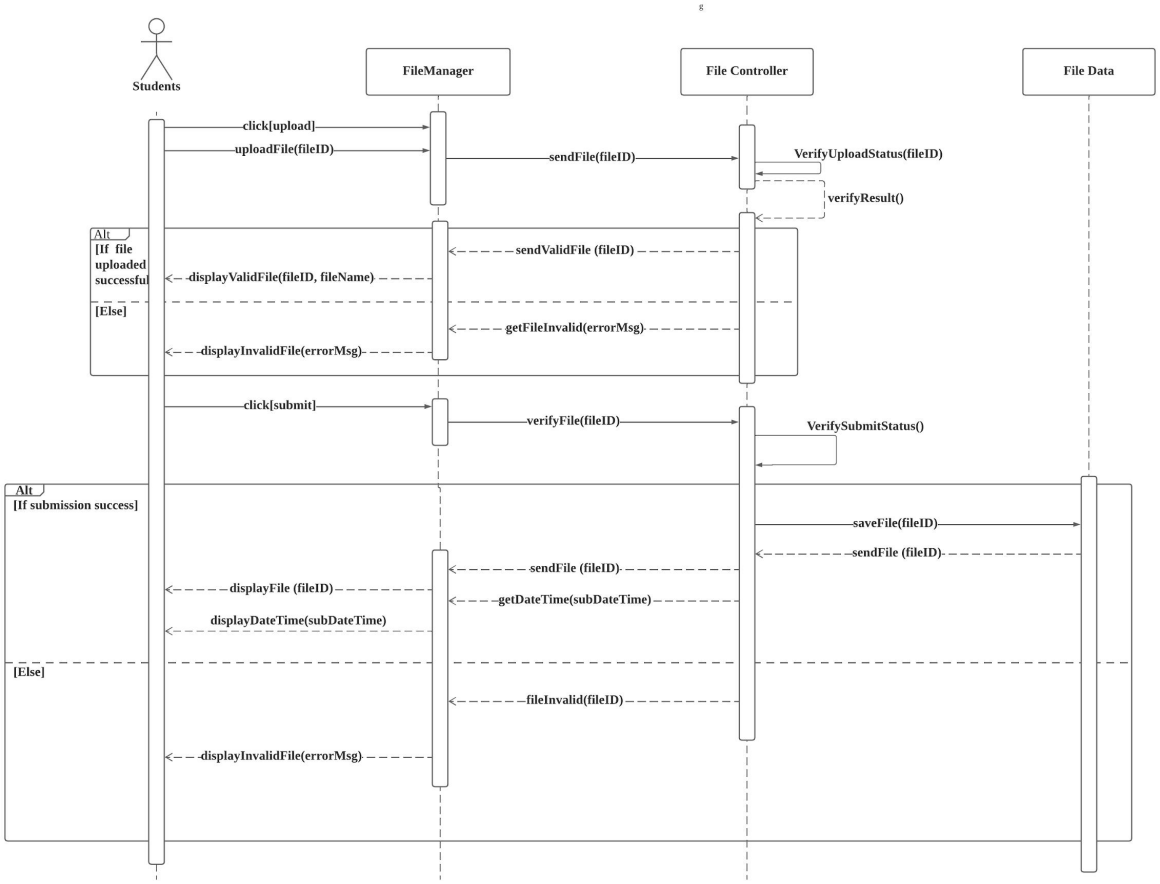

Figure 2.19: Sequence diagram for SubmitProject use case

2.2.9 UC009: Use Case <ChatBetweenUser>

Table 2.9: Use case description for ChatBetweenUser

Use case: ChatBetweenUser
ID: UC009
Actors: 1. Student 2. Supervisor 3. Coordinator 4. Evaluator
Preconditions: 1. User has logged into the system.
Flows of Events: 1. If the users wish to start the conversation 1.1. Users click on the “Conversation” button. 1.2. Chat Box will appear. 1.3. Users choose the person they wish to communicate with. 1.4. Users send messages to that person to start the conversation. 1.5. Users close the chat box. 2. Else if users wish to communicate with Student Care (customer service) 2.1. Users click on the “Conversation” button. 2.2. Users choose the “Student Care” at the bottom right side of the chatbox 2.3. Users send the problems they wish to ask for help 2.4. Users waiting for the reply 3. Else if the users received the message 3.1. Chat Box will pop out, and the message and the username who sent the message will be shown. 3.2. Users reply to the messages. 3.3. Users close the chat box.

4.	Users close the chat box and leave the conversation.
Postconditions: 1.	Users close the chat box and leave the communication session.

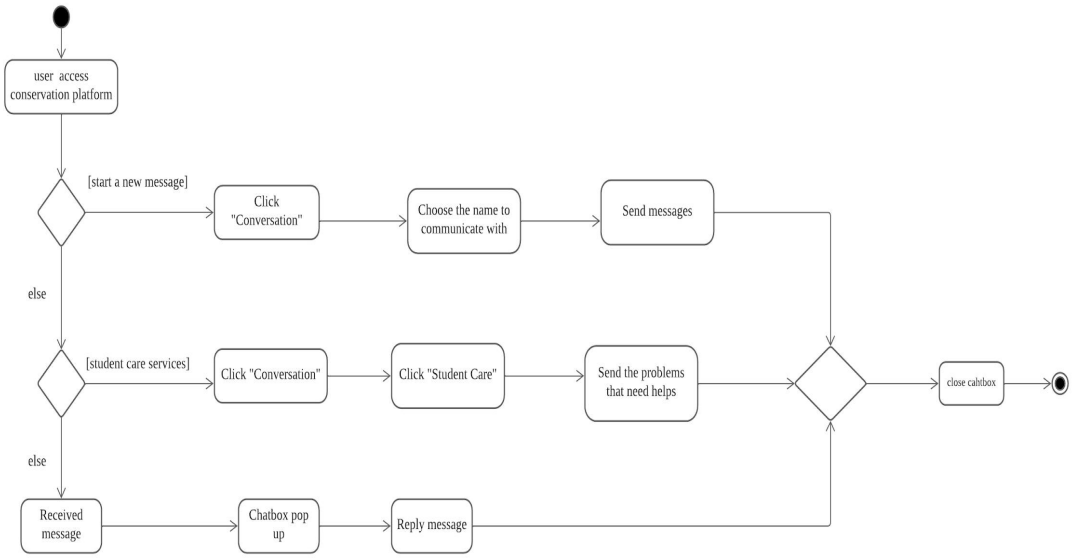

Figure 2.20: Activity diagram for ChatBetweenUser use case

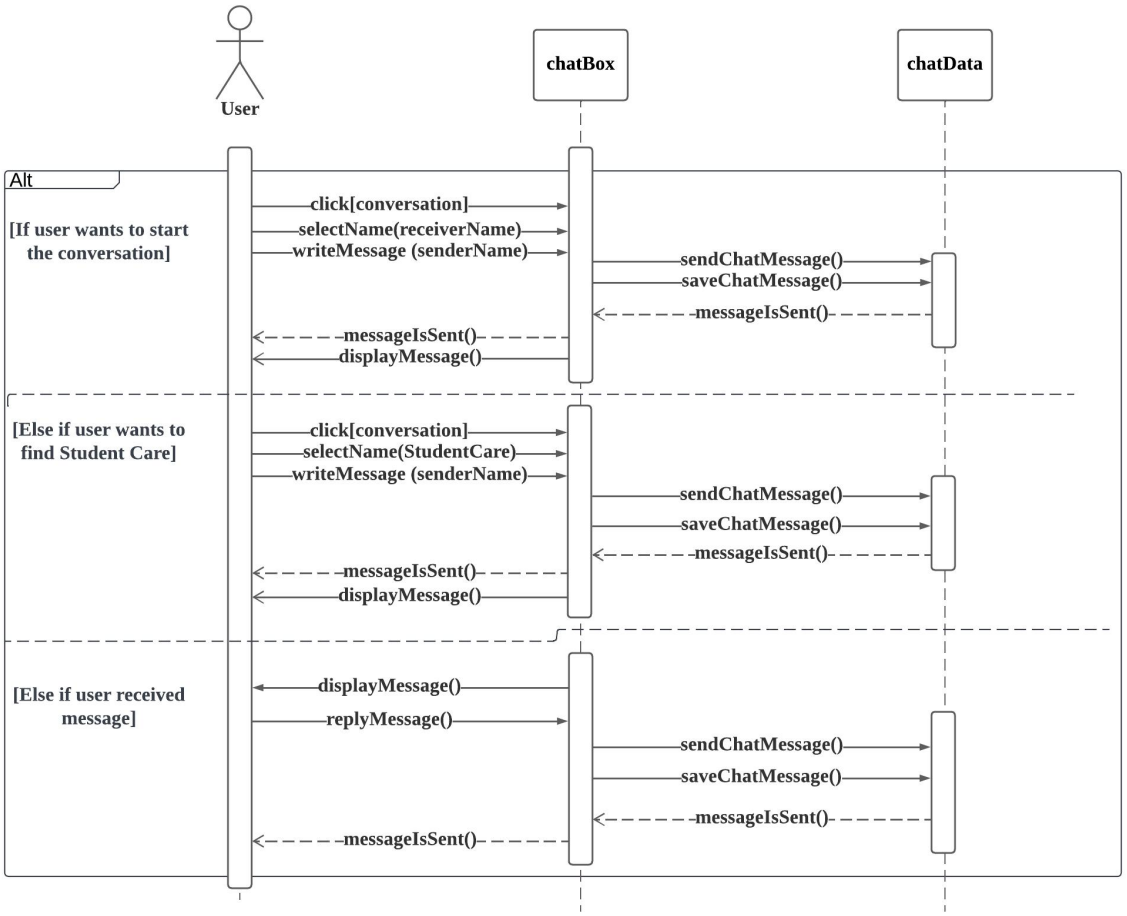

Figure 2.21: Sequence diagram for ChatBetweenUser use case

2.3 Performance and Other Requirements

a. Speed

- The average loading time for the system must be less than 500 milliseconds except if the user is facing an internet connection error.
- The system must save and update the changes done by the users in their accounts within 10 seconds.
- The system must send the notification set by the user within 30 seconds to the user's email when it is triggered.

b. Portability

- Users can access the system using different types of devices as long as there is an internet connection.

c. Usability

- The system will use the English language as the main communication language as it is the universal language.
- The settings option will represent the icon with a real-life metaphor so that users can understand the function of the option when they see it.

d. Efficiency

- The system provides easy and clear steps for users to perform any operation in the system.

2.4 Design Constraints

One of the design constraints is we need to connect to the UTM database for the registration function. To register in our system, the user must have a unique id that matches and exists in the UTM database. This is to ensure only UTM students, staff, and companies that hired our students can use the system.

Besides that, in the meeting function, the user cannot directly share the meeting room id with other users. They need to copy the id themselves and share it with other users by themselves.

Lastly, the notification function of our system also has some limitations. As our system is a web-based system, the notification can only be sent via email and will not pop up directly to the user.

2.5 Software System Attributes

a. Maintainability
    i. Software is able to be updated and evolved to meet the changing needs of users.

b. Efficiency
    i. The processing time should be faster and more responsive quickly.

c. Usability
    i. The system developed should be easy to understand and easy to use by users.

d. Reliability
    i. The mistakes for the system should be reduced as much as possible.

e. Effectiveness
    i. The system needs to complete program maintenance, modification, and resource consumption in a short time, so that high user satisfaction can be obtained.

3. System Architectural Design

3.1. Architecture Style and Rationale

The architecture style used in the One Touch FYP System is layer architecture. The layered architecture was chosen because it divides the system into layers, each with its own set of functions. There are three layers applied in the system. The lowest layer represents the basic services that may be used throughout the system, as each layer provides services to the layer above it.

The presentation layer is the first layer and is responsible for all the graphical user interface (GUI) and browser communication logic. It is a user interface layer that allows people to interact with the system. This layer will display information in a specific format that users will be able to interact with by viewing or clicking through the interface.

Second layer is business logic which accepts user requests from the browser, processes them, and determines data access paths. The business layer contains the workflow that guides data and requests through the backend.

The database layer is the central location that receives all data calls and provides access to the application's persistent storage. Since the database layer is tightly coupled to the business layer, the logic knows which database to communicate with, further simplifying the data fetching process.

For further explanation, the user must log in before access to the One Touch FYP System. The presentation layer will display the login form for the user to enter their ID and password. Then, business logic will determine the sequence of events that happens during login. Lastly, the database layer will compare the ID and password entered by the user and pass the verification result back to the presentation layer.

The One Touch FYP system applies a layered architecture because it already has an existing system. Therefore, it is more appropriate to add new facilities to an existing system by using a layered architecture. In addition, the multi-layer architecture can replace the entire layer while preserving the interface, reducing the cost of replacing the user interface. The

One Touch FYP system may require external redundancy features such as user form ID, so applying a layer architecture is able to increase system reliability.

3.2. Component Model

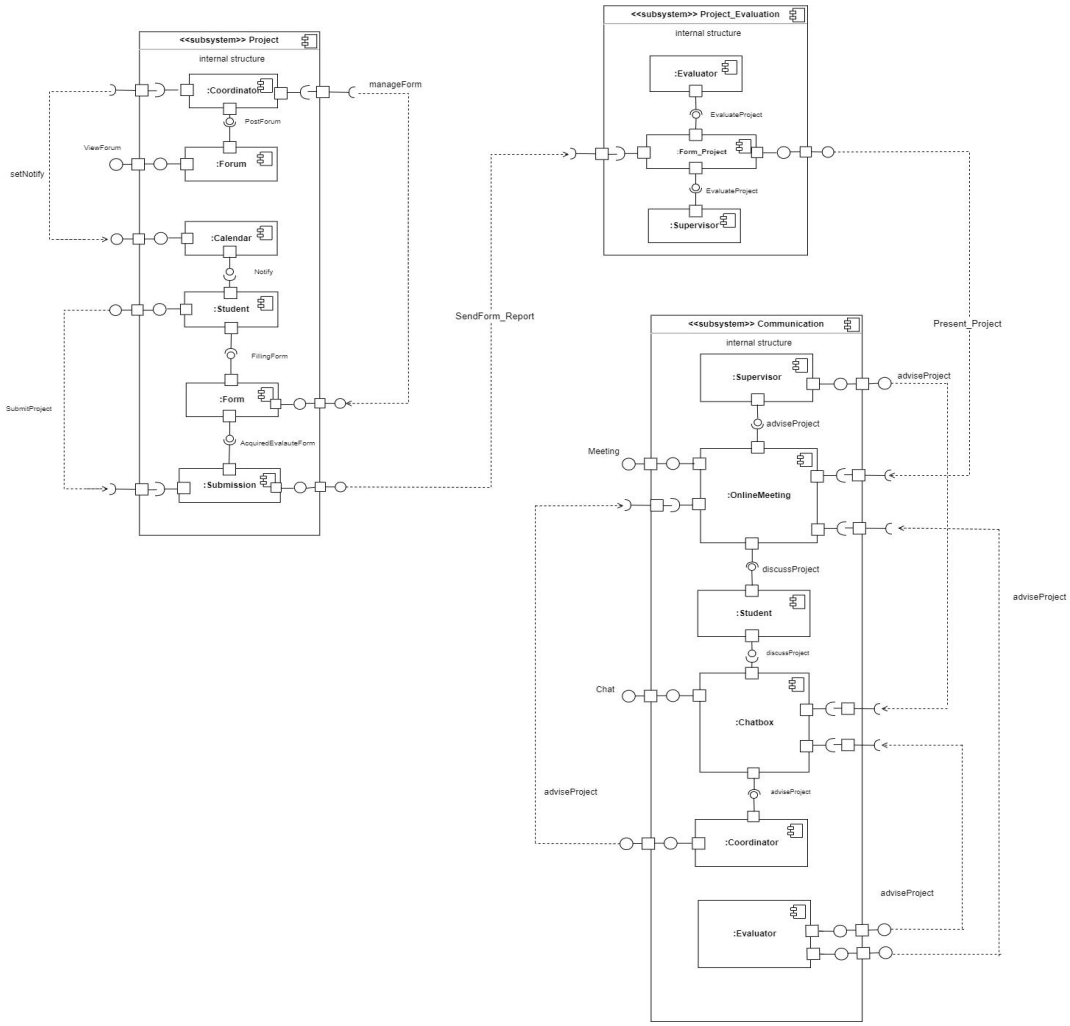

Figure 3.1: Component Diagram of <One Touch FYP System>

There are three high-level subsystems in the One Touch FYP System, which are Project, Project Evaluation and Communication subsystems.

In the Project subsystem, there are six components, which are **Coordinator**, **Forum**, **Calendar**, **Student**, **Form** and **Submission**. The **Coordinator** can *PostForum* in the **Forum** component. When a user wants to *viewForum*, they can view the post in the **Forum** component. Furthermore, the **Coordinator** can also *setNotify* in the **Calendar** component by receiving the activity list of students from the **Calendar** component. The Calendar

component will *notify* the **Student**. For the **Student**, they can *FillingForm* provided by the **Form** component. The **Coordinator** *manageForm* provided by the Form component. If the **Student** has finished their project, they can *SubmitProject* in the **Submission** component . The **Submission** component will *AcquiredEvaluatedForm* from the **Form** component. The **Submission** component has a relationship with the Project_Evalaution subsystem. The **Submission** component will *SendForm_Report* to the **Form_Project** component in the Project_Evaluation subsystem as input.

In the Project_Evaluation subsystem, there are three components, which are **Evaluator**, **Form_Project** and **Supervisor**. The **Form_Project** component will provide the input to the **Evaluator** and **Supervisor** to *EvalauteProject*. The **Form_Project** component has a relationship to the Communication subsystem. After the **Evaluator** and **Supervisor** have evaluated the project, they can watch the **Present_Project** in the **OnlineMeeting** component.

Lastly, for the Communication subsystem, there are six components, which are **OnlineMeeting**, **Student**, **Coordinator, Supervisor**, **Evaluator** and **Chatbox**. When the user needs to have a meeting, they can use the **OnlineMeeting** component. Otherwise, the user can use the **Chatbox** component to chat with each other. Besides that, **Student** *discussProject* in the **OnlineMeeting** component or in the **Chatbox** component. While for the other users, they can adviseProject to the **Student** in the **OnlineMeeting** or **Chatbox** component. The **OnlineMeeting** component have relationship with the Project_Evaluation subsystem, when the **Evaluator** and **Supervisor** have evaluated the project, they can watch the **Present_Project** by the **Student** in the **OnlineMeeting** component.

4. Detailed Description of Components

4.1. Complete Package Diagram

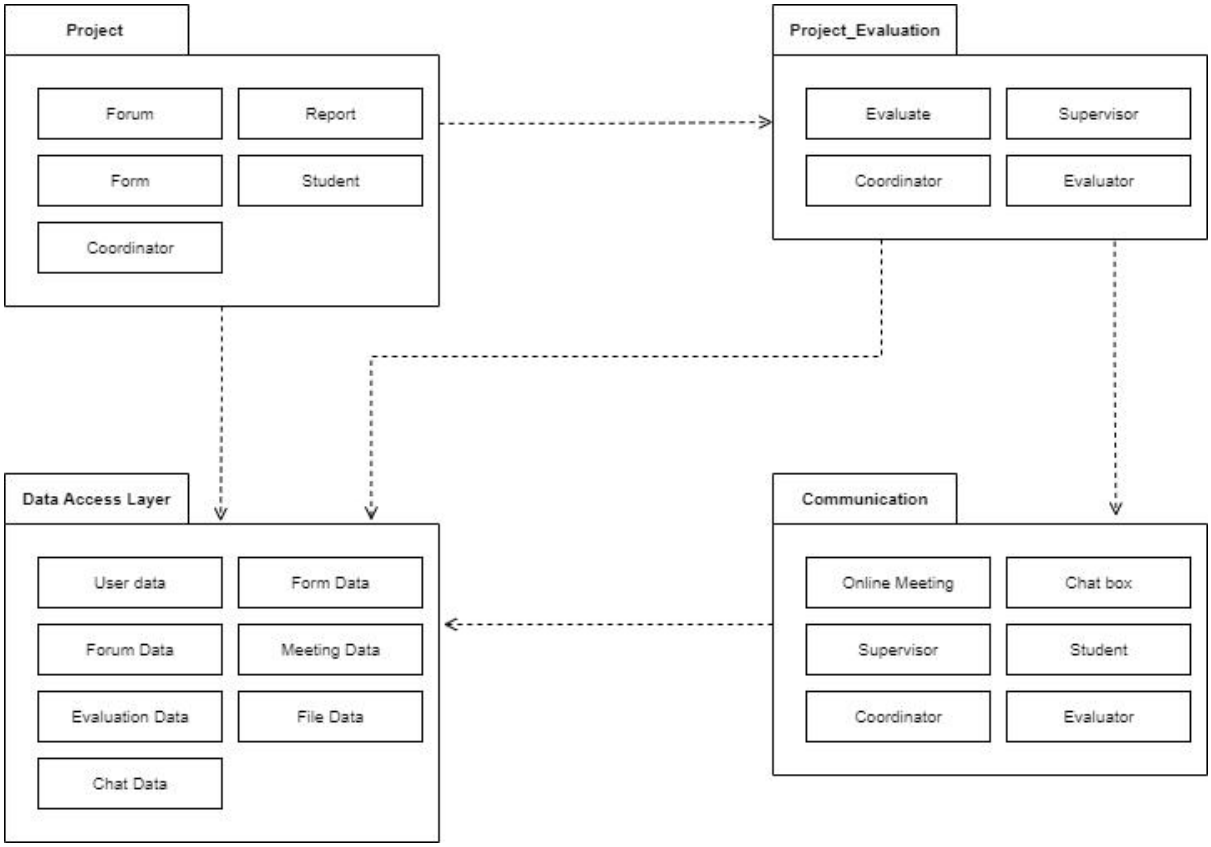

Figure 4.1: Package Diagram for <Name of the System>

4.2. Detailed Description

4.2.1. P001: <Project> Subsystem

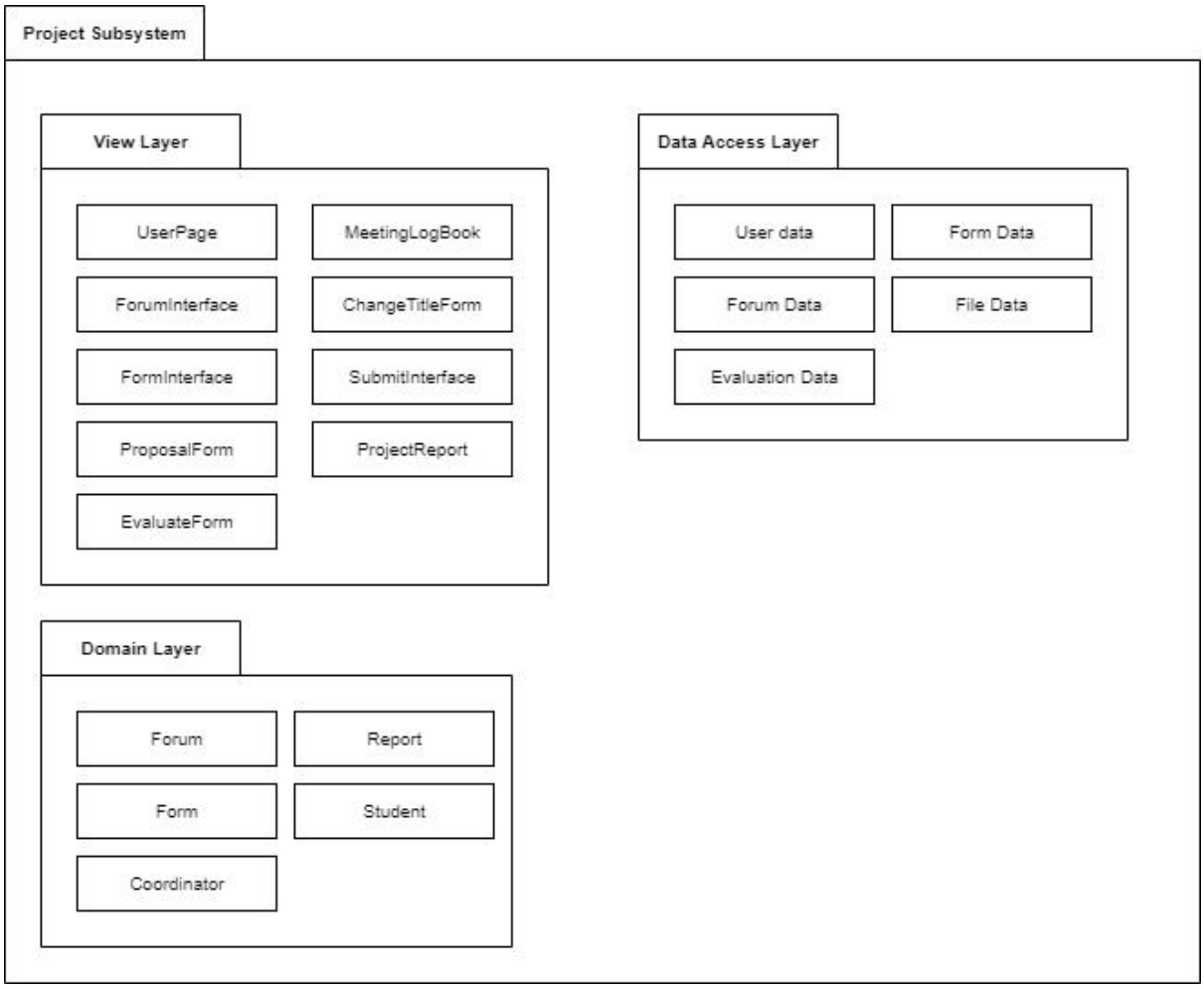

Figure 4.2: Package Diagram for <Project> Subsystem

4.2.1.1. Class Diagram

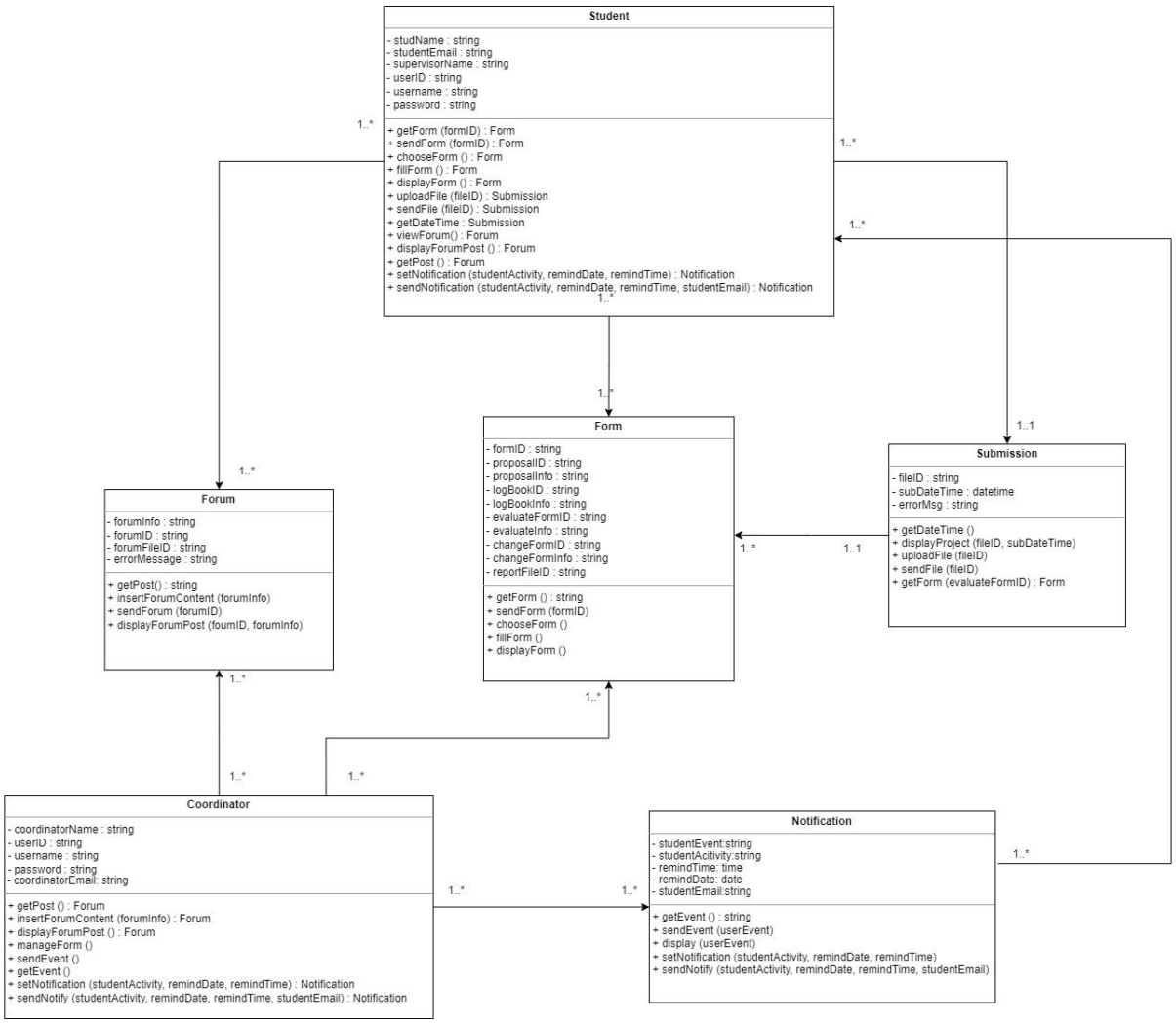

Figure 4.3: Class Diagram for <Project> Subsystem

List of Methods from the Class Diagram for <Project> Subsystem

Entity Name	Notification
Method Name	1. getEvent 2. sendEvent 3. display 4. setNotification

	5. sendNotify
Input	-
Output	Send notification to student via email.
Algorithm	1. Start. 2. Users choose the “calendar” button. 3. Users choose the “setting” option in the calendar interface. 4. If the user is the coordinator, the system will display a list of student events for the coordinator. 4.1 The coordinator selects the student’s activities they wish to notify the students. 4.2 The coordinator sets the date and time they wish to notify the students. 5. Else, the system displays a list of events the student has. 5.1 The student selects the activities they wish to be notified of. 5.2 The student sets the date and time they wish to be notified. 1. End.

Entity Name	Forum
Method Name	1. insertForumContent. 2. getPost. 3. displayForumPost. 4. End.
Input	-
Output	The post will be displayed by the system.

Algorithm	1. Start. 2. Users choose the “Forum” button. 3. The system will display a list of posts. 4. If the user is a coordinator, 3.1 User clicks on the “Post” button. 3.2 User creates the content to post on the forum. 3.3 If the user upload post success, 3.3.1 System display the post. 3.4 Else, the user reposts the post. 5. Else, the user chooses the post wish to view. 6. The system will display the post chosen by the user for view. 7. End.

Entity Name	Submission
Method Name	1. uploadFile 2. sendFile 3. getDateTime
Input	-
Output	The file will be submitted and saved to the system.
Algorithm	1. Start. 2. Users choose “Submission”. 3. Users click the “Upload” button. 4. Users choose the file to upload. 5. If the file is uploaded success, 5.1 The system will display the uploaded file. 6. Else, the users need to reupload the file. 7. Users click the “Submit” button.

	8. If the file is submitted success, 8.1 The system will save the file and display the submitted filename. 9. Else, the users need to resubmit file. 10. The system will display the date and time of success submitted. 11. End.

Entity Name	Form
Method Name	1. getForm 2. displayForm 3. chooseForm 4. fillForm 5. sendForm
Input	-
Output	The system will display the form filled by the user.
Algorithm	1. Start. 2. Users choose the “Form” button. 3. The system will display a list of forms to be chosen. 4. Users choose the form to fill in. 5. Users click “Submit” after filling in. 6. If submitted success, 5.1 System will display the form submitted. 7. Else, users need to resubmit the form. 8. End.

Entity Name	Student
Method Name	1. getForm 2. chooseForm 3. sendForm 4. displayForm
Input	-
Output	The student will fill in the form and submit
Algorithm	1. Start. 2. Users choose the “Form” button. 3. Users select the form to fill. 4. Users fill in the form. 5. Users click the “Submit” button. 6. End.

Entity Name	Student
Method Name	1. uploadFile 2. sendFile 3. getDateTime
Input	-
Output	The file will be uploaded to be submitted.
Algorithm	1. Start. 2. Users choose the “Submission” button. 3. Users select the file to upload. 4. Users click the “Submit” button. 5. End.

Entity Name	Student
Method Name	1. viewForum 2. getPost 3. displayForumPost
Input	-
Output	The post in the forum will be displayed.
Algorithm	1. Start. 2. Users choose the “Forum” button. 3. Users select the forum post to view. 4. End.

Entity Name	Student
Method Name	1. displayEvent 2. sendEvent 3. setNotification 4. sendNotify
Input	Activity set by the coordinator.
Output	System will send notification to student through email
Algorithm	1. Start 2. Users choose “Calendar”. 3. Users choose “Setting”. 4. Users select the activity. 5. Users set the date and time that wish to be notified. 6. Users click “Save” to save the setting.

	7. End.

Entity Name	Coordinator
Method Name	1. displayEvent 2. sendEvent 3. setNotification 4. sendNotify
Input	-
Output	The system will send notification to student by email
Algorithm	1. Start. 2. Users choose “Calendar”. 3. Users choose “Setting”. 4. Users set the event for the student. 5. Users set the date and time that wish to be notified. 6. Users click “Save” to save the setting. 7. End.

Entity Name	Coordinator
Method Name	1. insertForumContent 2. getPost 3. displayForumPost
Input	-
Output	Post with content will be uploaded to the forum.

Algorithm	2. Start. 3. Users choose the “Forum” button. 4. Users click the “Post” button. 5. Users insert the forum content to the post. 6. Users click submit. 7. End.

4.2.1.2. Sequence Diagram

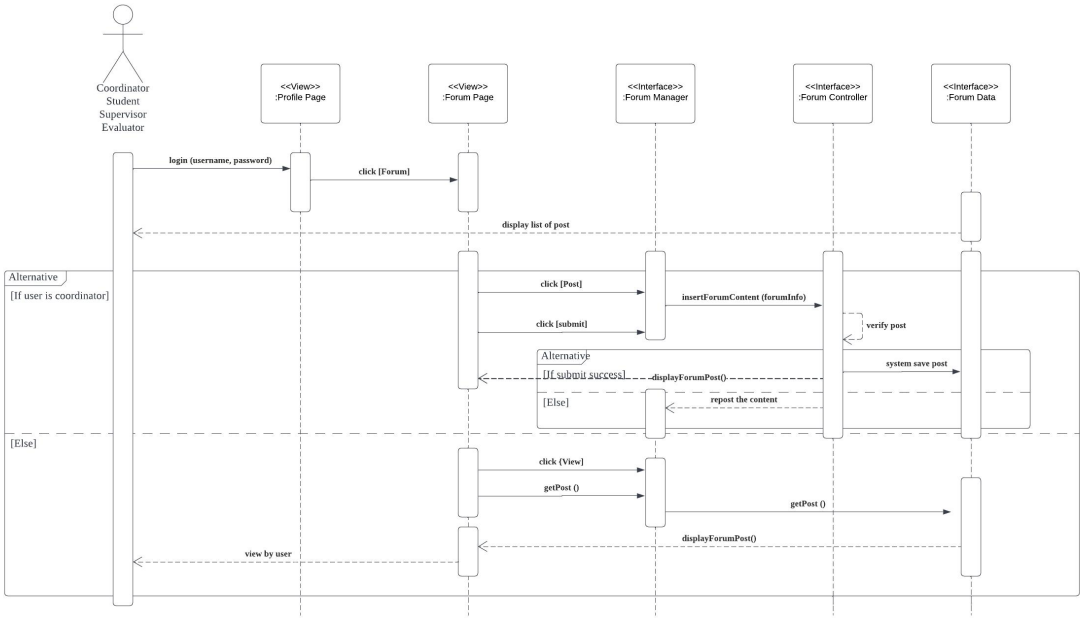

Figure 4.4: Sequence Diagram for <Post and View Forum Scenario>

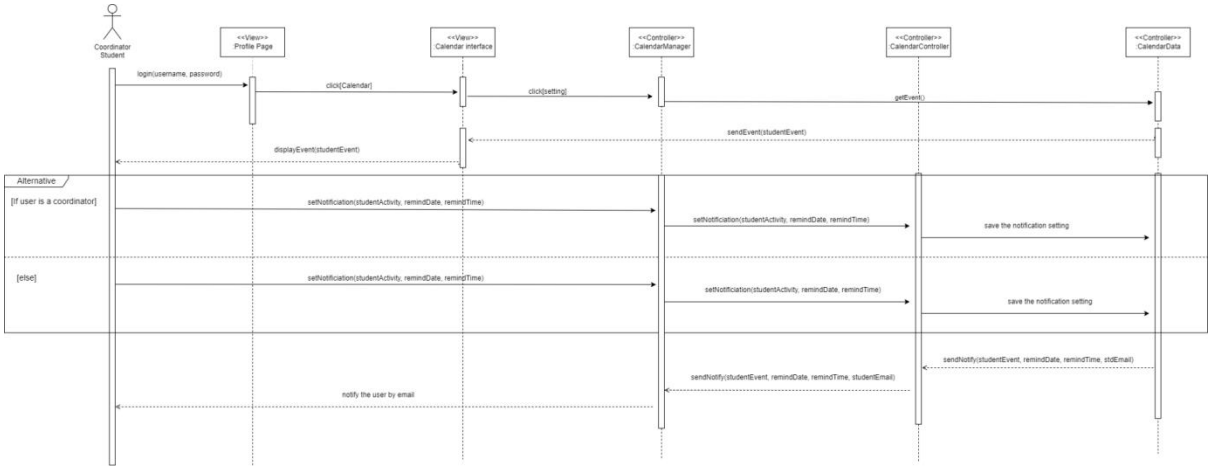

Figure 4.5: Sequence Diagram for <Set Up Notification Scenario>

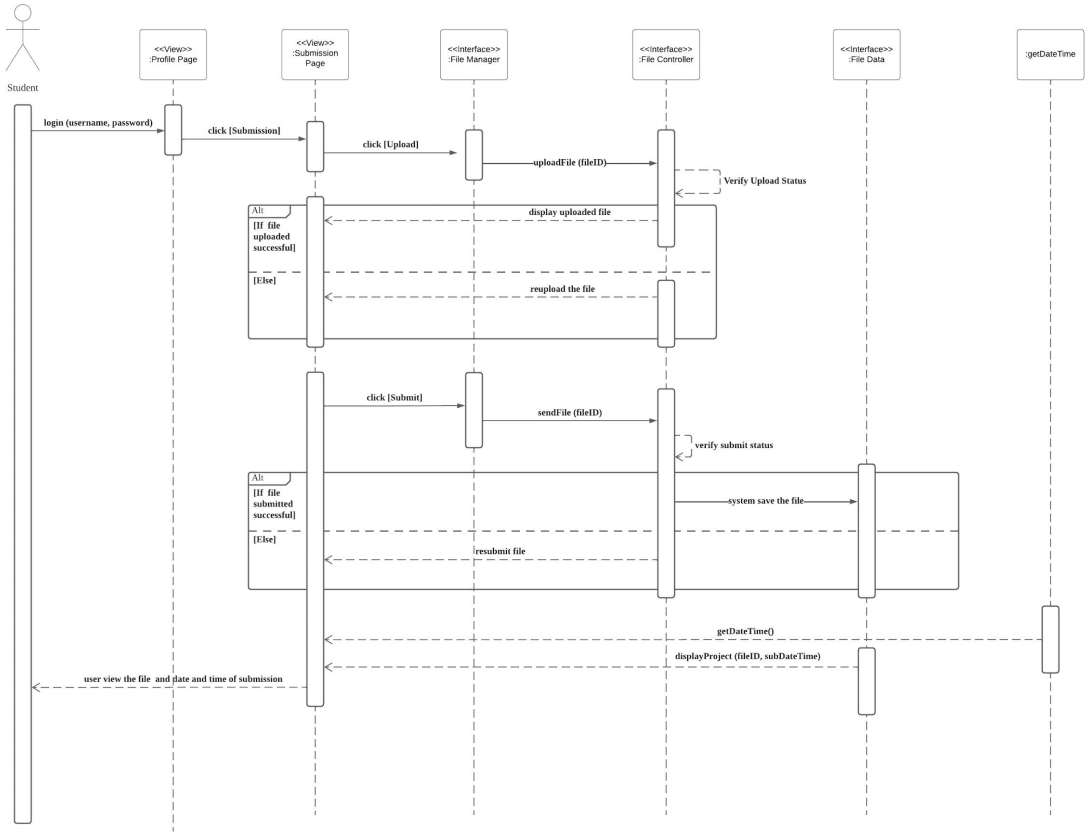

Figure 4.6: Sequence Diagram for <Submit Project Scenario>

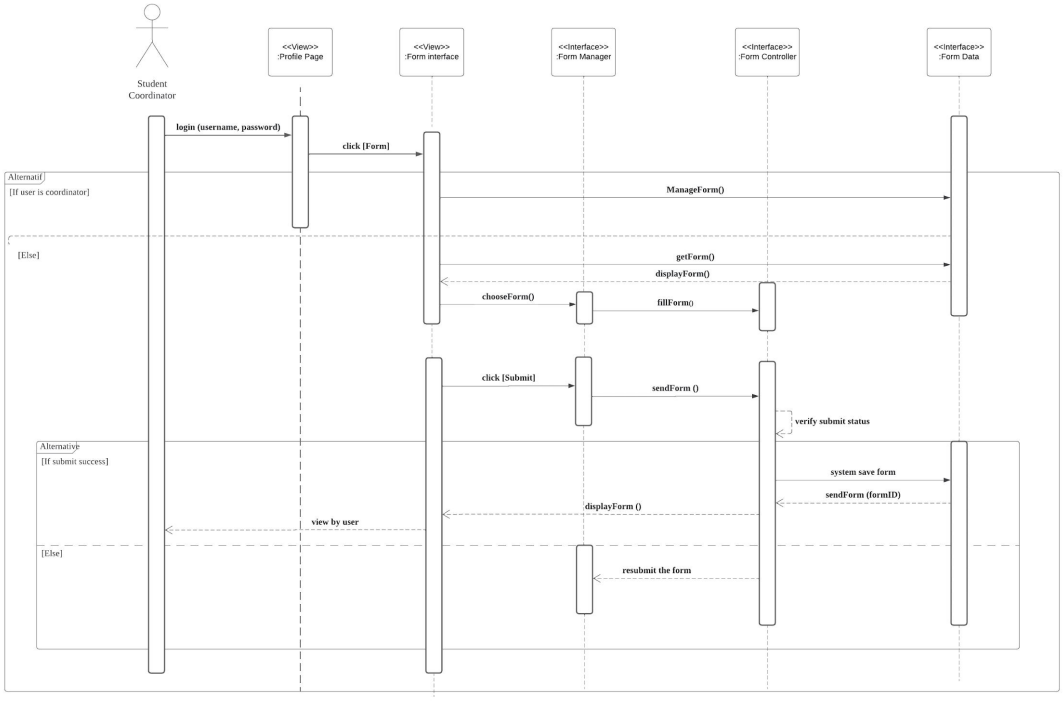

Figure 4.7: Sequence Diagram for <Manage and Fill Form Scenario>

4.2.2. P002: <Project_Evaluation> Subsystem

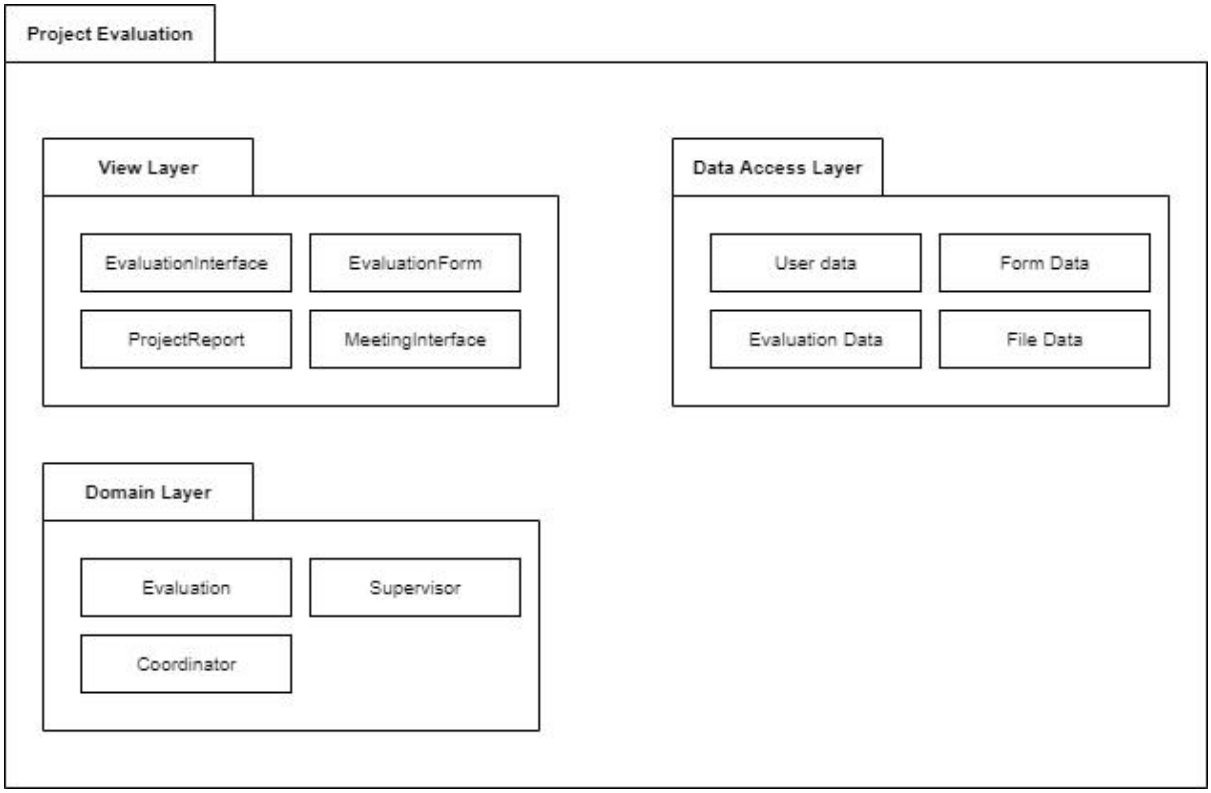

Figure 4.8: Package Diagram for <Project_Evaluation> Subsystem

4.2.2.1. Class Diagram

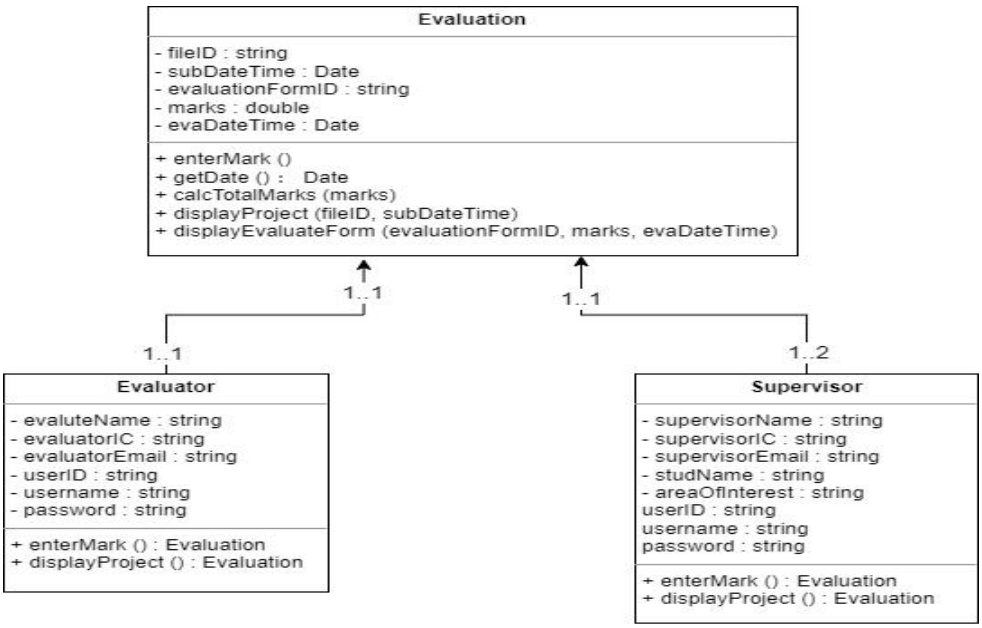

Figure 4.9: Class Diagram for <Project_Evaluation> Subsystem

List of Methods from the Class Diagram for <Project_Evaluation> Subsystem

Entity Name	Evaluator
Method Name	1. enterMark 2. displayProject
Input	-
Output	The marks will be displayed by the system.
Algorithm	1. Start. 2. Users choose the “Evaluation” button. 3. Users select one of the student’s projects. 4. Users click the “View” button. 5. The content of the project will be displayed by the system. 6. Users click on the “Evaluate” button. 7. Users enter marks. 8. If enter marks success, a. System save marks. 9. Else, users re-enter marks. 10. Users click the “Submit” button. 11. End.

Entity Name	Supervisor
Method Name	1. enterMark 2. displayProject
Input	-
Output	The marks will be displayed by the system.
Algorithm	1. Start. 2. Users choose the “Evaluation” button. 3. Users select one of the student’s projects.

	4. Users click the “View” button. 5. The content of the project will be displayed by the system. 6. Users click on the “Evaluate” button. 7. Users enter marks. 8. If enter marks success, 8.1 System save marks. 9. Else, users re-enter marks. 10. Users click the “Submit” button. 11. End.

Entity Name	Evaluation
Method Name	1. enterMark 2. getDate 3. calcTotalMarks 4. displayProject 5. displayEvaluateForm
Input	1. Evaluation Form entered by students. 2. Students’ project reports.
Output	1. The marks will be displayed by the system. 2. The date and time evaluating the marks will be displayed by the system. 3. The total students’ project marks will be displayed by the system.
Algorithm	1. Start. 2. Users choose the “Evaluation” button. 3. Users select one of the student’s projects. 4. Users click the “View” button. 5. The content of the project will be displayed by the system.

	6. Users click on the “Evaluate” button. 7. Users enter marks. 8. If enter marks success, 8.1 System save marks. 9. Else, re enter marks. 10. Users click the “Submit” button. 11. System gets the current date and time from the getDate method. 12. System gets the total marks of students’ projects by calculating the sum of marks entered by supervisors and evaluators. 13. End.

4.2.2.2. Sequence Diagram

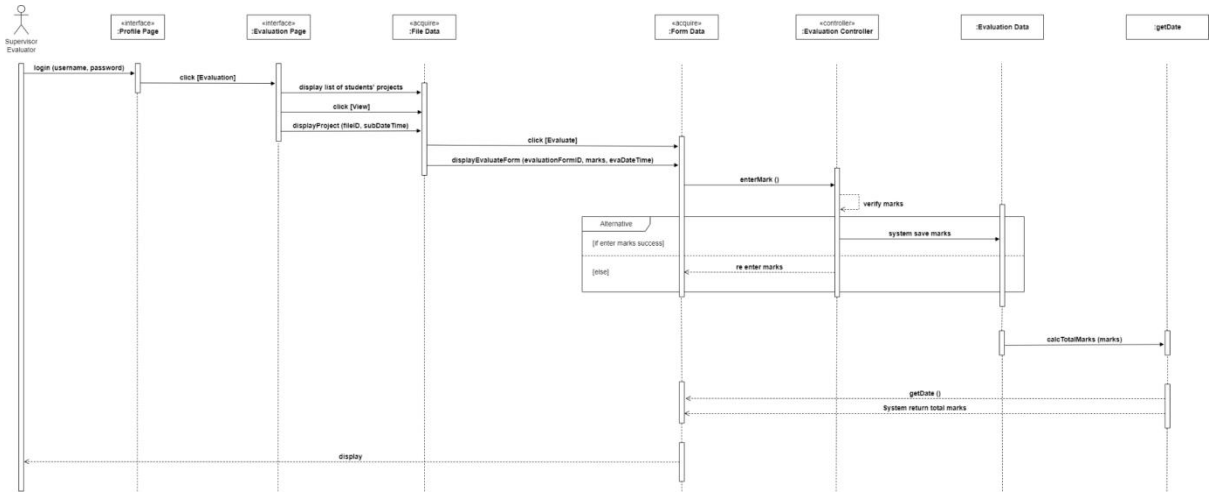

Figure 4.10 : Sequence Diagram for <Evaluate Project Scenario>

4.2.3. P003: <Communication> Subsystem

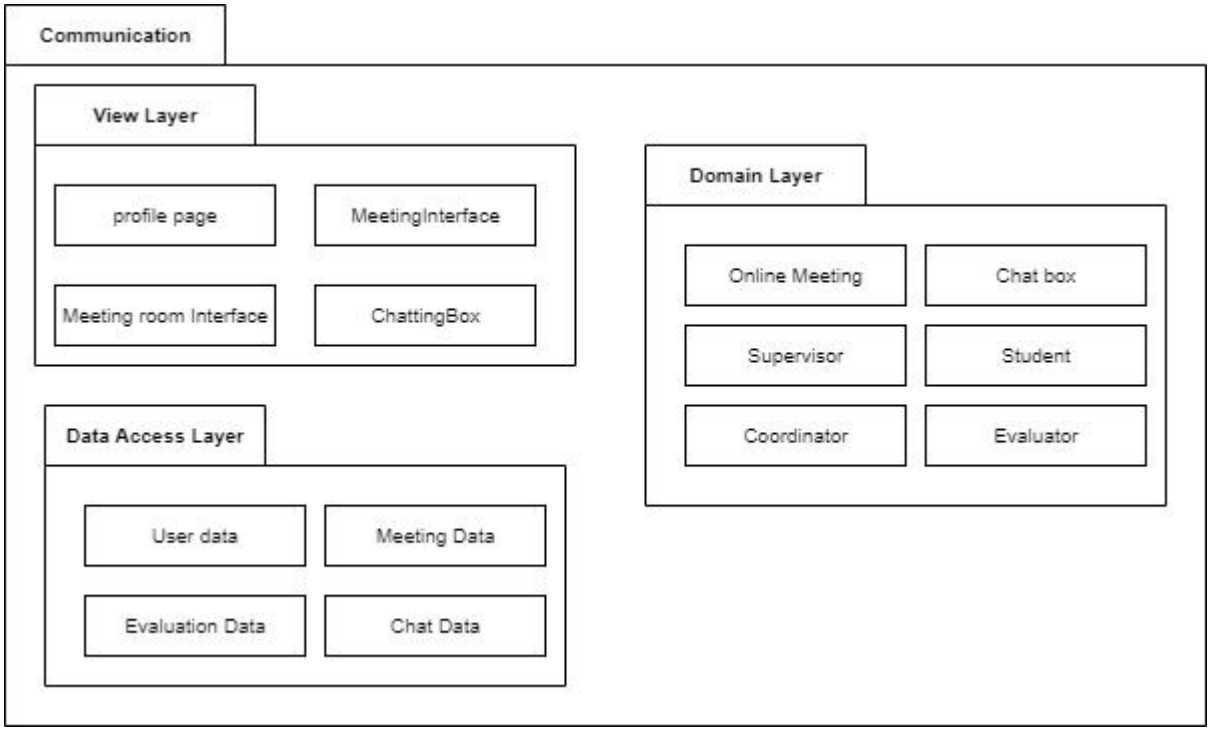

Figure 4.11: Package Diagram for <Communication> Subsystem

4.2.3.1. Class Diagram

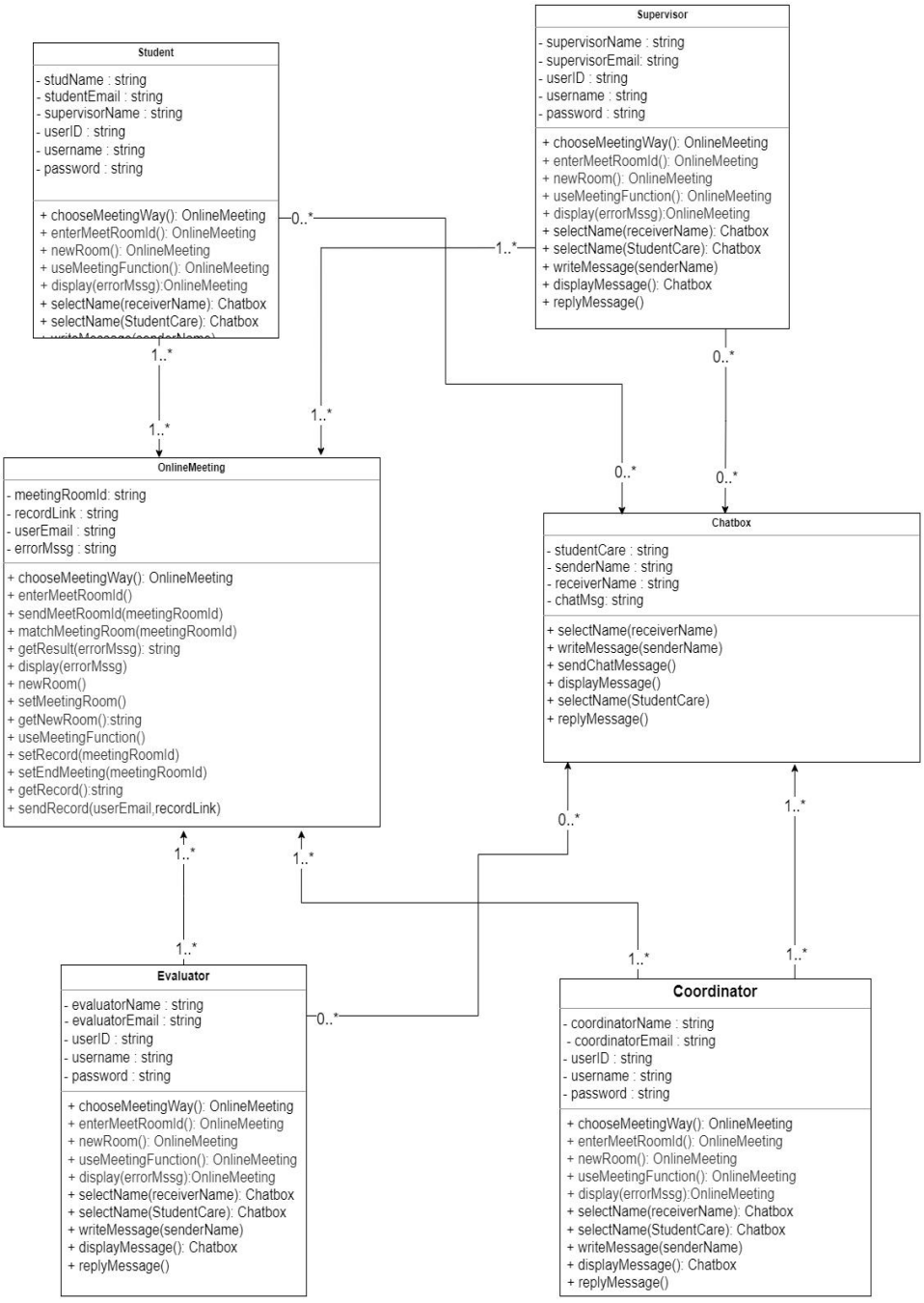

Figure 4.12: Class Diagram for <Communication> Subsystem

List of Methods from the Class Diagram for <Communication> Subsystem

Entity Name	Online Meeting
Method Name	1. chooseMeetingWay 2. enterMeetRoomId 3. sendMeetRoomId 4. matchMeetingRoom 5. display
Input	-
Output	-
Algorithm	1. Start 2. User chooses the enter meeting room id option. 3. Read the way the user wants to start the meeting. 4. Read the meeting room id of the user entered. 5. Find the matched meeting room based on the id 5.1. If meeting room id not found, 5.1.1. Display error message to the user 5.2. Else 5.2.1. the user direct to the meeting room 6. End

Method Name	1. chooseMeetingWay 2. newRoom 3. setMeetingRoom 4. getNewRoom
Input	-

Output	1. User get a new room id
Algorithm	1. Start 2. User choose “new meeting room” option 3. Read the way the user wants to start the meeting 4. Assign a new meeting room id to the user 5. User get the new meeting room id 6. End

Method Name	1. useMeetingFunction
Input	-
Output	-
Algorithm	1. Start 2. User choose the meeting room function in the online meeting platform 3. End

Method Name	1. setRecord 2. getRecord 3. sendRecord 4. setEndMeeting
Input	-
Output	1. User get the URL link of the meeting recorded video
Algorithm	1. Start 2. User choose “record” option 3. System generate URL link of the meeting recorded video to the user

	4. System send the recorded video URL link based on the user's email 5. End

Method Name	1. setEndMeeting
Input	-
Output	-
Algorithm	1. Start 2. User choose "leave" option 3. System end the meeting 4. End

Entity Name	Chatbox
Method Name	1. selectName 2. writeMessage 3. sendChatMessage 4. displayMessage
Input	-
Output	1. message successfully sent 2. display the message write by the user
Algorithm	1. Start 2. User select to start the conservation with other user 3. User write the message

	4. Message is sent 5. The message is displayed 6. End

Method Name	1. selectName 2. writeMessage 3. sendChatMessage 4. displayMessage
Input	-
Output	1. message successfully sent 2. display the message write by the user
Algorithm	1. Start 2. User select to chat with customer service 3. User write the message 4. Message is sent 5. The message is displayed 6. End

Method Name	1. displayMessage 2. replyMessage 3. sendChatMessage
Input	-
Output	1. message successfully sent 2. display the message write by the user

Algorithm	1. Start 2. Display the received message to the user 3. User write the reply message 4. Message is sent 5. End

Entity Name	Student
Method Name	1. chooseMeetingWay
Input	-
Output	-
Algorithm	1. Start 2. User select meeting way 3. End

Method Name	1. enterMeetRoomId 2. display
Input	-
Output	-
Algorithm	1. Start 2. User enter meeting room id 3. If valid meeting room id 3.1. view error message 4. End

Method Name	1. newRoom
Input	-
Output	-
Algorithm	1. Start 2. User choose new room option 3. End

Method Name	1. useMeetingFunction
Input	-
Output	-
Algorithm	1. Start 2. User choose meeting room function option 3. End

Method Name	1. selectName 2. writeMessage 3. displayMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User select the other user to start conservation

	3. User write the message 4. The message is displayed 5. End

Method Name	1. selectName 2. writeMessage 3. displayMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User select customer service 3. User write the message 4. The message is displayed 5. End

Method Name	1. displayMessage 2. replyMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User receives a message 3. User writes the reply message 4. The message is displayed 5. End

Entity Name	Supervisor
Method Name	1. chooseMeetingWay
Input	-
Output	-
Algorithm	1. Start 2. User select meeting way 3. End

Method Name	1. enterMeetRoomId 2. display
Input	-
Output	-
Algorithm	1. Start 2. User enter meeting room id 3. If valid meeting room id 3.1. view error message 4. End

Method Name	1. newRoom
Input	-
Output	-
Algorithm	1. Start

	2. User choose new room option 3. End

Method Name	1. useMeetingFunction
Input	-
Output	-
Algorithm	1. Start 2. User choose meeting room function option 3. End

Method Name	1. selectName 2. writeMessage 3. displayMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User select the other user to start conservation 3. User write the message 4. The message is displayed 5. End

Method Name	1. selectName 2. writeMessage

	3. displayMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User select customer service 3. User write the message 4. The message is displayed 5. End

Method Name	1. displayMessage 2. replyMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User receives a message 3. User writes the reply message 4. The message is displayed 5. End

Entity Name	Coordinator
Method Name	1. chooseMeetingWay
Input	-
Output	-

Algorithm	1. Start 2. User select meeting way 3. End

Method Name	1. enterMeetRoomId 2. display
Input	-
Output	-
Algorithm	1. Start 2. User enter meeting room id 3. If valid meeting room id 3.1. view error message 4. End

Method Name	1. newRoom
Input	-
Output	-
Algorithm	1. Start 2. User choose new room option 3. End

Method Name	1. useMeetingFunction
Input	-
Output	-

Algorithm	1. Start 2. User choose meeting room function option 3. End

Method Name	1. selectName 2. writeMessage 3. displayMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User select the other user to start conservation 3. User write the message 4. The message is displayed 5. End

Method Name	1. selectName 2. writeMessage 3. displayMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User select customer service 3. User write the message 4. The message is displayed 5. End

Method Name	1. displayMessage 2. replyMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User receives a message 3. User writes the reply message 4. The message is displayed 5. End

Entity Name	Evaluator
Method Name	1. chooseMeetingWay
Input	-
Output	-
Algorithm	1. Start 2. User select meeting way 3. End

Method Name	1. enterMeetRoomId 2. display
Input	-
Output	-

Algorithm	1. Start 2. User enter meeting room id 3. If valid meeting room id 3.1. view error message 4. End

Method Name	1. newRoom
Input	-
Output	-
Algorithm	1. Start 2. User choose new room option 3. End

Method Name	1. useMeetingFunction
Input	-
Output	-
Algorithm	1. Start 2. User choose meeting room function option 3. End

Method Name	1. selectName 2. writeMessage 3. displayMessage
Input	-

Output	User views the displayed message
Algorithm	1. Start 2. User select the other user to start conservation 3. User write the message 4. The message is displayed 5. End

Method Name	1. selectName 2. writeMessage 3. displayMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User select customer service 3. User write the message 4. The message is displayed 5. End

Method Name	1. displayMessage 2. replyMessage
Input	-
Output	User views the displayed message
Algorithm	1. Start 2. User receives a message 3. User writes the reply message

	4. The message is displayed 5. End

4.2.3.2. Sequence Diagram

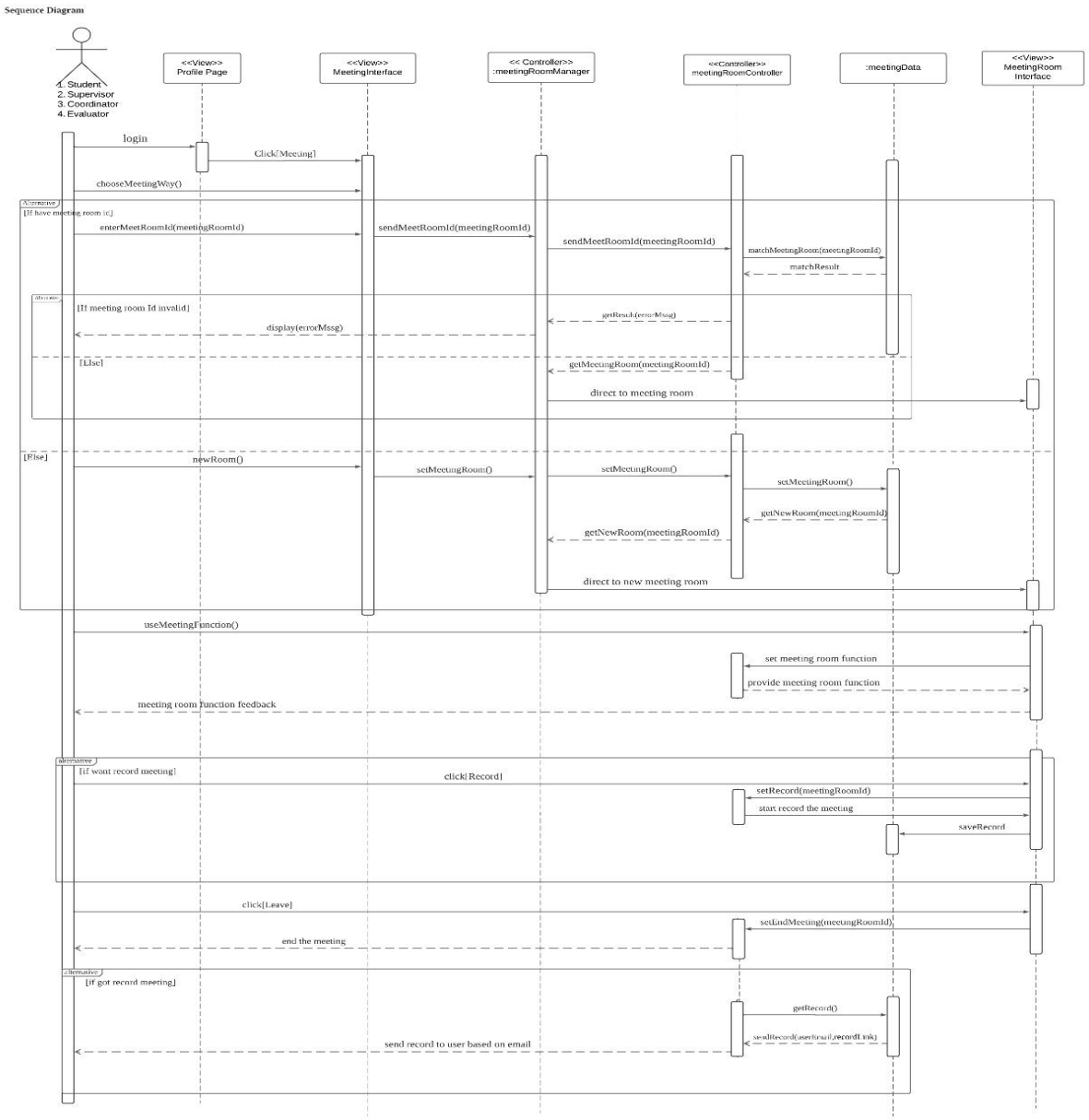

Figure 4.13 : Sequence Diagram for <Online Meeting Scenario>

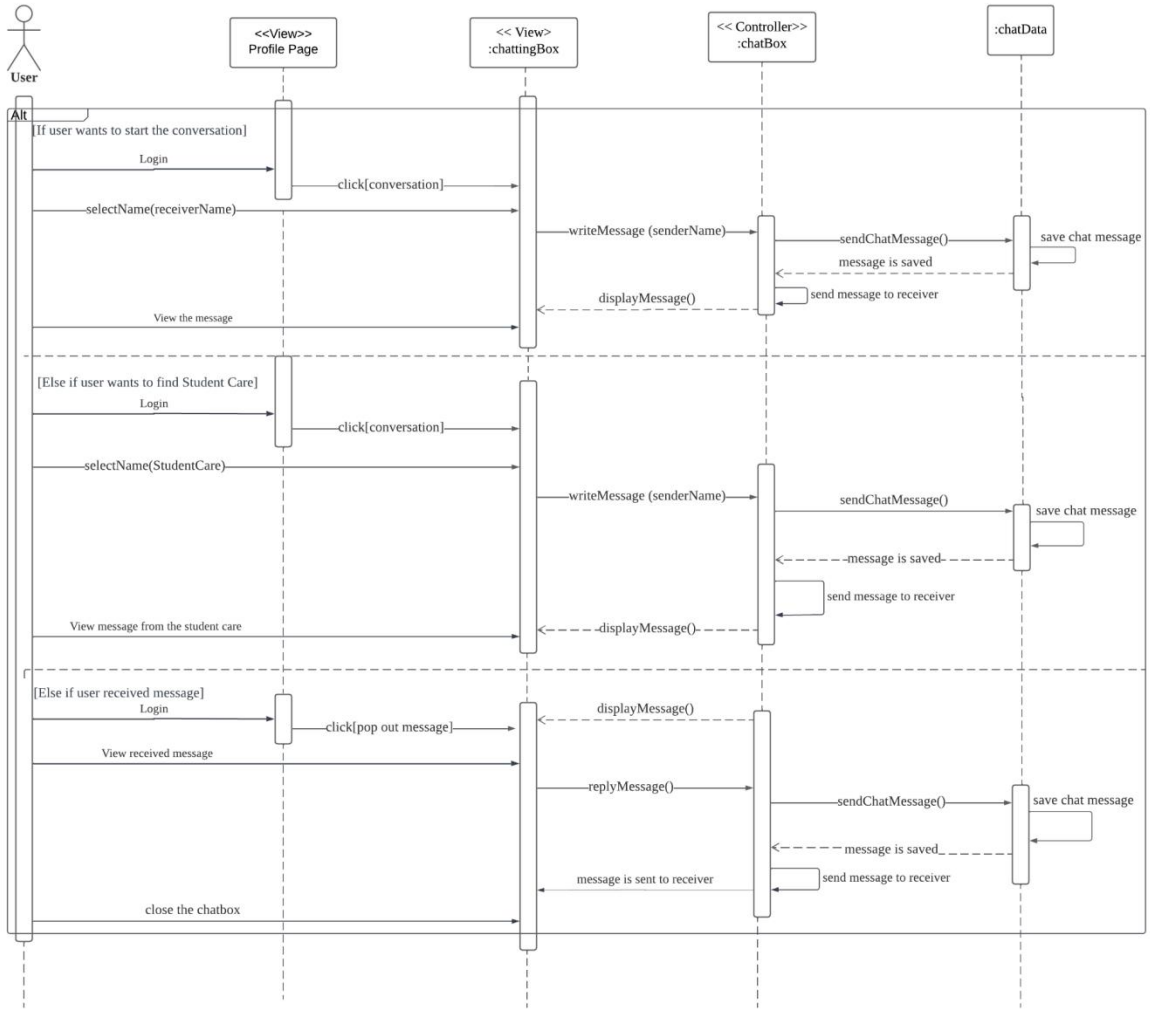

Figure 4.14 : Sequence Diagram for <Chatting Scenario>

Another Methods Used in System

Entity Name	1. Coordinator 2. Student 3. Supervisor 4. Evaluator

Method Name	1. login
Input	1. Entered username and password.
Output	1. Success to login the system. 2. Profile page will be displayed.
Algorithm	1. Start 2. Users click on the “Login” button. 3. Users enter their username. 4. Users enter their password. 5. Users click on the “Sign In” button. 6. System check for validity. 7. End.

Method Name	1. register
Input	1. Users enter personal information, username and password.
Output	1. Users will be registered successfully. 2. UserID will be returned to the users.
Algorithm	1. Start 2. Users click on the “Register” button. 3. Users enter their information and userID (matricNo). 4. System verify userID. 5. If userID is in UTM Database 5.1 System will add new users. 6. Else, users unable to register. 7. End.

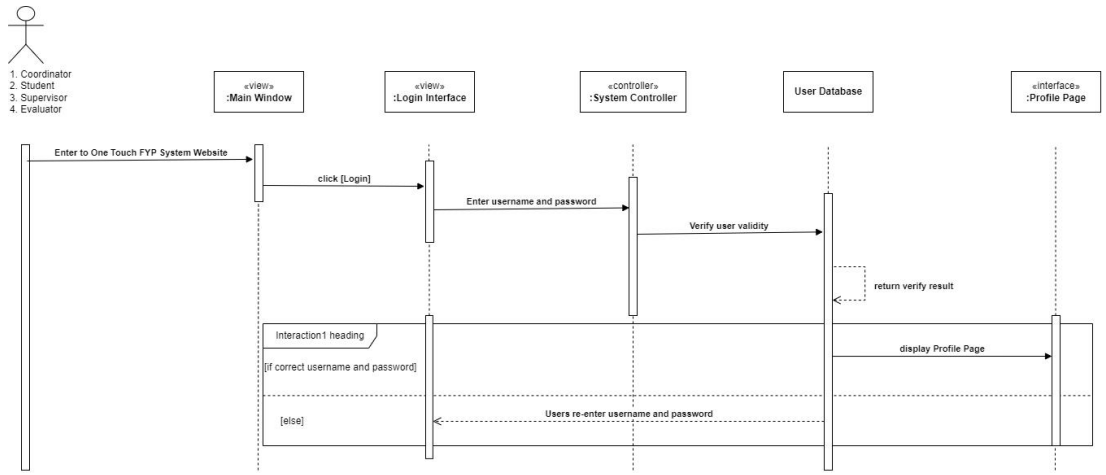

Figure 4.15 : Sequence Diagram for <Login Scenario>

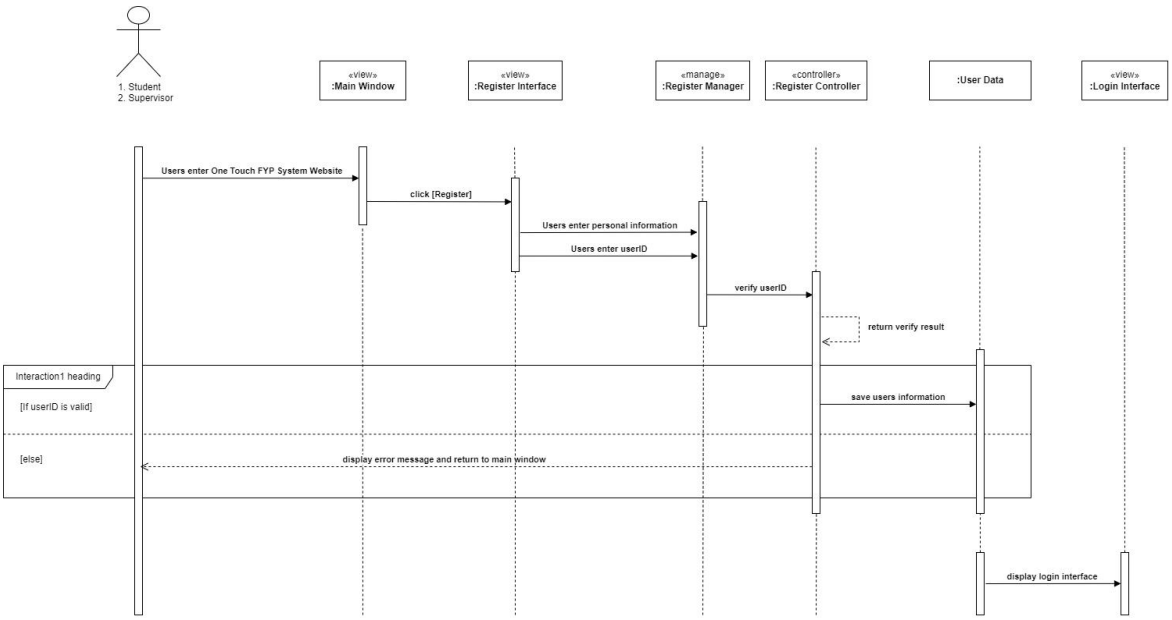

Figure 4.16 : Sequence Diagram for <Register Scenario>

5. Data Design

5.1. Data Description

The major data or systems entities are stored into a relational database named as One Touch FYP Database, processed and organized into 11 entities as listed in Table 5.1.

Table 5.1: Description of Entities in the Database

No.	Entity Name	Description
1.	Coordinator	● It will store the coordinator's information in the user database. ● Data will be used for validation in the login process. ● Data will be displayed in the profile page.
2.	Supervisor	● It will store the coordinator's information in the user database. ● Data will be used for validation in the login process. ● Data will be displayed in the profile page.
3.	Evaluator	● It will store the coordinator's information in the user database. ● Data will be used for validation in the login process. ● Data will be displayed in the profile page.
4.	Student	● It will store the coordinator's information in the user database. ● Data will be used for validation in the login process. ● Data will be displayed in the profile page.
5.	Evaluation	● It will acquire the student's report and student's evaluation form from the file database and form database. ● It will store the marks that had been given by the supervisor and evaluators in the evaluation database. ● It will calculate the total marks of the project.
6.	OnlineMeeting	● It will store the meeting data of the user into the meeting database. ● The data stored include the meeting recorded video. ● The users can get the URL link of the meeting recorded video when the video is saved in the meeting database.
7.	Chatbox	● It will store the chat data of the user into the chat database. ● The data can be used for users to review their chat history with other users.
8.	Forum	● It will store the content uploaded by the coordinator ● The user can view the posts in the forum

9.	Form	● It will store the templates of different forms ● It will store the forms filled by users
10.	Calendar	● It will store the student event and student activity set by coordinator ● The coordinator and student can set the date and time to notify student ● It will send notification to student by email
11.	Submission	● It will store the file submitted by student ● It will display the date and time of submission

5.2. Data Dictionary

5.2.1. Entity: <Student>

Attribute Name	Type	Description
studName	string	Name of student
studentIC	string	IC number of student
studentEmail	string	Registered email of student
userID	string	Unique identification used by student in the system
studMatricNo	string	Matric number of student
supervisorName	string	Name of the student's supervisor
subjectCode	string	Code of the subject taken by the student
username	string	Used by student to login.
password	string	Used by student to login.

5.2.2. Entity: <Supervisor>

Attribute Name	Type	Description
supervisorName	string	Name of supervisor
supervisorIC	string	IC number of supervisor
supervisorEmail	string	Email of supervisor
studName	string	Students who are under supervisor
areaOfInterest	string	Area of Interest of supervisor
userID	string	Unique identification used by supervisor in the system
username	string	Used by the supervisor to login.
password	string	Used by the supervisor to login.

5.2.3. Entity: <Evaluator>

Attribute Name	Type	Description
evaluateName	string	Name of evaluator
evaluatorIC	string	IC number of evaluator
evaluatorEmail	string	Email of evaluator
userID	string	Unique identification used by evaluator in the system
username	string	Used by evaluator to login.
password	string	Used by evaluator to login.

5.2.4. Entity: <Coordinator>

Attribute Name	Type	Description
coordinatorName	string	Name of coordinator
coordinatorIC	string	IC number of coordinator
coordinatorEmail	string	Email of coordinator
userID	string	Unique identification used by the coordinator in the system
username	string	Used by the coordinator to login.
password	string	Used by the coordinator to login.

5.2.5. Entity: <Forum>

Attribute Name	Type	Description
forumInfo	string	Content of the forum post
forumID	string	Unique ID for forum post
forumFileID	string	Unique ID of the file uploaded in the forum
errorMessage	string	Message that is displayed by the system when the post is uploaded/ submitted failed

5.2.6. Entity: <Notification>

Attribute Name	Type	Description
studentEvent	string	Event set by the coordinator
studentActivity	string	Activity
studentEmail	string	Registered email of student
remindTime	time	Time to notify student
remindDate	date	Date to notify student

5.2.7. Entity: <Form>

Attribute Name	Type	Description
formID	string	ID of the form
proposalID	string	ID of the proposal form
proposalInfo	string	Information to be filled in proposal form
logBookID	string	ID of the log book
logBookInfo	string	Information to be filled in the log book
evaluateFormID	string	ID of the evaluation form
evaluateFormInfo	string	Information to be filled in the evaluation form
changeFormID	string	ID of the change project title form
changeFormInfo	string	Information to be filled in the change project title form
reportFileID	string	ID of the report file

5.2.8. Entity: <Submission>

Attribute Name	Type	Description
fileID	string	Unique ID of the file submitted
subDateTime	datetime	Date and time of submission
errorMsg	string	Message that is displayed by the system when the file is uploaded/ submitted failed

5.2.9. Entity: <Evaluation>

Attribute Name	Type	Description
fileID	string	The project report submitted by students
subDateTime	Date	Project submission date and time
evaluationFormID	string	Evaluation Form filled by students
marks	double	Marks of students' project
evaDateTime	Date	Project evaluate date and time

5.2.10. Entity: <OnlineMeeting>

Attribute Name	Type	Description
meetingRoomId	string	A set of alphabet and number that represent the identification of the meeting room

recordLink	string	The URL link of the meeting recorded video
userEmail	string	The user's email
errorMssg	string	Error message when user enter invalid meeting room id

5.2.11. **Entity: <Chatbox>**

Attribute Name	**Type**	**Description**
studentCare	string	The name of student care service in the system
senderName	string	The sender's name in the chat box
receiverName	string	The receiver's name in the chat box
chatMsg	string	The chat message

6.Interface Design

6.1. Overview of Interface

When the user enters our system website, there is a main interface. The left side is a brief about our system and the right side is the interface for users to log in or register to the system. Before the user can use the system, the user needs to log in to the system. If the user does not have an account, the user needs to register an account by using their personal information such as name, IC number, email and userID at the register interface.

If the user has an account, they can login to the system using username and password. After login to the account, the system will direct the user to the user profile page. Here, there is a list of the navigation bar on the left side so the user can choose the function such as Forum, Calendar, Form, Evaluation, Meeting, and Conversation.

When the user clicks the Forum, the system will redirect the user to the forum interface. The forum interface will display a list of posts. Only the coordinator can create a new post, the top of the interface has a “Post” button for the coordinator to create a new post. Other users can only view the post here. When the coordinators wish to create the post, they can type the content and/ or upload the file. After inserting the content, the coordinator needs to click “Submit” to post it. After that, the post can be viewed by all users.

Next is the Calendar interface. Only the coordinator and student can set the notification for the student. When the user clicks to enter the calendar interface, the system will display a calendar. On the top right of the calendar is a gear-like setting button for the user to set the notification. Below the calendar are the notifications that are set. In the coordinator view, they can set the event by clicking on the “Setting” button on the top right of the interface. Then, the coordinator can select the student’s event. After that, they can set the date and time for the system for the notifications. The system will notify the students by their email according to the date and time set. In the student view, the student can click on the “Setting” button and choose the event set by the coordinator. The student can also set the time and date for their activity. The system will notify them via their email according to the date and time set.

In the form interface, there is a list of forms for students to choose. By clicking on the form, the system will display the form and the student can fill in that form. After filling the form, the student needs to click the “Submit” button to submit and save the form. The form filled out by students will also be sent to the coordinator, supervisor, and evaluator for further purposes.

There is an evaluation interface that is only accessible by the supervisor and evaluator. A list of students’ names and submitted projects can be viewed here. At the bottom right of each student list have a “View” button. By clicking the “View” button, the users can view the full report submitted by students. There is an “Evaluate” button at the bottom right of the students’ report. The supervisor and evaluator can click the “Evaluation” button to obtain the Evaluation Form and enter the marks. After that, they need to click the “Submit” button to save the marks.

The users can have an online meeting session in our system’s meeting room interface. At the meeting interface, the users can choose to enter the meeting room id or open a new room. If the room id entered exists, the system will redirect the user to the meeting room with matched id. If the meeting room id entered is invalid, the system will display an error message. If the user chooses to open a new room, the system will redirect the user to an empty room and assign a meeting room id so the user can share the id with other users. In the meeting room, there is a setting icon with three vertical dots. Users can click this icon to use some functions such as record meeting and share the meeting link. The recorded meeting will be sent to the user's email.

The last function is the conversation function. If the user wishes to start a message, the user can click on the “Conversation” button and a list of the name will be shown. The user can choose the person he or she wishes to chat with. Then, the chatbox will appear and the user can enter the chat. After finishing typing, the user needs to click the “Send” button to send the message. If the user wishes to find the Student Care, the user can click the “Conversation” button and the Student Care is on the top of the list. After that, the user can send the question they want to ask for help. If the user receives messages, the chatbox will pop out and the message is shown. The user can reply to the message by typing in the chat box and click the “Send” button.