

FYP Library

Mahnoor.	BCS191012
M. Kamran	BCS191037
M. Asim Amir	BCS191022



Spring-2022

Supervised By

Sir Hafiz Anas Billal

Department of Computer Science

Capital University of Science & Technology, Islamabad

Submission from the Final-Year

PROJECT REPORT

Version

V 1.0

NUMBERS OF
MEMBERS

03

TITLE

FYP Library

SUPERVISOR NAME

Sir Hafiz Anas Billal

MEMBER NAME	REG.NO.	EMAIL ADDRESS
Mahnoor	BCS191012	Mahnoorkhan0468@gmail.com
M. Kamran	BCS191037	Kamran11106@gmail.com
M. Asim Amir	BCS191022	Malixxasim111@gmail.com

MEMBERS' SIGNATURE

Supervisor's Signature

Table of Contents

Chapter 1.....	9
1.1 Introduction	9
1.2 Problem statement	9
1.3 Aim of the Project.....	9
1.4 FYP Library	10
2.2.1 Features	10
2.2.2 Super admin	10
2.2.3 Admin	10
2.2.4 Students	10
1.5 Business Scope	11
1.6 Existing example	11
1.7 Useful Tools and Technologies	11
1.8 Work Breakdown Structure	12
1.9 Project Timeline	13
Chapter 2.....	14
Requirement Specification and Analysis.....	14
2.1 Requirement Specification	14
2.1.1 Functional Requirements	14
2.1.2 Non-Functional Requirements.....	18
2.2 System Use case modeling	18
2.2.1 Use case of Super admin	19
2.2.2 Use case of Admin.....	20
2.2.3 Use case of Student	21
2.3 Use cases Descriptions	22
2.2.3 Add new admin.....	22
2.3.2 Search admin	25
2.3.3 Update admin.....	26
2.3.4 List of admin.....	29
2.3.5 Generate Accounts.....	30
2.3.7 Login	31

2.3.8	View the list of current semester Final Year Projects	Error! Bookmark not defined.
2.3.9	Admin can add projects	33
2.3.10	Search projects	34
2.3.11	Delete Projects.....	Error! Bookmark not defined.
2.3.12	Modify project.....	35
2.3.13	Student can view list of projects	36
2.3.14	Add own project information	Error! Bookmark not defined.
2.3.15	Upload project report	36
2.3.16	Logout	37
2.4	System sequence diagram.....	39
2.5	Domain Model.....	52
Chapter 3.....		53
System design		53
3.1	Software Architecture.....	53
3.2	Class Diagram.....	54
3.3	Sequence Diagram.....	55
3.4	Database schema.....	64
3.5	User Interface Design	65
Chapter 4.....		69
Software Development		69
4.1	Coding Standards	69
4.2	Development Environment	69
4.2.1	Visual Studio code	69
4.2.2	Apache MYSQL.....	69
4.3	Code Snippet.....	69
Chapter 5.....		73
Software Testing.....		73
5.1	Testing Methodology.....	73
5.1	Test Cases	73
5.3.1	Test Case 1	73
	Test case description	73

Expected result of the test case.....	73
Actual result of the test case.....	73
5.3.2 Test Case 2	74
Test case description	74
Expected result of the test case.....	74
Actual result of the test case.....	74
5.3.2 Test Case 3	75
Test case description	75
Expected result of the test case.....	75
Actual result of the test case.....	75

List of Figures

Figure 1 Work breakdown structure	12
Figure 2 Project timeline	13
Figure 3 Use case diagram of super admin.....	19
Figure 4 Use case of admin	20
Figure 5 Use case diagram of student.....	21
Figure 6 Super admin login	39
Figure 7 Add admin.....	39
Figure 8 update admin information	40
Figure 9 add project.....	40
Figure 10 add student	40
Figure 11 add supervisor	41
Figure 12 search project	41
Figure 13 Search admin.....	41
Figure 14 search students	42
Figure 15 list of admin.....	42
Figure 16 search supervisor.....	42
Figure 17 update project	43
Figure 18 update student.....	43
Figure 19 update supervisor	44
Figure 20 list of past fyp project.....	44
Figure 21 list of projects.....	44
Figure 22 list of students	45
Figure 23 list of supervisors	45
Figure 24 Super admin logout	45
Figure 25 Admin login	46
Figure 26 Current project	46
Figure 27 Past projects	46
Figure 28 Add project.....	47
Figure 29 Search projects	47
Figure 30 Delete projects.....	47
Figure 31 Update project information	48
Figure 32 Admin logout	48
Figure 33 Student login	49
Figure 34 Search project.....	49
Figure 35 Project list.....	50
Figure 36 Current project list.....	50
Figure 37 Upload report	50
Figure 38 upload project information	51
Figure 39 student logout	51

Figure 40 Domain model	52
Figure 41 Software Architecture	53
Figure 42 Class diagram	54
Figure 43 Super admin login	55
Figure 44 Add admin	55
Figure 45 Search admin	56
Figure 46 Update admin	56
Figure 47 list of admin	57
Figure 48 Super Admin logout	57
Figure 49 Admin login	58
Figure 50 Request list of project	58
Figure 51 Search project	59
Figure 52 Add project	59
Figure 53 update project	60
Figure 54 Admin logout	61
Figure 55 Student login	61
Figure 56 Request list of projects	62
Figure 57 Search project	62
Figure 59 Upload report to check plagiarism	63
Figure 60 Student logout	63
Figure 61 Database Schema	64
Figure 62 Sign in GUI	65
Figure 63 List of projects	66
Figure 64 CRUD operation	66
Figure 65 Add project	67
Figure 66 past projects	67
Figure 67 view report	68
Figure 68 upload report for plagiarism checking	68

List of Tables

Table 1 Functional requirements	14
Table 2 selected functional requirement.....	17
Table 3 Non-Functional requirements	18
Table 4 Add new admin.....	22
Table 5 Add supervisor.....	23
Table 6 Add student.....	24
Table 7 Search admin	25
Table 8 Search Supervisor.....	26
Table 9 Search Student	27
Table 10 update admin	28
Table 11 List of admin.....	29
Table 12 super admin and Admin can generate accounts	30
Table 13 Login to the system	31
Table 14 List of Supervisor	32
Table 15 Super Admin and Admin can add projects.....	33
Table 16 Search projects.....	34
Table 17 Modify project	35
Table 18 can view all projects	36
Table 19 Upload project report.....	37
Table 20 Logout to the system	38
Table 21 user successful login.....	74
Table 22 failed user login	74
Table 23 add admin	75
Table 24 add Admin	76

Chapter 1

Introduction

1.1 Project Introduction

Due to globalization in 21st century, the growth in interdependence of the economy, culture and population have resulted in creativity and innovation in goods/services and technology. In this regard, the current market trends are changing day by day due to increase in demands of the people. In order to enter into the market and to be more competitive, the business idea is required to be unique and innovative.

When newly graduate students apply for jobs and go to interview the first thing the interviewer see is that person's ability, skills and his/her FYP. Students' main focus is to bring a new positive change, a new innovation in the environment, which can help the country and business industry to grow. Our focus is to help these students to see what changes they can bring.

1.2 Problem statement

In our university when are near graduation (7th semester) they have to do a Final Year Project which help them to learn new skills and technologies. This Final Project also tells the ability of a student and further help them in their professional life. But the problem is that it becomes difficult for the students to select the topic for their FYP. As they cannot select the projects which had been done before. They can add new features to the previous project or bring new innovation.

Students have to visit the libraries to check the previous Projects. As there are lesser number of projects present there, so it is no help to students. To overcome this problem our application will help the students to get a glimpse at the projects that had been done before, at what year and who was their supervisor.

1.3 Aim of the Project

The objective of the project is to develop such customized system that will be user friendly and will meet the needs of the customers. The target audience is both males and females. After development, the app will be used in pilot testing in order to check its reliability and validity.

1.4 FYP Library

FYP Library is a web-based application which will help the undergraduates to see the FYPs' done in the past. Which will help them gain perspective and see what changes they can bring.

1.4.1 Features

- Account Creation.
- Login.
- FYP Repository.
- Searching of projects.
- Plagiarism checking on projects.

1.4.2 Super admin

- Will generate accounts of admin, student and supervisor.
- Will be able to search, add, and modify the accounts.
- Will be able to search, add, and modify the projects.
- Can view the list of previous projects.
- Can view the list of current semester projects.
- Will be able to see his profile.

1.4.3 Admin

- Will generate accounts.
- Will be able to search, add, and modify the Projects.
- Can view the list of previous projects.
- Can view the list of current semester projects.

1.4.4 Students

- Sign in using that registration number and password.
- Will be able to view the past Final Year Projects, their description, the year they were done and that specific project supervisor.
- Can view the list of current semester projects.
- Can search for specific project.
- Can upload report.

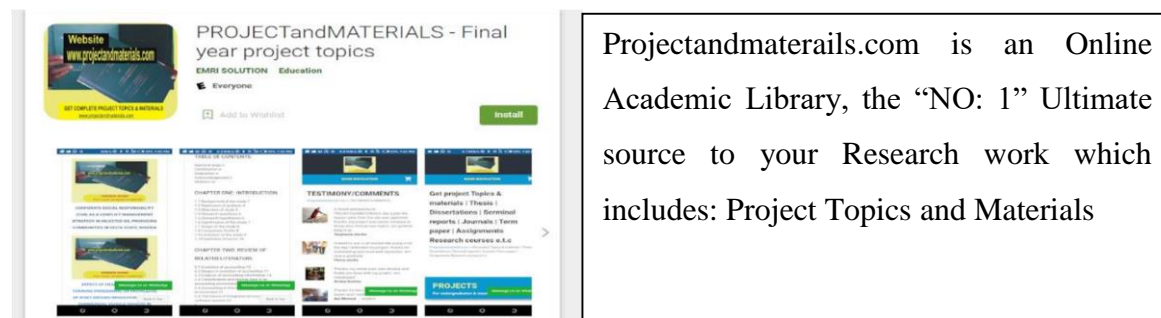
- Can generate plagiarism report.

1.5 Business Scope

This application will help the students of CUST (Capital University of Science and Technology) to choose best projects without thinking that this project was done in the past. The business scope is very clear because such system can enhance the reputation of the university, by providing their students a system which will help them to bring the best innovative ideas. They will be able to see the description which will help them to see the missing points in the previous project.

The scope of this application for our FYP is just CUST University. We can make this application international in which the students can enter their projects and help other students to gain knowledge.

1.6 Existing example



1.7 Useful Tools and Technologies

Following is a list of all possible technologies that will be required during our project:

- **DJango:** For web application development including HTML5, CSS, Bootstrap and Java script.
- **SQL database:** For database.
- **Python/PyCharm**

1.8 Work Breakdown Structure

The figure below shows the work breakdown structure of the complete project. It involves the activities which are required to complete the project.

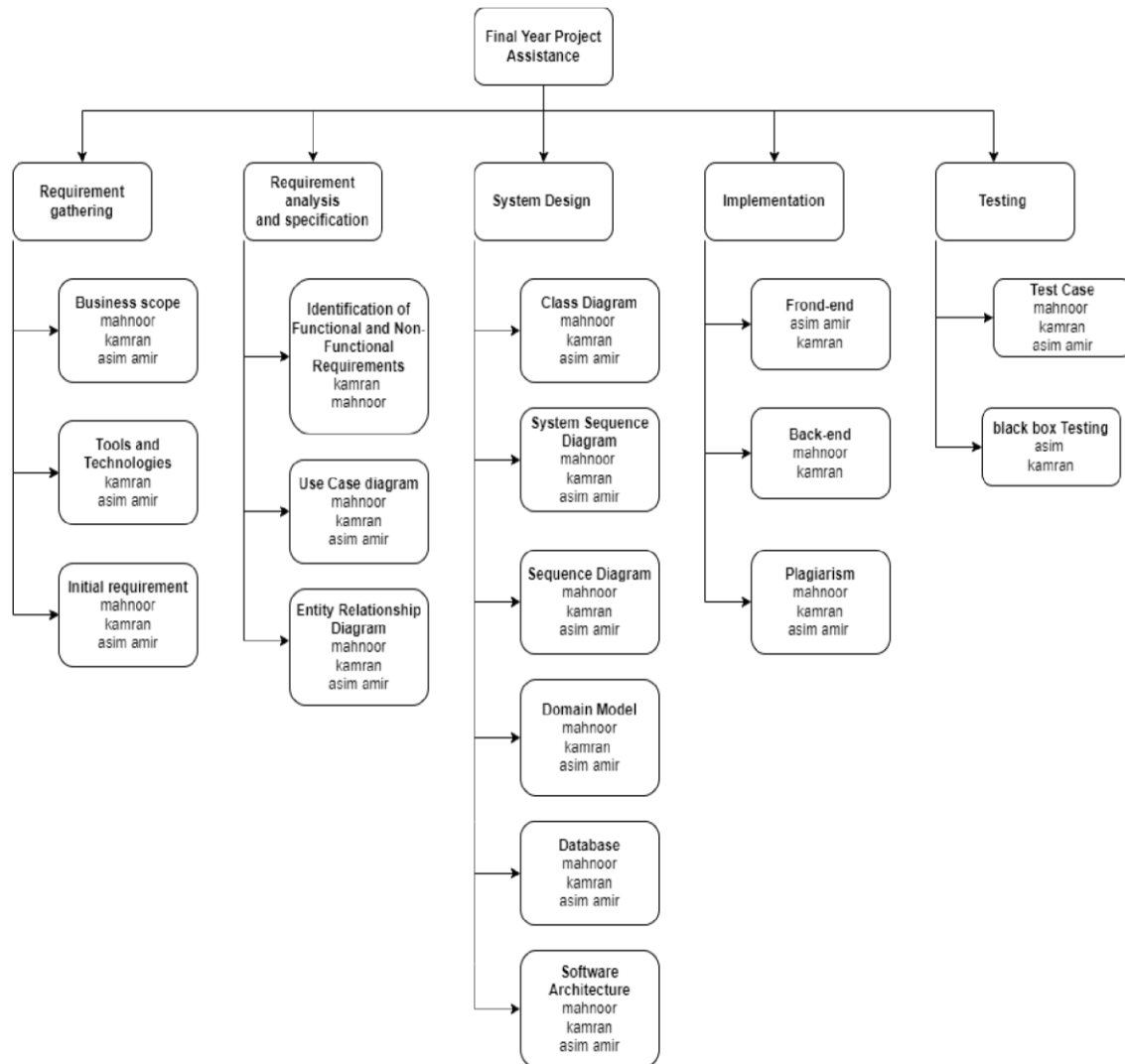


Figure 1 Work breakdown structure

1.9 Project Timeline

The figure below shows the project timeline which includes amount of time utilized on each task.

ID	Task Name	Start	Finish	Duration	Q2 22			Q3 22			Q4 22			Q1 23	
					Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb
1	Requirements Gathering	23/03/2022	15/04/2022	18d											
2	Requirement Analysis	15/04/2022	15/06/2022	44d											
3	Design and architecture	12/05/2022	19/08/2022	72d											
4	implementation	15/07/2022	01/02/2023	144d											
5	Testing	01/08/2022	15/02/2023	143d											

Figure 2 Project timeline

Chapter 2

Requirement Specification and Analysis

The emphasis of this chapter is on getting an idea of what the requirements are for the intended software. Students who are doing a research related project would provide literature surveys for their problems. They are expected to understand the relevant papers and provide a summary of the existing work presented in each research paper. Such students should consult their project supervisor for the detailed instructions related to this chapter.

2.1 Requirement Specification

Requirements specification involves frequent communication with system users to determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by the various users or groups of users and documentation of all aspects of the project development process from start to finish. Requirements are a description of how a system should behave or a description of system properties or attributes. It can alternatively be a statement of 'what' an application is expected to do.

2.1.1 Functional Requirements

Functional requirement defines a system or its component. It describes the functions a software must perform. A function is nothing but inputs, its behavior, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform.

Functional requirements included in our system is shown in Table.

Table 1 Functional requirements

Sr.	Requirements	Type	Status
1.	Super admin can login.	Core	Completed
2.	Super admin can logout.	Core	Completed
3.	Super admin can generate accounts of admin.	Core	Completed

4.	Super admin can generate accounts of Supervisor.	Core	Completed
5.	Super admin can generate accounts of student.	Core	Completed
6.	Super admin can update information of admin.	Core	Completed
7.	Super admin can update information of supervisors.	Core	Completed
8.	Super admin can update information of student.	Core	Completed
9.	Super admin can see its own profile.	Core	Completed
10.	Super admin can see list of admins.	Core	Completed
11.	Super admin can see list of supervisors.	Core	Completed
12.	Super admin can see list of students.	Core	Completed
13.	Super admin can search admin.	Core	Completed
14.	Super admin can search supervisor.	Core	Completed
15.	Super admin can search students.	Core	Completed
16.	Super admin can search projects.	Core	Pending
17.	Super admin can add projects.	Core	Pending
18.	Super admin can update projects.	Core	Pending
19.	Super admin can see list of projects.	Core	Pending
20.	Super admin can view the list of past Final Year projects.	Core	Pending
21.	Admin can login.	Core	Completed
22.	Admin can view the list of past Final Year projects.	Core	Pending
23.	Admin can add projects	Core	Pending

24.	Admin can update the information about the project.	Core	Pending
25.	Admin can update the information about the students.	Core	Completed
26.	Admin can update the information about supervisors.	Core	Completed
27.	Admin can search project.	Core	pending
28.	Admin can search student.	Core	Completed
29.	Admin can search supervisor.	Core	Completed
30.	Admin can generate student accounts.	Core	Completed
31.	Admin can generate supervisor accounts.	Core	Completed
32.	Admin can check plagiarism	Core	Pending
33.	Admin can logout.	Core	Completed
34.	Student can login.	Core	Completed
35.	Student can view the list of past Final Year projects.	Core	Pending
36.	Student can search project by its title	Core	Pending
37.	Student can add Project.	Core	Pending
38.	Student can upload report for plagiarism.	Core	Pending
39.	Student can logout.	Core	Completed
40.	Supervisor can Login.	Optional	Pending
41.	Supervisor can logout.	Optional	Pending
42.	Supervisor can add ideas of projects.	Optional	Pending

2.1.2 Selected Functional Requirements

Table 2 selected functional requirement

Sr.	Requirements	Type	Status
1.	Super admin can login.	Core	Completed
2.	Super admin can logout.	Core	Completed
3.	Super admin can generate accounts of admin.	Core	Completed
4.	Super admin can generate accounts of Supervisor.	Core	Completed
5.	Super admin can generate accounts of student.	Core	Completed
6.	Super admin can update information of admin.	Core	Completed
7.	Super admin can update information of supervisors.	Core	Completed
8.	Super admin can update information of student.	Core	Completed
9.	Super admin can see its own profile.	Core	Completed
10.	Super admin can see list of admins.	Core	Completed
11.	Super admin can see list of supervisors.	Core	Completed
12.	Super admin can see list of students.	Core	Completed
13.	Super admin can search admin.	Core	Completed
14.	Super admin can search supervisor.	Core	Completed
15.	Super admin can search students.	Core	Completed
16.	Admin can login.	Core	Completed
17.	Admin can update the information about the students and supervisors.	Core	Completed

18.	Admin can generate student accounts.	Core	Completed
19.	Admin can logout.	Core	Completed
20.	Student can login.	Core	Completed
21.	Student can logout.	Core	Completed

2.1.3 Non-Functional Requirements

A non-functional requirement defines the quality attribute of a software system. They represent a set of standards used to judge the specific operation of a system.

List of non-functional requirements of the system as shown:

Table 3 Non-Functional requirements

Sr.	Non-Functional Requirements	Category
1.	System shall have online connectivity 24/7.	Availability
2.	Response within 30 seconds.	Response time
3.	End user can learn this web app within 10 minutes.	Usability
4.	Cannot login without proper authentication.	Security

2.2 System Use case modeling

Use Case Diagram of the overall system in which graphic depiction of the interactions among the elements of a system and the relationships between and among the actors. We show use cases with respect to particular actors related with actions.

2.2.1 Use case of Super admin

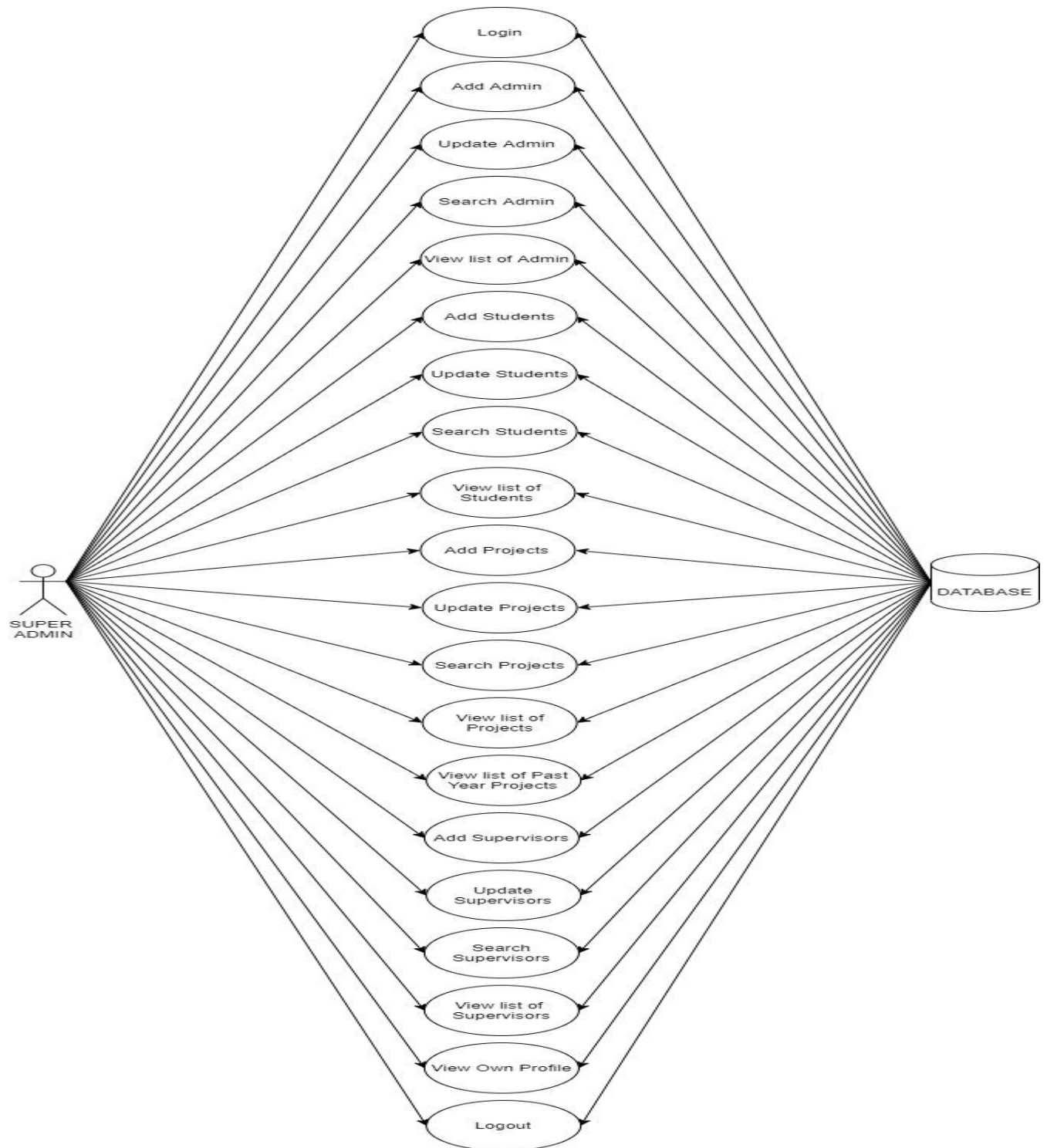


Figure 3 Use case diagram of super admin

2.2.2 Use case of Admin

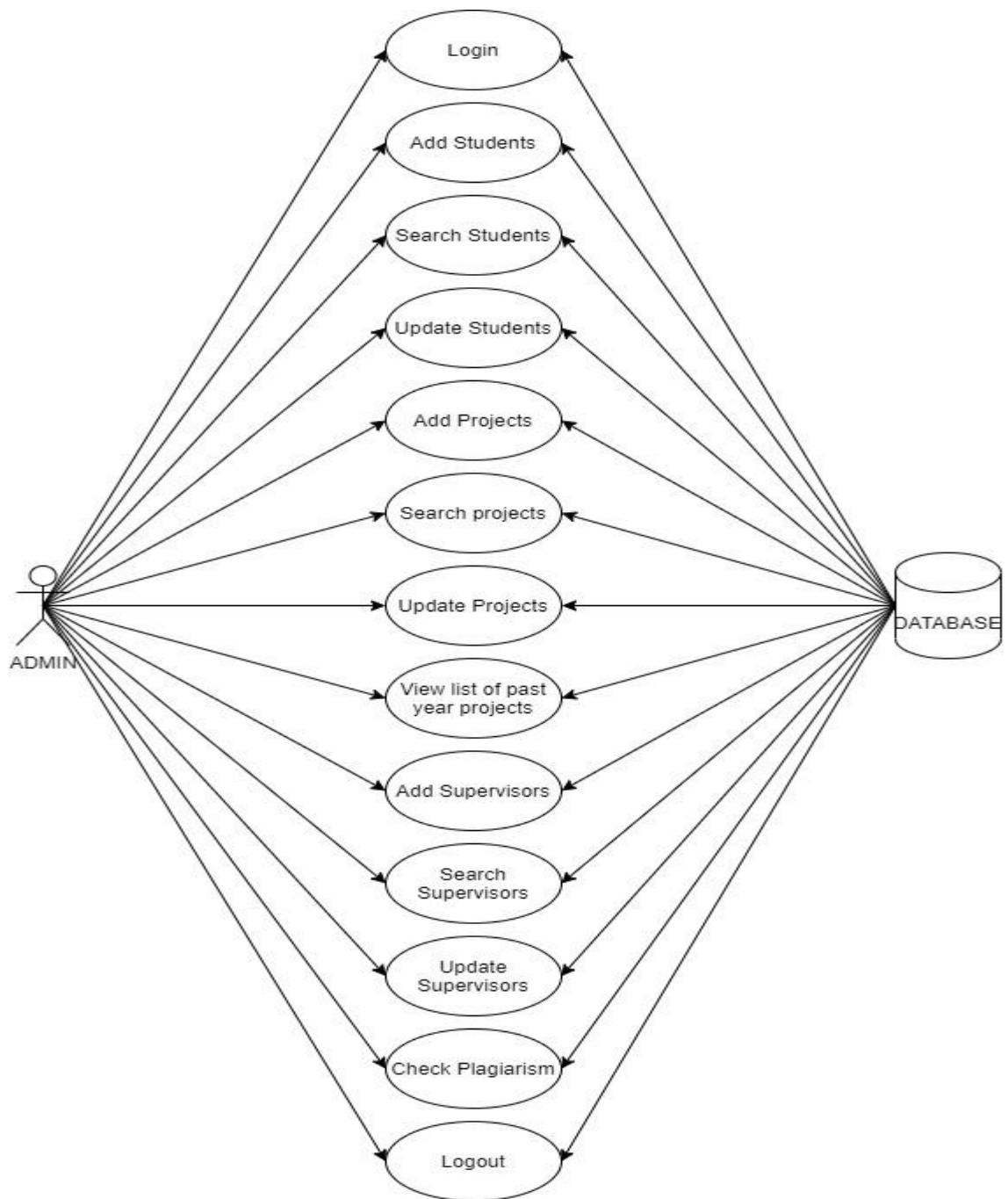


Figure 4 Use case of admin

2.2.3 Use case of Student

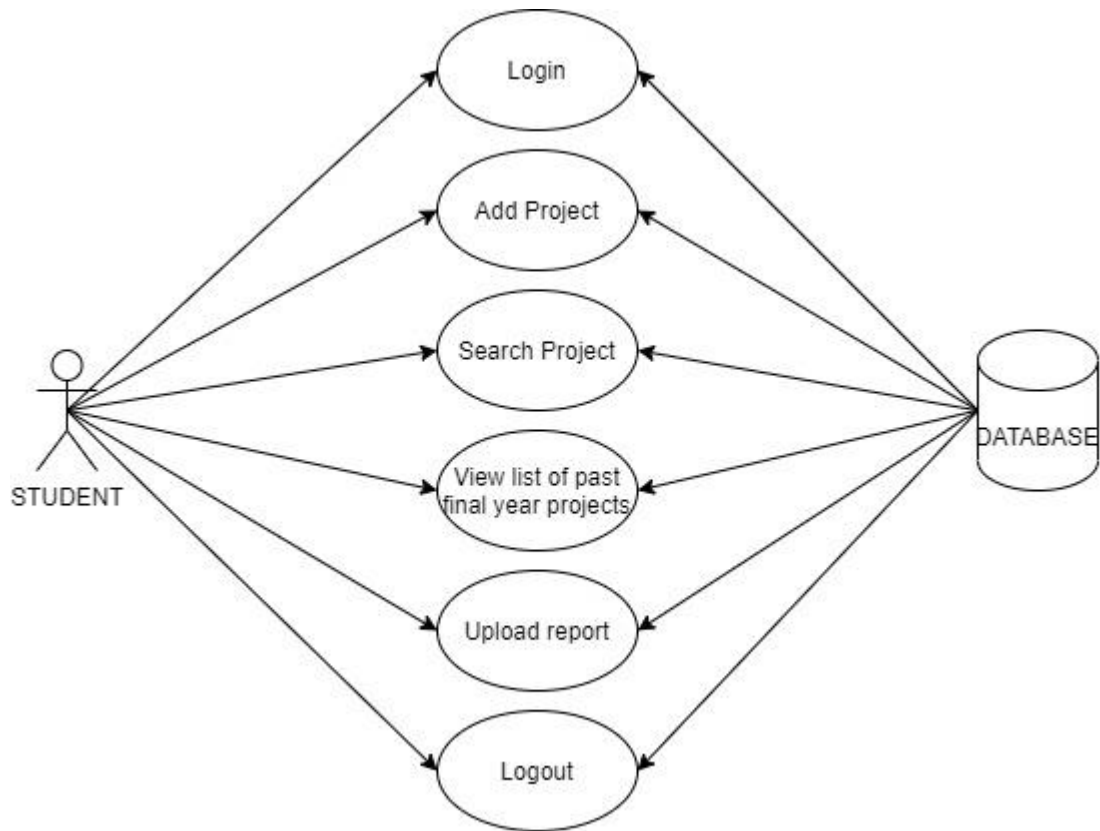


Figure 5 Use case diagram of student

2.3 Use cases Descriptions

A use case is a written description of how users will perform tasks on your application. It outlines, from a user's point of view, a system's behavior as it responds to a request. Each use case is represented as a sequence of simple steps, beginning with a user's goal and ending when that goal is fulfilled.

2.3.1 Add new admin

Here is the Use Case in which super admin can login valid credentials and after successful login the super admin can perform CRUD operations on admin. Super admin will see an open portal homepage where there will be a button of Add admin and after pressing that button the system will add admin in database as shown:

Table 4 Add new admin

Use Case ID:	01		
Use Case Name:	Super admin can add new admin.		
Created By:	Mahnoor	Last Updated By:	N/A
Date Created:	2/4/2022	Last Revision Date:	4/4/2022
Actors:	Super Admin.		
Description:	Super Admin can add a new admin in the system.		
Trigger:	Click on Add admin button.		
Preconditions:	Super Admin must be login into the system.		
Post conditions:	Admin added successfully.		
Normal Flow:	Super admin 1-Super admin must enter login credentials. 3-Successful Login. 5-Click on Add button. 7-Super admin will add admin	System 2-Validate login credentials. 4-Open Portal Homepage. 6-System add admin in database. 9-Information will be stored in the Database.	

	information. 8-Enter submit button.	
Alternative Flows:	Admin already exists.	
Exceptions:	1. Internet connection is not available. 2. Database is not responding.	

2.3.2 Add Supervisor

Here is the Use Case in which super admin and admin can login valid credentials and after successful login the super admin and admin can perform operations on supervisor. Actors will see an open portal homepage where there will be a button of Add supervisor and after pressing that button the system will add supervisor in database as shown:

Table 5 Add supervisor

Use Case ID:	02		
Use Case Name:	Add Supervisor		
Created By:	Mahnoor	Last Updated By:	N/A
Date Created:	2/4/2022	Last Revision Date:	4/4/2022
Actors:	Super Admin, Admin.		
Description:	Super Admin and Admin can add a new supervisor in the system.		
Trigger:	Click on Add button.		
Preconditions:	Actors must be login into the system.		
Post conditions:	Supervisor added successfully.		
Normal Flow:	Actors 1-Super admin must enter login credentials. 3-Successful Login. 5-Click on Add button. 7-Super admin will add admin	System 2-Validate login credentials. 4-Open Portal Homepage. 6-System add admin in database. 9-Information will be stored in the Database.	

	information. 8-Enter submit button.	
Alternative Flows:	N/A.	
Exceptions:	1. Internet connection is not available. 2. Database is not responding. 3. Supervisor already exists.	

2.3.3 Add Student

Here is the Use Case in which super admin and admin can login valid credentials and after successful login the super admin and admin can perform operations on student. Actors will see an open portal homepage where there will be a button of Add student and after pressing that button the system will add student in database as shown:

Table 6 Add student

Use Case ID:	03		
Use Case Name:	Add Student		
Created By:	Mahnoor	Last Updated By:	N/A
Date Created:	2/4/2022	Last Revision Date:	4/4/2022
Actors:	Super Admin, Admin.		
Description:	Super Admin and Admin can add a new student in the system.		
Trigger:	Click on Add button.		
Preconditions:	Actors must be login into the system.		
Post conditions:	Supervisor added successfully.		
Normal Flow:	Actors 1-Super admin must enter login credentials. 3-Successful Login. 5-Click on Add button. 7-Super admin will add admin	System 2-Validate login credentials. 4-Open Portal Homepage. 6-System add admin in database. 9-Information will be stored in	

	information. 8-Enter submit button.	the Database.
Alternative Flows:	N/A.	
Exceptions:	1. Internet connection is not available. 2. Database is not responding. 3. Supervisor already exists.	

2.3.4 Search admin

Here is the Use Case in which super admin can login valid credentials and after successful login then they will see an open portal homepage where there will be a button of Search through which super admin can search admin and system will show the information of admin as shown below:

Table 7 Search admin

Use Case ID:	04		
Use Case Name:	Search admin		
Created By:	Mahnoor	Last Updated By:	N/A
Date Created:	2/4/2022	Last Revision Date:	4/4/2022
Actors:	Super admin		
Description:	Super admin can search for admin by entering a admin id/name.		
Trigger:	Click on the search button.		
Preconditions:	1. User must be logged in to the system. 2. Must enter admin id/name		
Post conditions:	User will be able to view the admin details.		
Normal Flow:	Super admin 1-Super admin must enter the user name and password. 3-Successful Login. 5-Click on Search button.	System 2-Validate login credentials. 4-Open Portal Homepage.	

	6-Search by name/ id and press enter.	7-System will show the admin.
Alternative Flows:	See list of admins.	
Exceptions:	1. Internet connection is not available. 2. Database is not responding. 3. Admin not found	

2.3.5 Search Supervisor

Here is the Use Case in which super admin and admin can login valid credentials and after successful login then they will see an open portal homepage where there will be a button of Search through which super admin and admin can search supervisor and system will show the information of supervisor as shown below:

Table 8 Search Supervisor

Use Case ID:	05		
Use Case Name:	Search Supervisor		
Created By:	Mahnoor	Last Updated By:	N/A
Date Created:	2/4/2022	Last Revision Date:	4/4/2022
Actors:	Super admin, Admin		
Description:	Actors can search for supervisor by entering a supervisor id/name.		
Trigger:	Click on the search button.		
Preconditions:	1. User must be logged in to the system. 2. Must enter admin id/name		
Post conditions:	User will be able to view the admin details.		
Normal Flow:	Actors 1-Actors must enter the user name and password. 3-Successful Login. 5-Click on Search button.	System 2-Validate login credentials. 4-Open Portal Homepage.	

	6-Search by name/ id and press enter.	7-System will show the supervisor.
Alternative Flows:	See list of supervisor.	
Exceptions:	1. Internet connection is not available. 2. Database is not responding. 3. Supervisor not found	

2.3.6 Search Student

Here is the Use Case in which super admin and admin can login valid credentials and after successful login then they will see an open portal homepage where there will be a button of Search through which super admin and admin can search student and system will show the information of student as shown below:

Table 9 Search Student

Use Case ID:	06		
Use Case Name:	Search Student		
Created By:	Kamran	Last Updated By:	N/A
Date Created:	2/4/2022	Last Revision Date:	4/4/2022
Actors:	Super admin, Admin		
Description:	Actors can search for students by entering a student id/name.		
Trigger:	Click on the search button.		
Preconditions:	1. User must be logged in to the system. 2. Must enter student id/name		
Post conditions:	User will be able to view the student details.		
Normal Flow:	Actors 1-Actors must enter the user name and password. 3-Successful Login. 5-Click on Search button.	System 2-Validate login credentials. 4-Open Portal Homepage.	

	6-Search by name/ id and press enter.	7-System will show the student.
Alternative Flows:	See list of student.	
Exceptions:	4. Internet connection is not available. 5. Database is not responding. 6. Student not found	

2.3.7 Update admin

Here is the Use Case in which super admin will login through valid credentials and after successful login super admin will see an open portal homepage where there will be a button of update through which super admin can update the information of an admin, after updating the system will show the updated information of the admin.

Table 10 update admin

Use Case ID:	07		
Use Case Name:	Update admin		
Created By:	M. Asim Amir	Last Updated By:	N/A
Date Created:	3/4/2022	Last Revision Date:	4/4/2022
Actors:	Super Admin		
Description:	Super Admin will be able to update admin information.		
Trigger:	Click on update button.		
Preconditions:	1. User must be logged in. 2. Admin must exist.		
Post conditions:	Admin information must be updated.		
Normal Flow:	Super admin 1-Super admin must enter user name and password. 3-Successful Login.	System 2-Validate login credentials. 4-Open Portal Homepage.	

	5-Click on update button. 7-Enter updated information. 8-Press enter.	6-Show update form. 9-System will store updated Information in the database.
Alternative Flows:	N/A.	
Exceptions:	1. Internet connection is not available. 2. Database is not responding. 3. Admin does not exists.	

2.3.8 List of admin

Here is the Use Case in which super admin can login credentials and after successful login then super admin will see an open portal homepage. After that they will see the list of admins button on the open portal homepage through which they will see the list of admins as shown:

Table 11 List of admin

Use Case ID:	08		
Use Case Name:	The list of admin.		
Created By:	Mahnoor	Last Updated By:	N/A
Date Created:	2/4/2022	Last Revision Date:	4/4/2022
Actors:	Super Admin		
Description:	Super admin will be able to view the list of admin through his account.		
Trigger:	Click on view list of admin button.		
Preconditions:	Super admin must login to the system.		
Post conditions:	Super admin will be able to see the list of admins.		
Normal Flow:	Super admin 1-Super admin must enter login credentials.	System 2-Validate login credentials. 4-Open Portal Homepage.	

	3-Successful Login. 5-Click on list of admin button.	6-System will show list of admin.
Alternative Flows:	1. System will display empty fields.	
Exceptions:	1. Internet connection is not available. 2. Database is not responding. 3. No admin exists.	

2.3.9 Generate Accounts

Here is the Use Case of generate accounts in which user must have valid login credentials through which they can login and after successfully login, admin will see an open portal homepage where there will be a button of generate accounts. Admin will press that button of generate accounts and data will be fetch from server and accounts will be generated as shown below.

Table 12 super admin and Admin can generate accounts

Use Case ID:	09		
Use Case Name:	Super admin and Admin can generate student accounts.		
Created By:	Mahnoor	Last Updated By:	N/A
Date Created:	1/4/2022	Last Revision Date:	4/4/2022
Actors:	Super admin and Admin		
Description:	Super Admin can generate students' accounts which are not registered yet.		
Trigger:	Click on the generate accounts button.		
Preconditions:	Super admin and Admin must login to the system.		
Post conditions:	Accounts generated Successfully.		
Normal Flow:	Super Admin 1. User must login credentials. 3. Successful Login.	System 2. Validate login credentials. 4. Open Portal Homepage.	

	5. Admin press on generate account button.	6. Fetch data from server and generate accounts.
Alternative Flows:	Error message that Account already exists.	
Exceptions:	1. Internet connection is not available. 2. Database is not responding.	

2.3.10 Login

Here is the Use Case in which admin and student can login credentials and after successful login then they will see an open portal homepage as shown below:

Table 13 Login to the system

Use Case ID:	10		
Use Case Name:	Login to the system		
Created By:	Mahnoor	Last Updated By:	N/A
Date Created:	1/4/2022	Last Revision Date:	4/4/2022
Actors:	Super admin, Admin, Student		
Description:	Users can login to the system.		
Trigger:	Click on the login button.		
Preconditions:	User must have an account.		
Post conditions:	User logged in successfully.		
Normal Flow:	Actors 1. User must login credentials. 3. Successful Login.	System 2. Validate login credentials. 4. Open Portal Homepage.	
Alternative Flows:	User provide invalid credentials system will ask user to enter valid fields.		
Exceptions:	1. Internet connection is not available. 2. Database is not responding.		

2.3.11 List of Supervisor

Here is the Use Case in which super admin and admin can login credentials and after successful login they will see an open portal homepage. After that they will see the list of supervisor button on the open portal homepage through which they will see the list of supervisor as shown:

Table 14 List of Supervisor

Use Case ID:	11		
Use Case Name:	List of supervisor.		
Created By:	M. Asim Amir	Last Updated By:	N/A
Date Created:	2/4/2022	Last Revision Date:	4/4/2022
Actors:	Super admin, admin		
Description:	Users will be able to view the list of supervisors.		
Trigger:	Click on view list of supervisor button.		
Preconditions:	User must login to the system.		
Post conditions:	User must view the list of supervisor.		
Normal Flow:	Actors 1. User must login credentials. 3. Successful Login. 5. Click on current semester project button.	System 2. Validate login credentials. 4. Open Portal Homepage. 6. System will show list of current semester Final Year Projects.	
Alternative Flows:	N/A		
Exceptions:	1. Internet connection is not available. 2. Database is not responding. 3. System will display empty fields.		

2.3.12 Add projects

Here is the Use Case in which admin can login valid credentials and after successful login then admin will see an open portal homepage where there will be a button of Add project and after pressing that button the system will add project in database as shown:

Table 15 Super Admin and Admin can add projects

Use Case ID:	12		
Use Case Name:	Super Admin, Admin and can add new projects.		
Created By:	M. Asim Amir	Last Updated By:	N/A
Date Created:	2/4/2022	Last Revision Date:	4/4/2022
Actors:	Super Admin, Admin and students.		
Description:	Super Admin, Admin and students can add projects for students whose records are not in the database.		
Trigger:	Click on Add project button.		
Preconditions:	Super Admin and Admin must login to the system.		
Post conditions:	Project added successfully.		
Normal Flow:	Actors 1. User must login credentials. 3. Successful Login. 5. Click on Add project.	System 2. Validate login credentials. 4. Open Portal Homepage. 6. System add project in database.	
Alternative Flows:	Project already exists.		
Exceptions:	1. Internet connection is not available. 2. Database is not responding.		

2.3.13 Search projects

Here is the Use Case in which admin and student can login valid credentials and after successful login then they will see an open portal homepage where there will be a button of Search through which they can search projects by title and system will show the projects as shown below:

Table 16 Search projects

Use Case ID:	13		
Use Case Name:	Search project		
Created By:	M. Kamran	Last Updated By:	N/A
Date Created:	2/4/2022	Last Revision Date:	4/4/2022
Actors:	Super Admin, Admin, student		
Description:	User can search for a project by entering a project title.		
Trigger:	Click on the search button.		
Preconditions:	1. User must be logged in to the system. 2. Must enter a project title.		
Post conditions:	User will be able to view the project details.		
Normal Flow:	Actors 1. User must login credentials. 3. Successful Login. 5. Click on Search button.	System 2. Validate login credentials. 4. Open Portal Homepage. 6. System will show the project.	
Alternative Flows:	NA		
Exceptions:	1. Internet connection is not available. 2. Database is not responding.		

2.3.14 Modify project

Here is the Use Case in which admin will login through valid credentials and after successful login then admin will see an open portal homepage where there will be a button of update through which admin can update projects and then the system will display the updated information of projects as shown:

Table 17 Modify project

Use Case ID:	14		
Use Case Name:	Modify project		
Created By:	Mahnoor	Last Updated By:	N/A
Date Created:	3/4/2022	Last Revision Date:	4/4/2022
Actors:	Super Admin, Admin		
Description:	Both Admins will be able to update project information.		
Trigger:	Click on update button.		
Preconditions:	1. User must be logged in. 2. Project must exist.		
Post conditions:	Must updated information.		
Normal Flow:	Actors 1. User must login credentials. 3. Successful Login. 5. Click on update button.	System 2. Validate login credentials. 4. Open Portal Homepage. 6. System will show the list project. 7. System will display updated information.	
Alternative Flows:	Project does not exist.		
Exceptions:	1. Internet connection is not available. 2. Database is not responding.		

2.3.15 View list of projects

Here is the Use Case in which admin and student can login valid credentials and after successful login then they will see an open portal homepage where there will be a button of all projects and after pressing all projects button, system will show the list of all projects as shown:

Table 18 can view all projects

Use Case ID:	15		
Use Case Name:	User can view the list of Past Final Year projects.		
Created By:	M. Asim Amir	Last Updated By:	N/A
Date Created:	3/4/2022	Last Revision Date:	4/4/2022
Actors:	Super admin, Admin, Student.		
Description:	User will be able to view the list of past Final Year projects.		
Trigger:	Click on all projects button.		
Preconditions:	Actors must be logged in.		
Post conditions:	Must view details of projects.		
Normal Flow:	Actors 1. User must login credentials. 3. Successful Login. 5. Click on all projects button.	System 2. Validate login credentials. 4. Open Portal Homepage. 6. System will show the list all projects.	
Alternative Flows:	If projects are not their system will display message with empty fields.		
Exceptions:	1. Internet connection is not available. 2. Database is not responding.		

2.3.16 Upload project report

Here is the Use Case in which student will login through valid credentials and after successful login then student will see an open portal homepage where student will see a report upload button

by clicking that button, a dialog box will open where student will upload the file and system will check plagiarism.

Table 19 Upload project report

Use Case ID:	16		
Use Case Name:	Upload project report		
Created By:	M. Kamran	Last Updated By:	N/A
Date Created:	4/4/2022	Last Revision Date:	5/4/2022
Actors:	Student		
Description:	Student will be able to upload project report to check plagiarism.		
Trigger:	Click on upload report.		
Preconditions:	Student must be login.		
Post conditions:	Display percentage of plagiarism.		
Normal Flow:	1. User must login credentials. 3. Successful Login. 5. Click on Report Upload button.	2. Validate login credentials. 4. Open Portal Homepage. 6. Show dialog box to upload file. 7. System will check plagiarism on file. If plagiarism is less than 20% report will be added in database.	
Alternative Flows:	Show Error message that File cannot uploaded.		
Exceptions:	1. Internet connection is not available. 2. Database is not responding.		

2.3.17 Logout

Here is the Use Case in which super admin, admin and student can log out by clicking on log out button and system will allow them to log out of the system as shown:

Table 20 Logout to the system

Use Case ID:	17		
Use Case Name:	Logout to the system		
Created By:	Mahnoor	Last Updated By:	N/A
Date Created:	1/4/2022	Last Revision Date:	4/4/2022
Actors:	Super admin, Admin, Student		
Description:	Users can logout to the system.		
Trigger:	Click on the logout button.		
Preconditions:	User must be logged in.		
Post conditions:	User will be logged out.		
Normal Flow:	1. User will click on logout button to request for logging out. 3. Successfully log out.	2. System will allow user to logout of the system.	
Alternative Flows:	User will leave the current action.		
Exceptions:	1. Internet connection is not available. 2. Database is not responding.		

2.4 System sequence diagram

System sequence diagram (SSD) is a sequence diagram that shows, for a particular scenario of a use case, the events that external actors generate their order, and possible inter-system events.

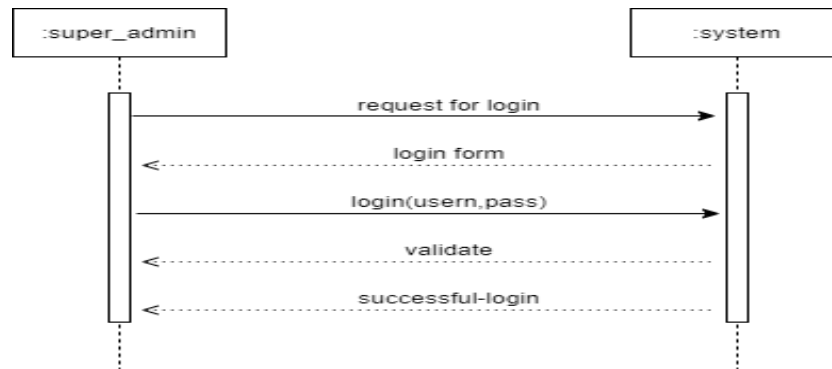


Figure 6 Super admin login

User will enter the credentials for login. The system will check the credentials for validation process. If the credentials are correct the user will be logged in otherwise an error message will be shown.

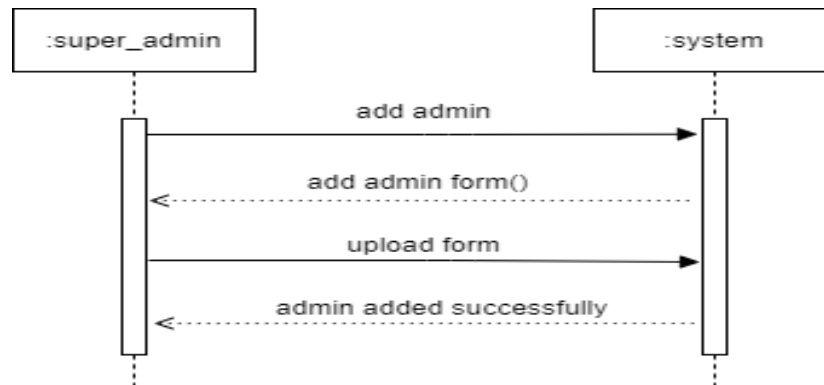


Figure 7 Add admin

Super admin has the authority to add admin. Super admin will add admin information and will upload form and the admin will be added.

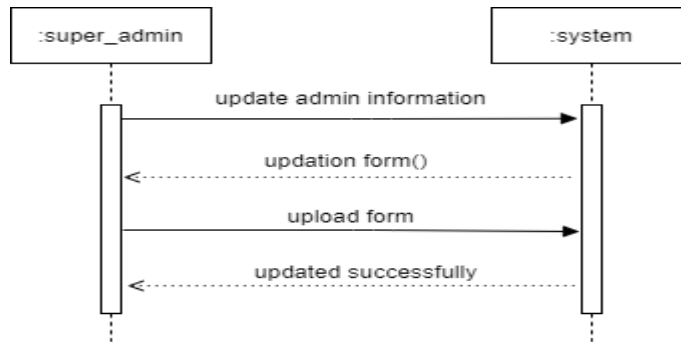


Figure 8 update admin information

Super admin can update the information of an admin. System will show an update form, super admin will update information and it will be updated.

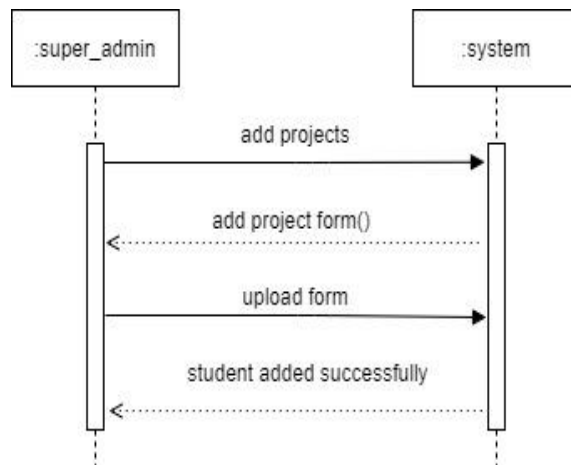


Figure 9 add project

Super admin can add project into the system if admin or student is unable to.

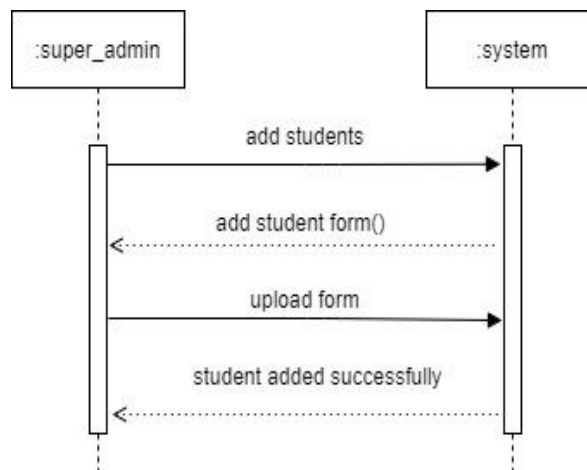


Figure 10 add student

Super admin has the access to add students into the system.

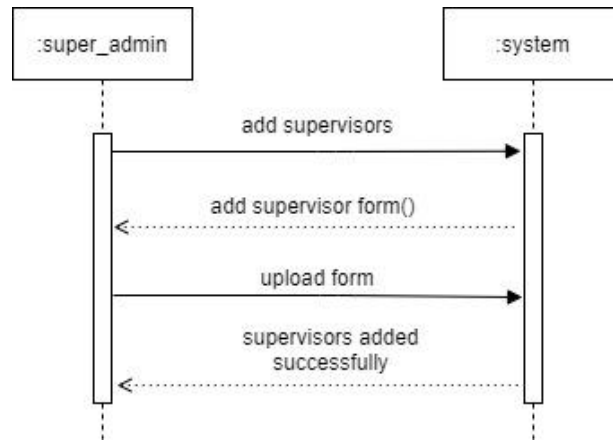


Figure 11 add supervisor

Super admin can add the information about supervisor in the system.

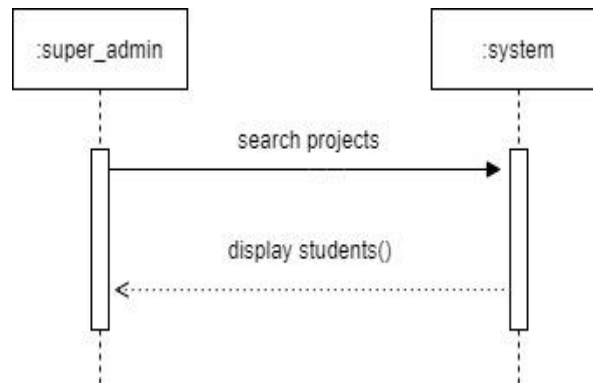


Figure 12 search project

Super admin can search for a specific project. And the system will display the project information if it exists.

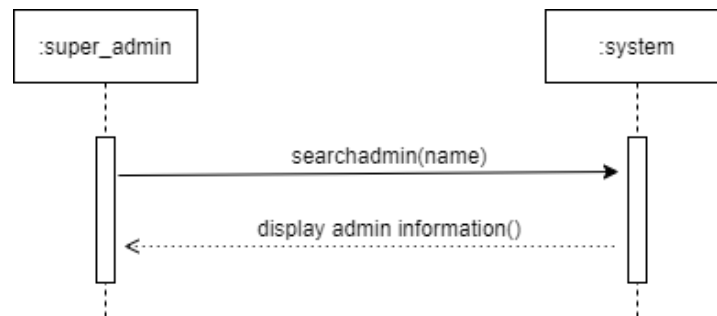


Figure 13 Search admin

Super admin can also search an admin by adding admin name in the search bar and the information will be displayed if that admin exists.

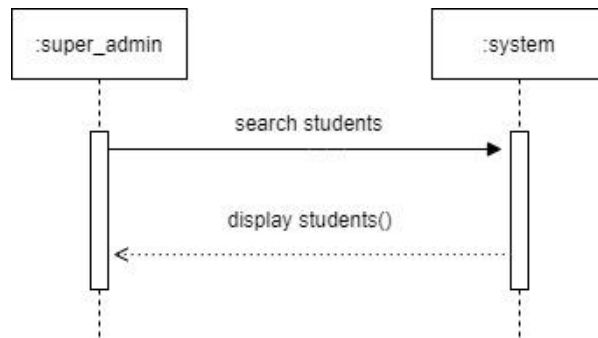


Figure 14 search students

Super admin can search for a specific student. And the system will display the student information if it exists.

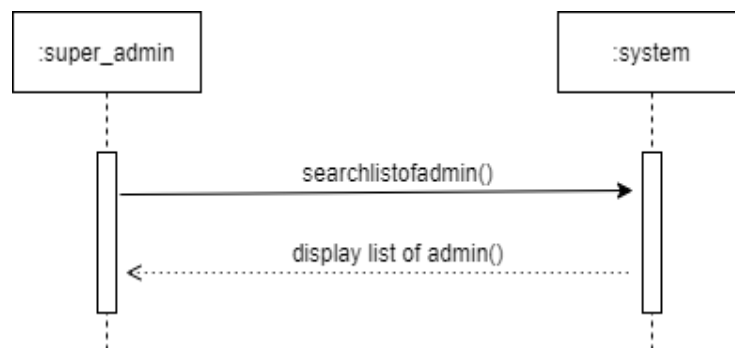


Figure 15 list of admin

Super admin can also the list of admin. The system will get the list of admin and will display it to the super admin.

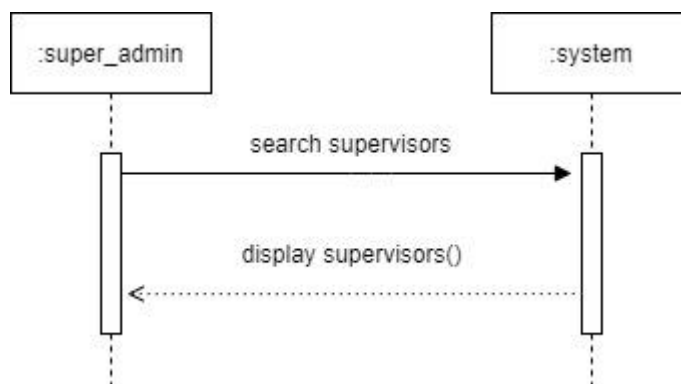


Figure 16 search supervisor

Super admin can search for a supervisors. And the system will display the supervisor information if it exists.

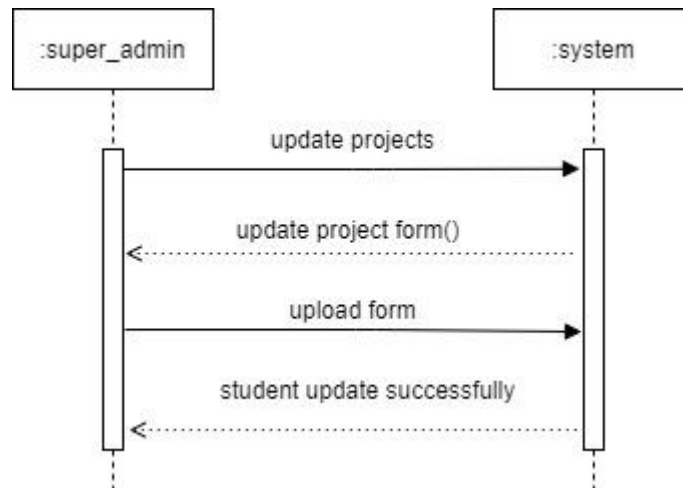


Figure 17 update project

Super admin can update the project. And the system will update the project information.

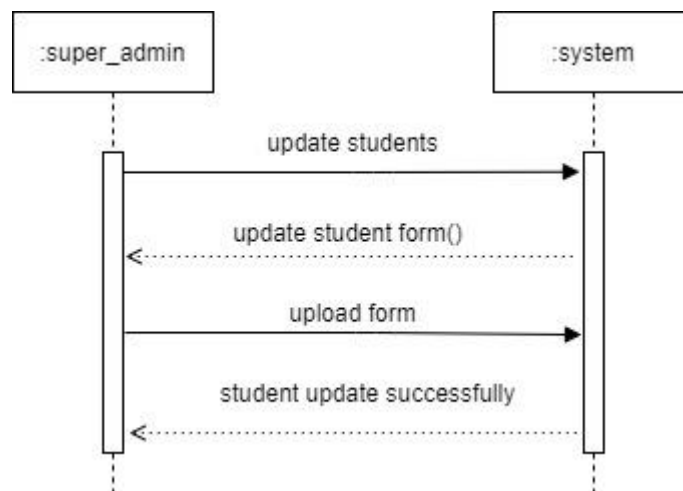


Figure 18 update student

Super admin can update the information of student. And the system will update the student information if exists.

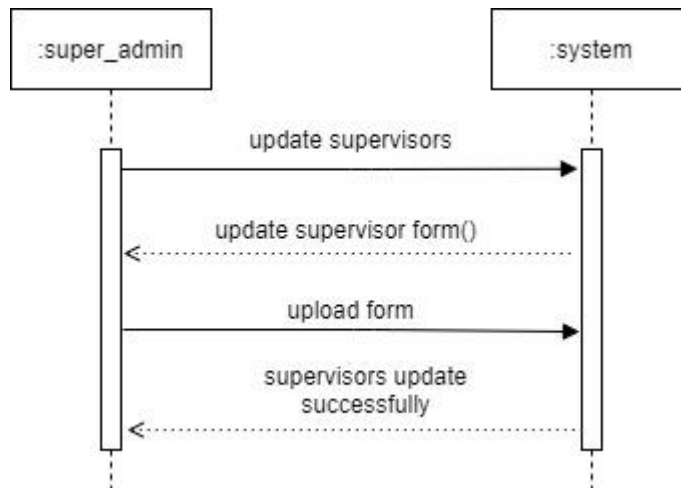


Figure 19 update supervisor

Super admin can update the information of supervisor. And the system will update the supervisor information if exists.

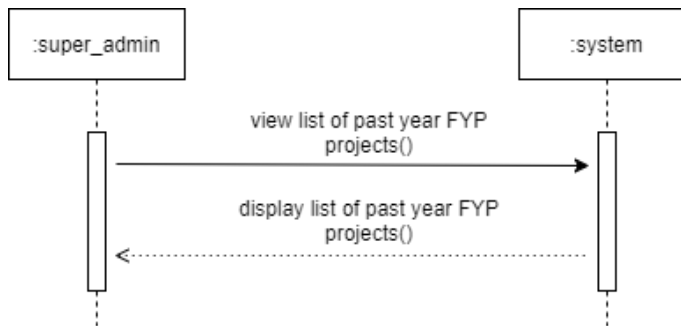


Figure 20 list of past fyp project

Super admin can view the list of past FYP projects. And the system will view the list of past FYP projects.

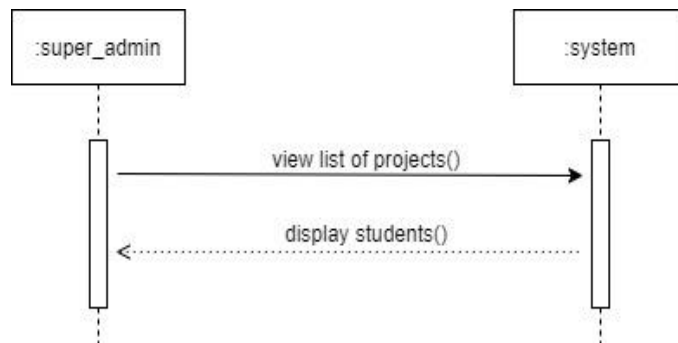


Figure 21 list of projects

Super admin can view the list of projects. And the system will view the list of projects.

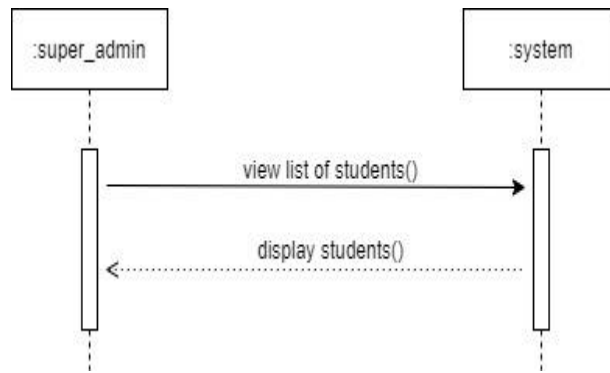


Figure 22 list of students

Super admin can view the list of students. And the system will view the list of students.

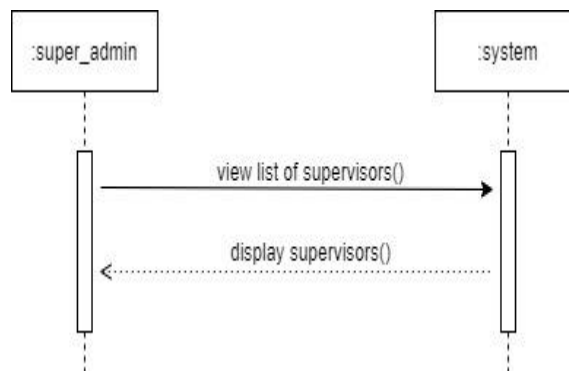


Figure 23 list of supervisors

Super admin can view the list of supervisor. And the system will view the list of supervisor.

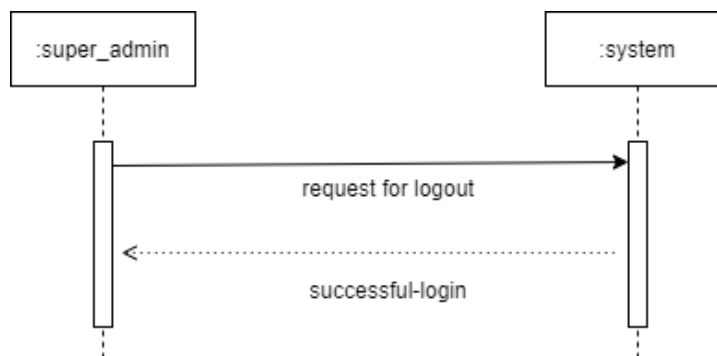


Figure 24 Super admin logout

Super admin can request to logout the system.



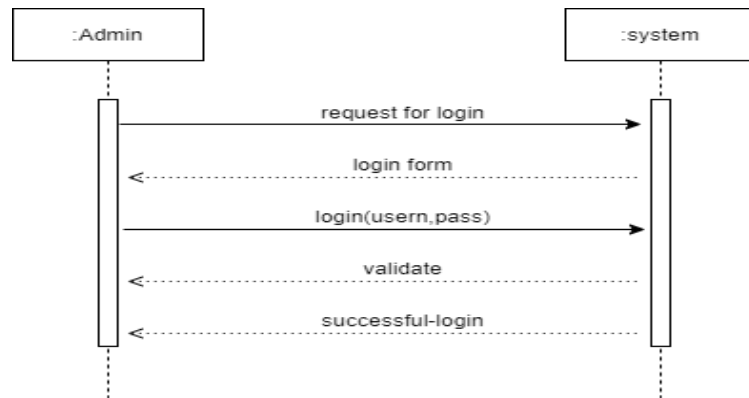


Figure 25 Admin login

Admin can request to login. The admin will enter login credentials and the system will check the credentials for validation process. If the credentials are correct the user will be logged in otherwise an error message will be shown.

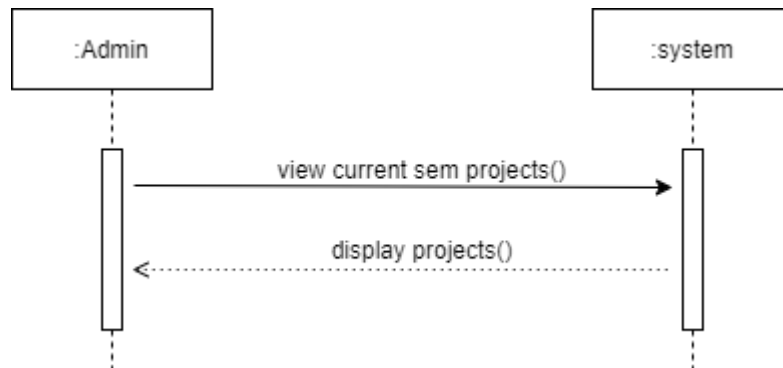


Figure 26 Current project

Admin can see the list of current semester projects name, so that project won't get repeated.

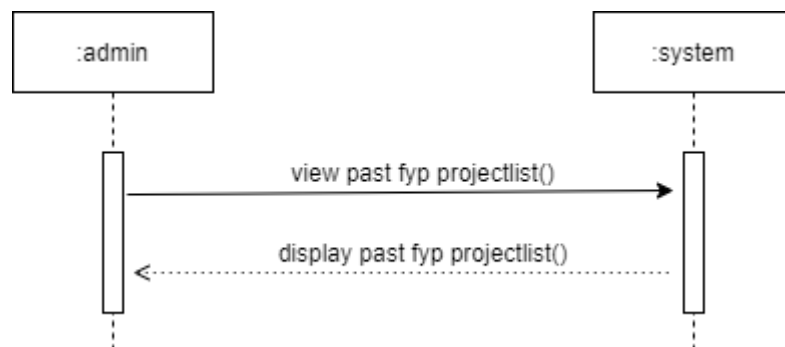


Figure 27 Past projects

Admin can view the list of past fyp projects. If any project exist it will be displayed.

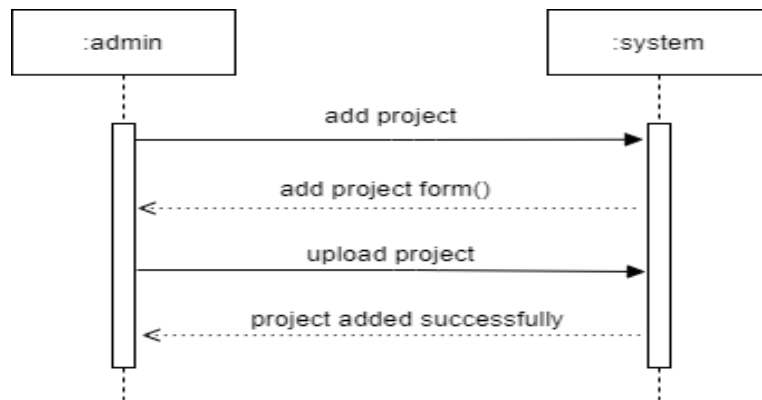


Figure 28 Add project

Admin can add projects into the system by adding the information about the project and upload the file.

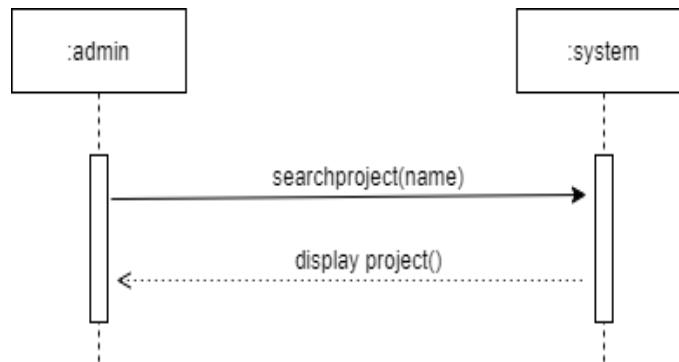


Figure 29 Search projects

Admin can search for specific project. If the project exists the information regarding the project will be displayed. On the other hand if the project does not exists a message will be displayed that the project does not exists.



Figure 30 Delete projects

Admin can delete projects. As if the wrong file is uploaded the admin can delete the project to upload the correct one.

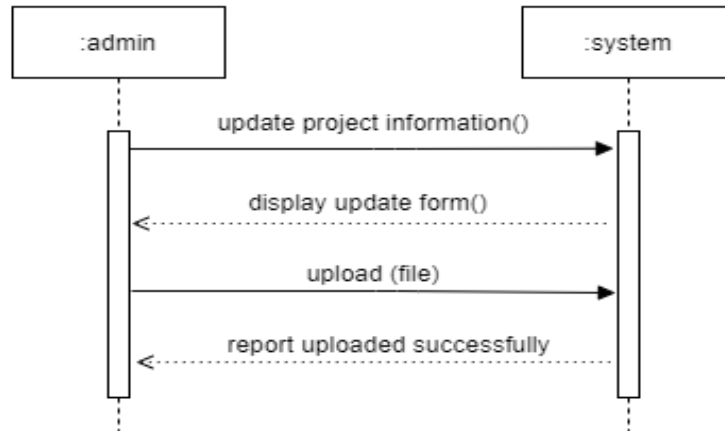


Figure 31 Update project information

Admin can update the project information. If the project exists the form will be displayed and admin can update the information and it will be updated.

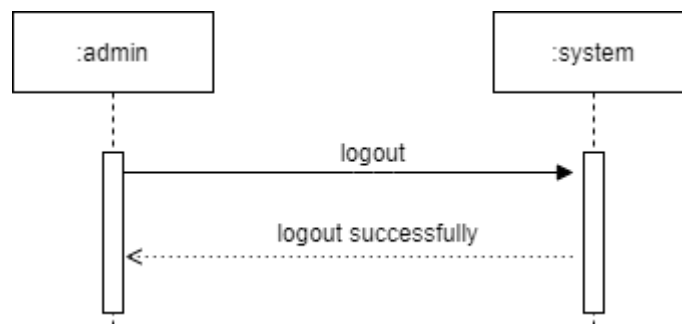


Figure 32 Admin logout

Admin can logout from the system.

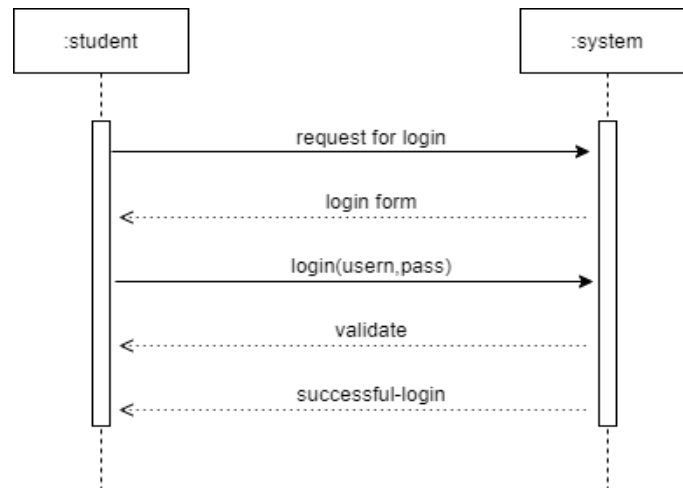


Figure 33 Student login

Student can request to login. The admin will enter login credentials and the system will check the credentials for validation process. If the credentials are correct the user will be logged in otherwise an error message will be shown.

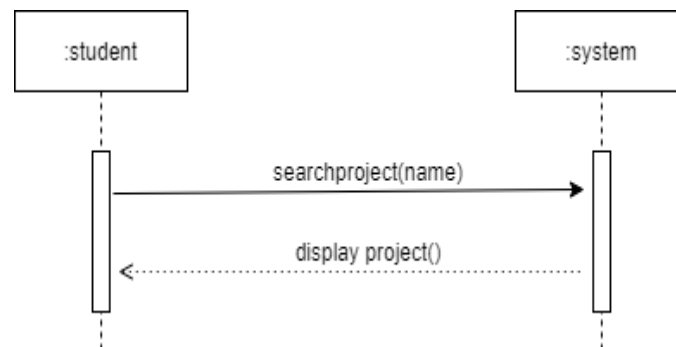


Figure 34 Search project

Just like admin student can also search for specific project. If the project exists the information regarding the project will be displayed. Otherwise a message will be displayed that the project does not exists.

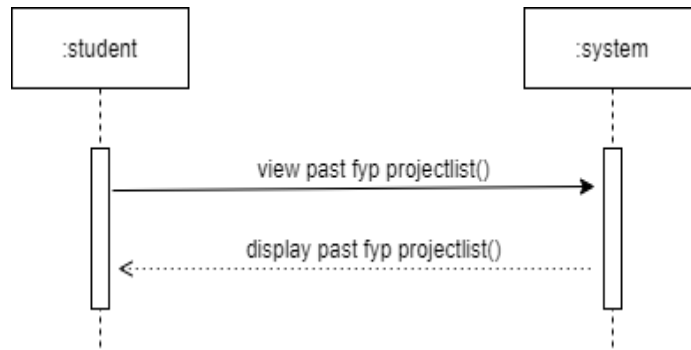


Figure 35 Project list

Student can view the list of past fyp projects. If any project exist it will be displayed.

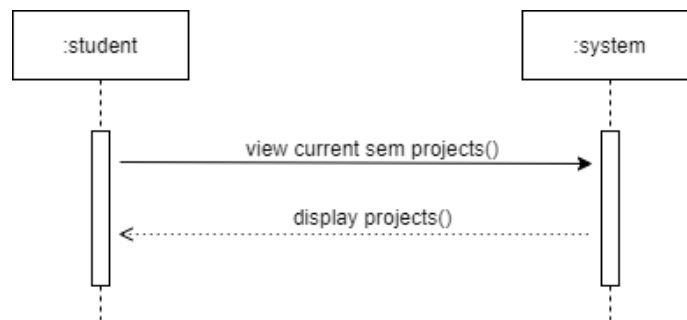


Figure 36 Current project list

The student is able to see the list of current semester project to avoid reputation in the project. If list is available it will be displayed.

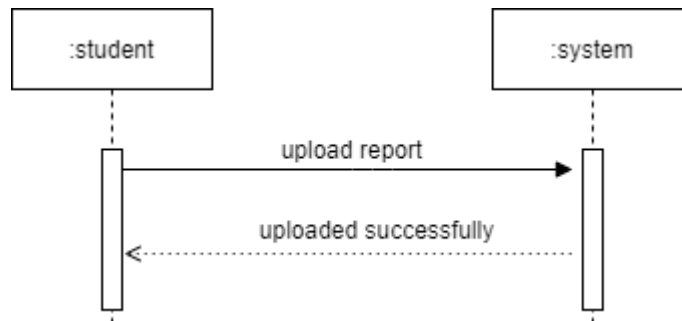


Figure 37 Upload report

Student can upload the report for plagiarism check.

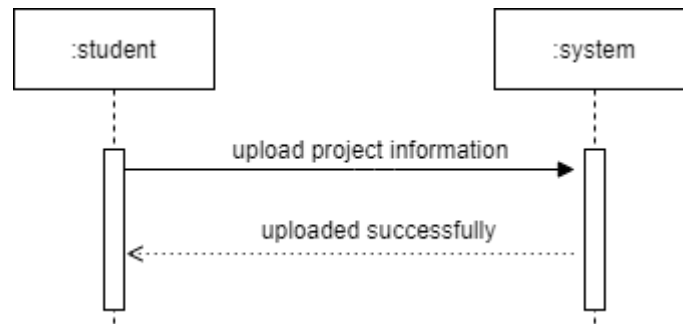


Figure 38 upload project information

Student can upload the information about the project.

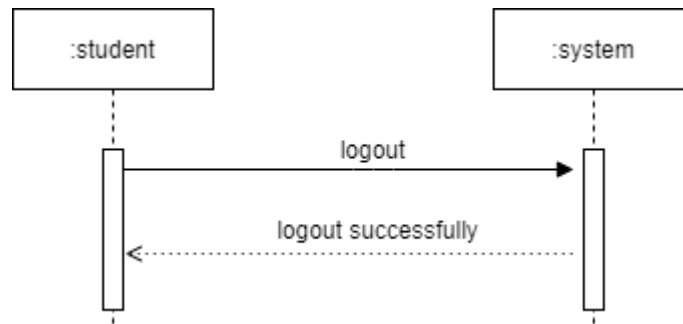


Figure 39 student logout

Student can logout from the system.

2.5 Domain Model

The basic concepts of the domain are customer, order, food item and dining table are shown in the following figure representing domain model.

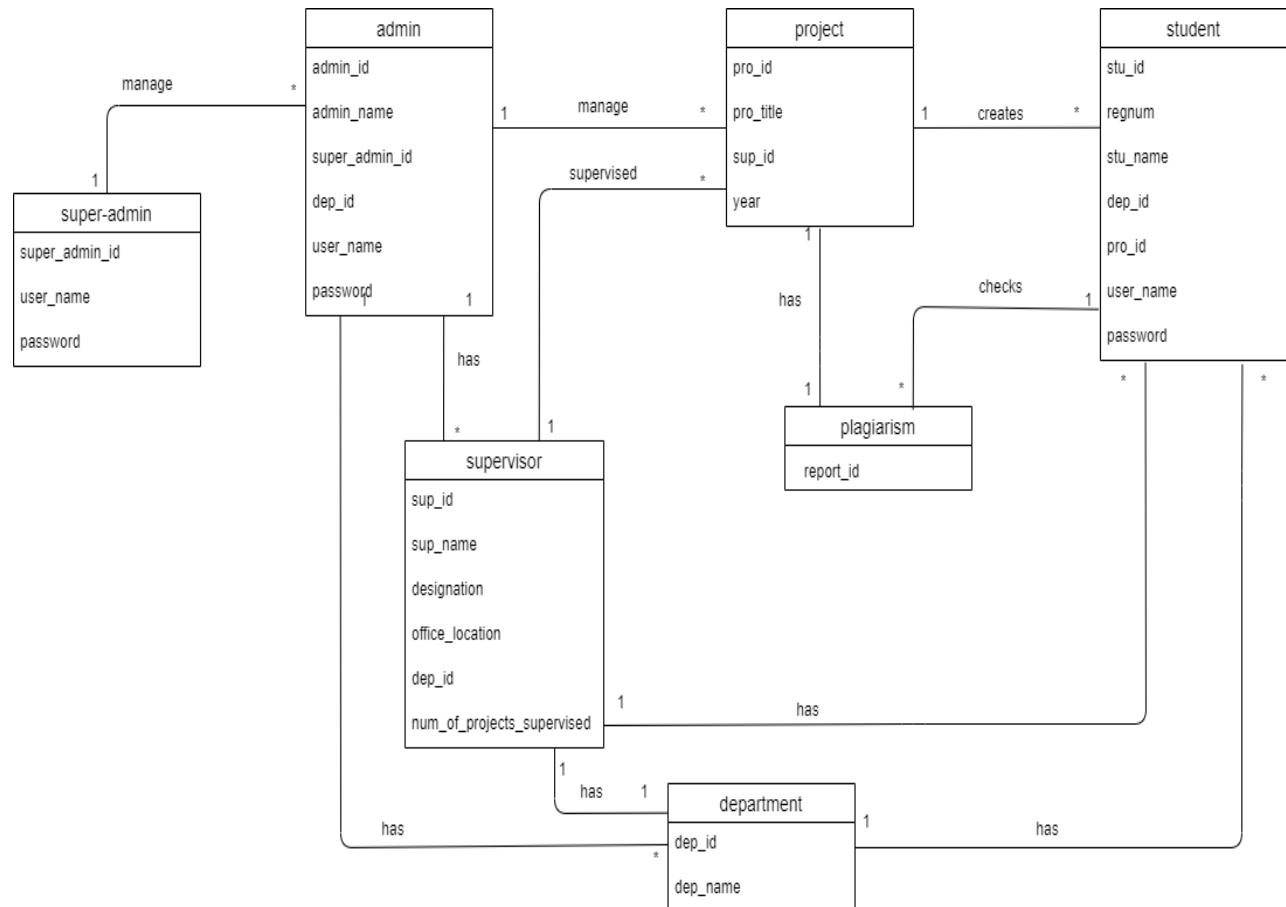


Figure 40 Domain model

Chapter 3

System design

In this chapter we deal with the information that is complementary to the code. Also in this chapter we deal with structural as well as behavioral view. Structural view includes Architecture diagram, class diagram etc. whereas behavioral view include sequence diagram, activity diagram etc. At this we will discuss the database schema as well as entity relationship diagram of our system abstractly.

3.1 Software Architecture

While dealing with software architecture we know that it describes the organization or structure of a system, where the system represents a collection of components that accomplish a specific function or set of functions.

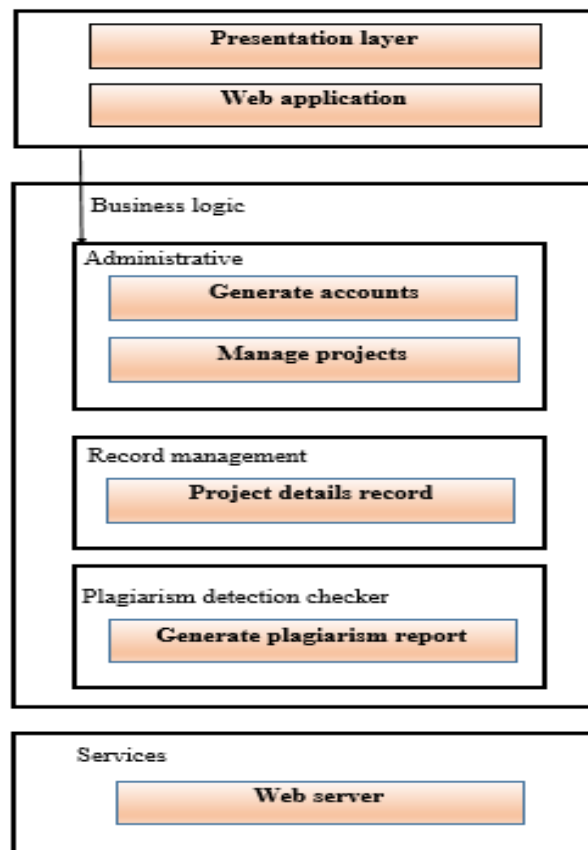


Figure 41 Software Architecture

3.2 Class Diagram

The class diagram describes the attributes and operations of a class and the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages:

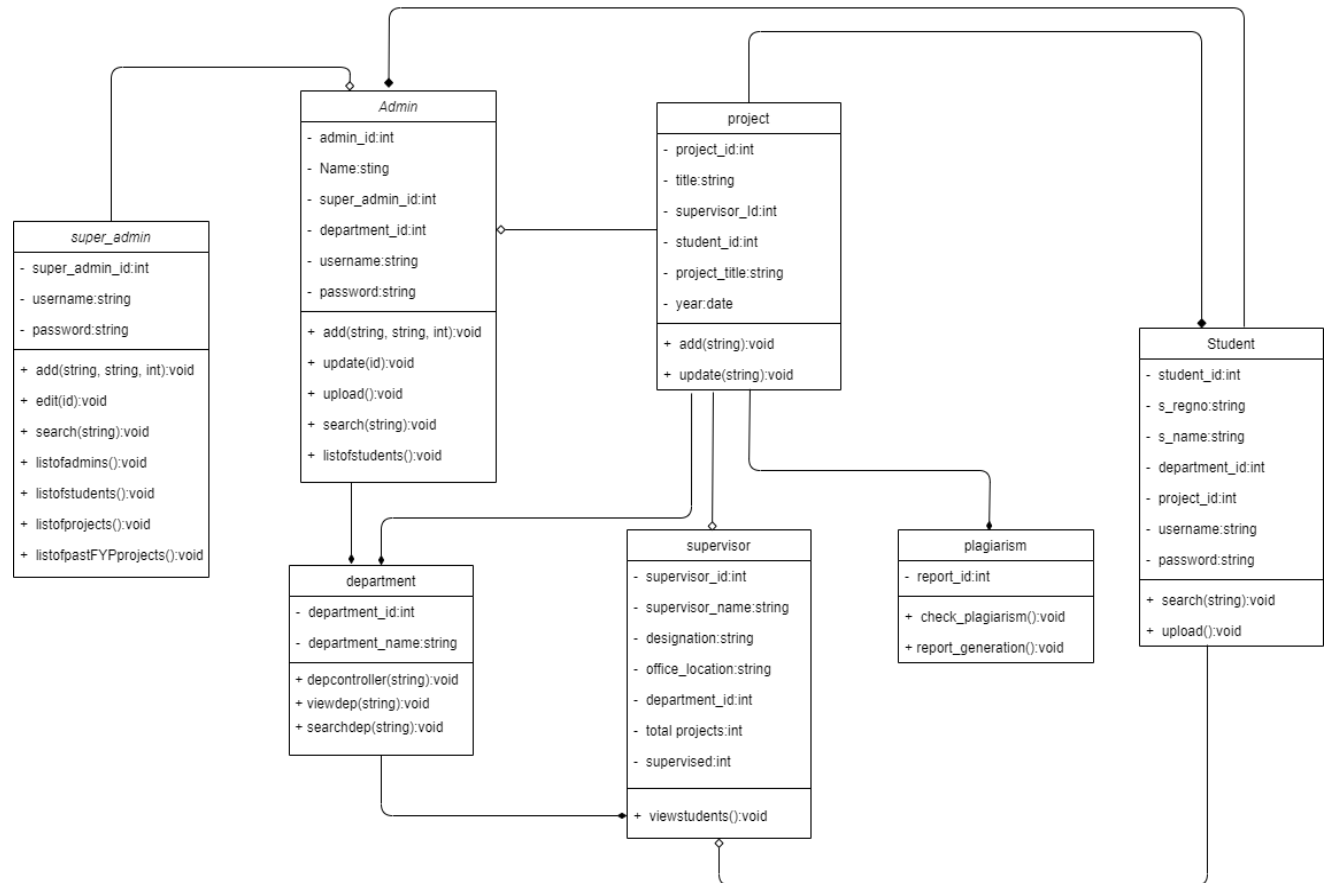


Figure 42 Class diagram

3.3 Sequence Diagram

Sequence Diagram model the flow of logic within your system in a visual manner enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes.

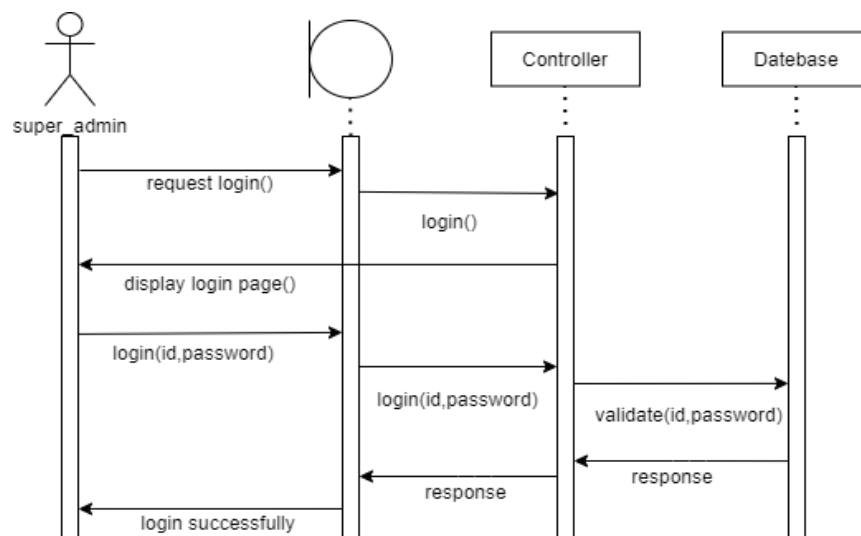


Figure 43 Super admin login

Super admin request for login. The controller will display the login page. Super admin will enter the login credentials those will be validated through the database. If the validation process is a success the super admin will be logged in.

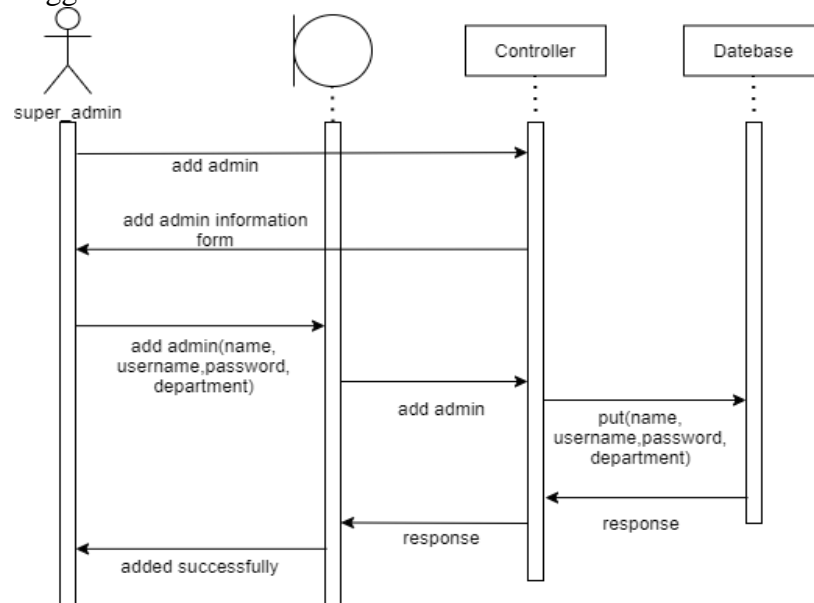


Figure 44 Add admin

Super admin can add admin. For that super admin have to add information about the admin like

name, username, password and department. This information will be stored in the database and the admin will be added successfully.

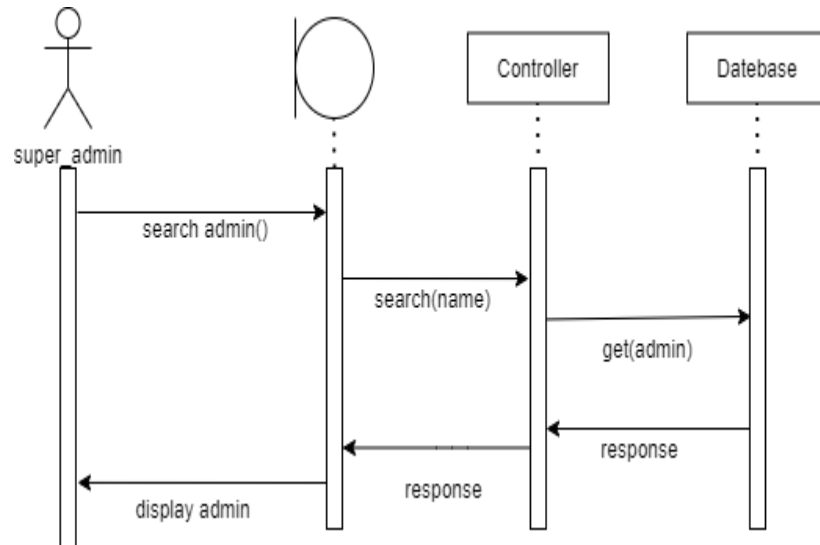


Figure 45 Search admin

Super admin can search for specific admin. Super admin will write the name of the admin in the search bar. The controller will search for the admin and if that admin exists the information will be displayed.

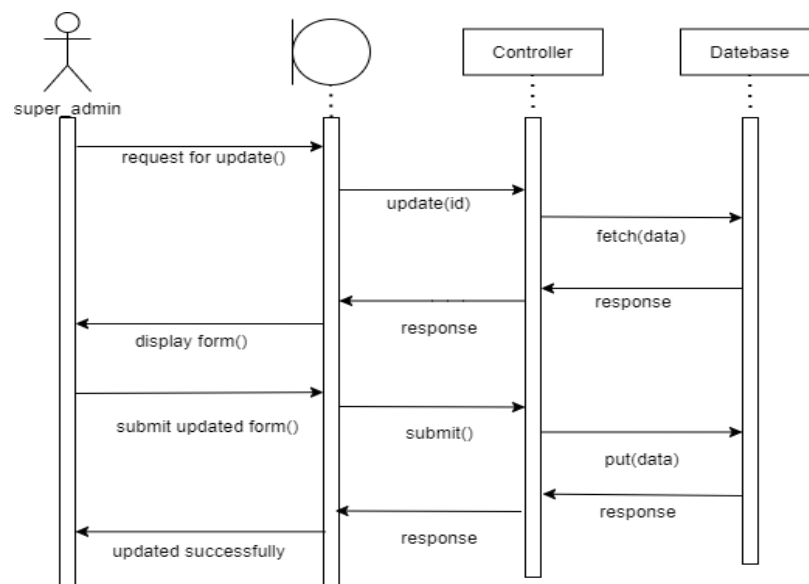


Figure 46 Update admin

Super admin can request to update the information about an admin. The controller will fetch the data

from the database and will display a form. Super admin can update the information and submit it the controller will put the updated information in the database.

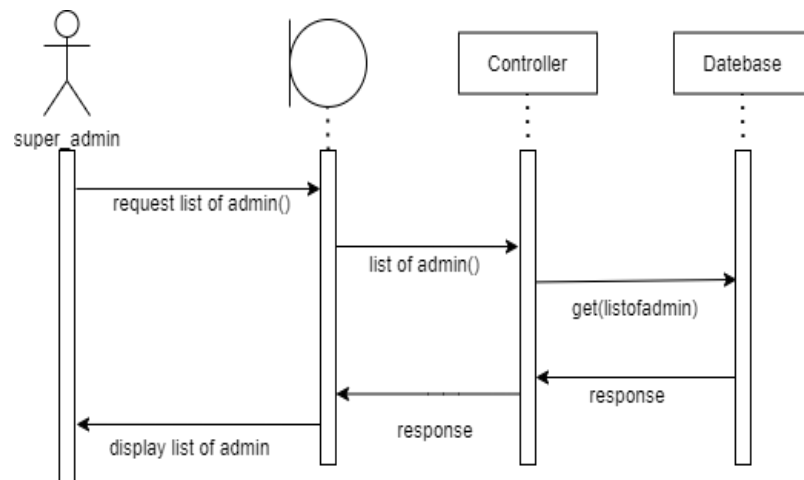


Figure 47 list of admin

Super admin can request for the list of admin. The controller will search for the list. If it exists it will be displayed to the user.

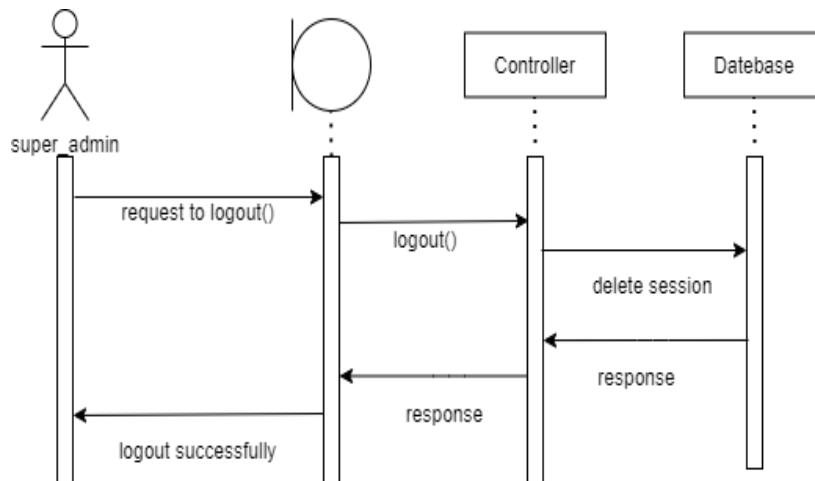


Figure 48 Super Admin logout

Super admin can request for logout. The controller will delete the session from the database and the user will be logged out.

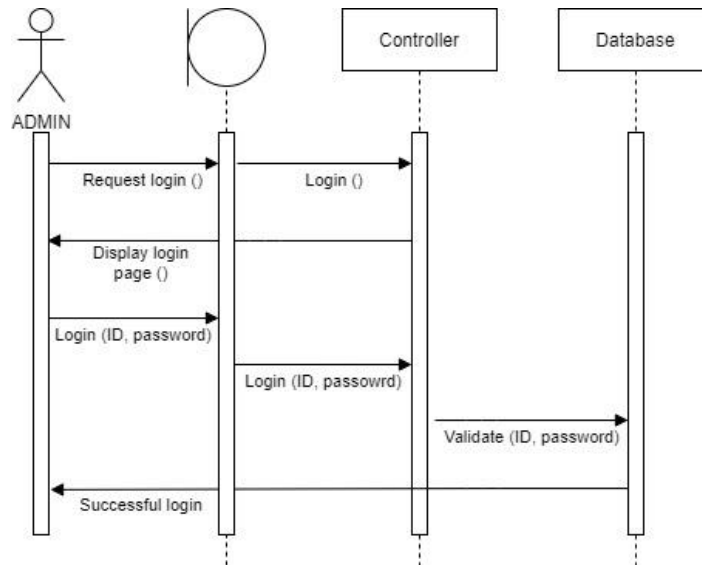


Figure 49 Admin login

Admin request for login. The controller will display the login page. Admin will enter the login credentials those will be validated through the database. If the validation process is a success the admin will be logged in.

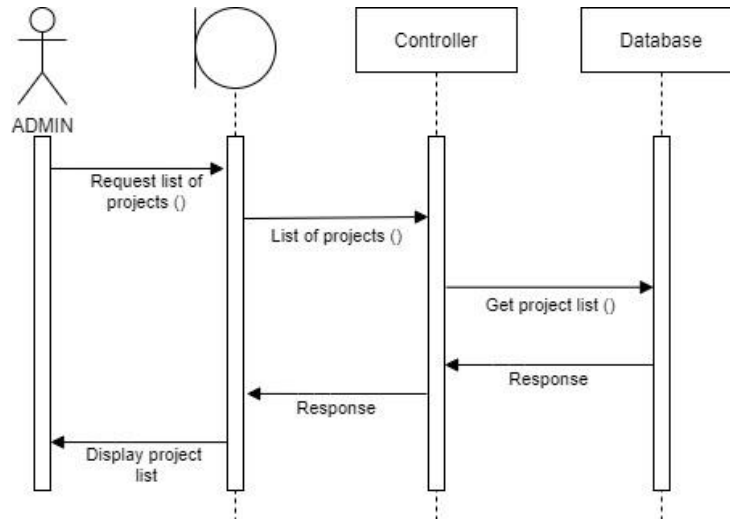


Figure 50 Request list of project

The admin can request to list the list of project. The controller will check for the data in the database if the data exists it will be displayed.

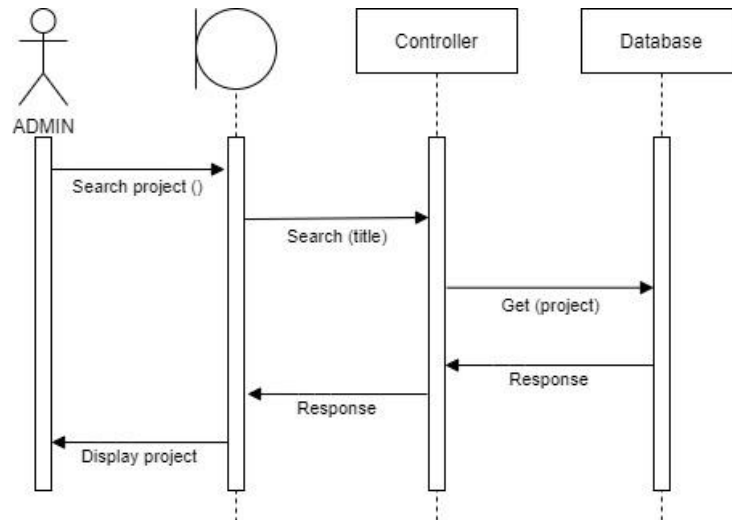


Figure 51 Search project

Admin can search for project. The admin will write the title in the search bar the controller will search for the specific project in the database. If that project exists the information regarding that project will be displayed otherwise a message will be displayed that the project does not exists.

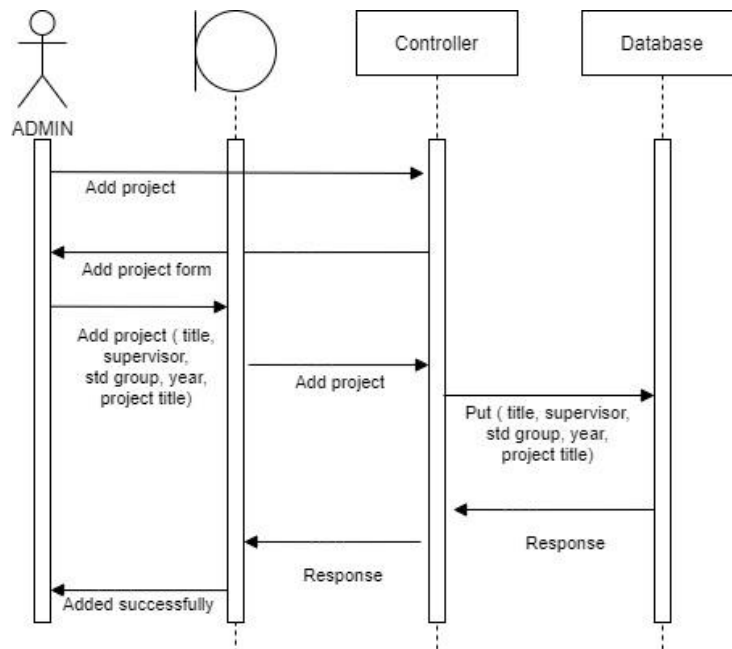


Figure 52 Add project

Admin can add a project in the database. An add project form will be displayed and the admin will add the information like title, supervisor, year etc. The controller will add the information in the database.

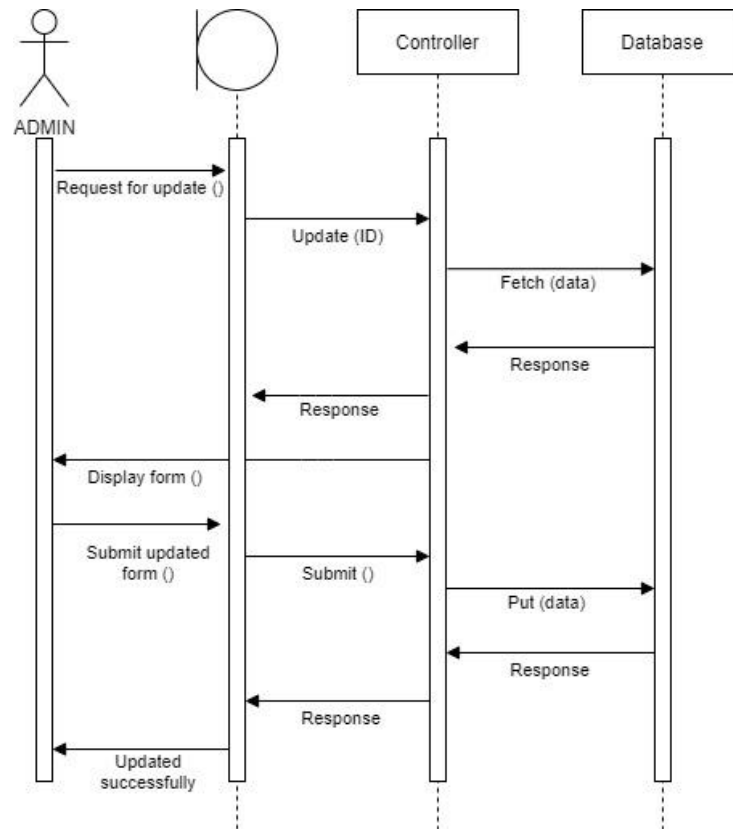


Figure 53 update project

Admin can request to update the information about the project. The controller will fetch the data from the database and will display a form. The admin can update the information and submit it. The controller will add the updated information in the database.

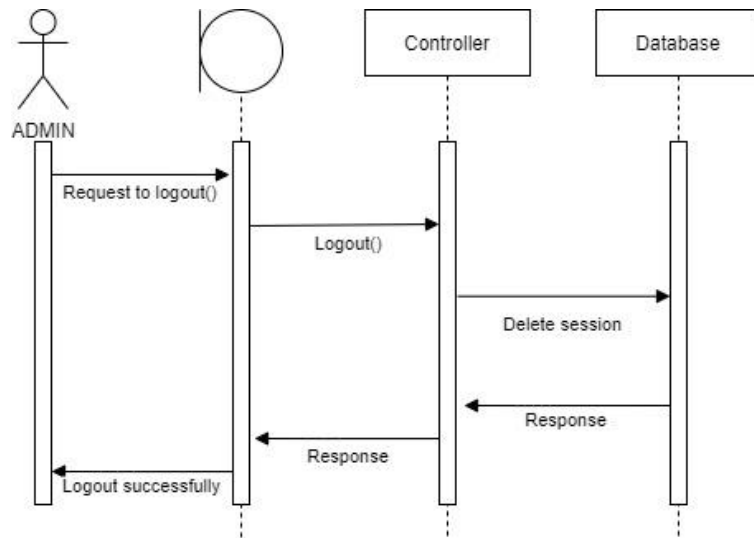


Figure 54 Admin logout

Admin can request for logout. The session will be deleted and the admin will be logged out.

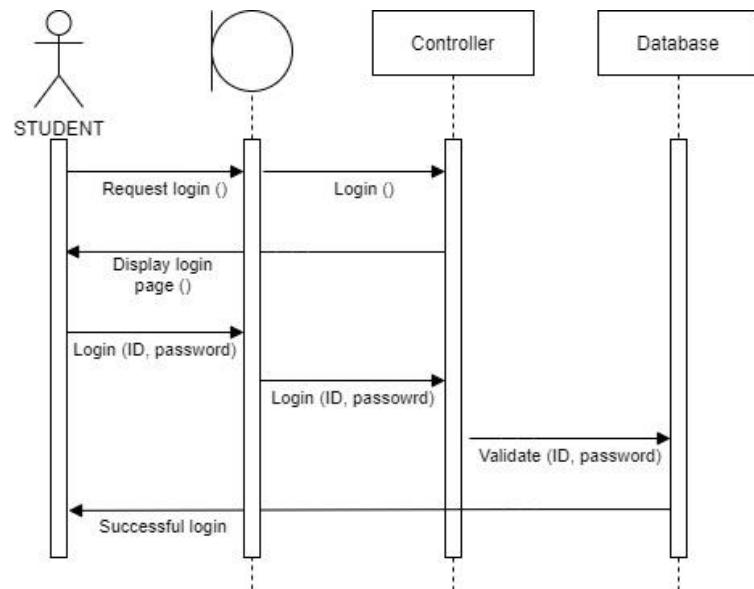


Figure 55 Student login

Student request for login. The controller will display the login page. Student will enter the login credentials those will be validated through the database. If the validation process is a success the student will be logged in.

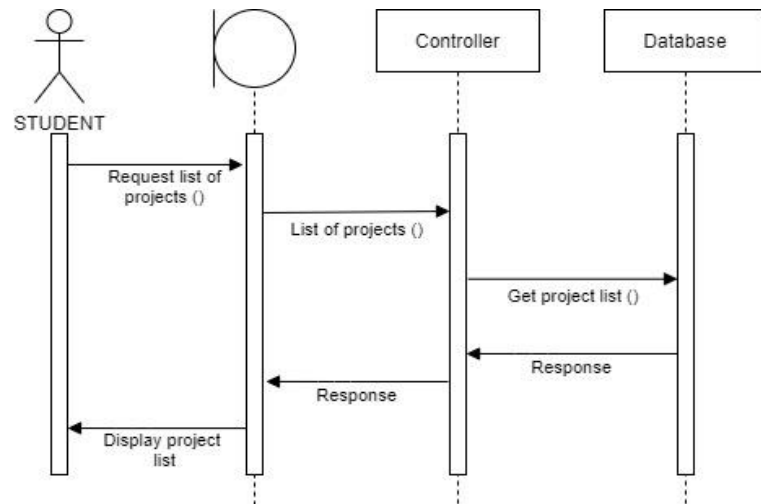


Figure 56 Request list of projects

The student can request to list the list of project. The controller will check for the data in the database if the data exists it will be displayed.

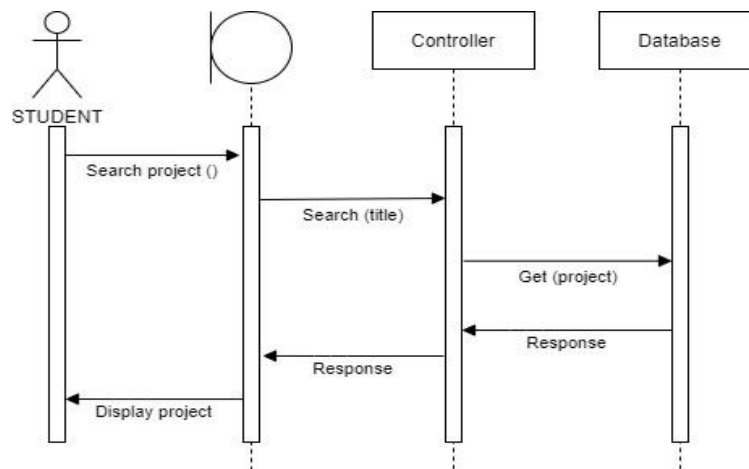


Figure 57 Search project

Student can search for project. The student will write the title in the search bar the controller will search for the specific project in the database. If that project exists the information regarding that project will be displayed otherwise a message will be displayed that the project does not exists.

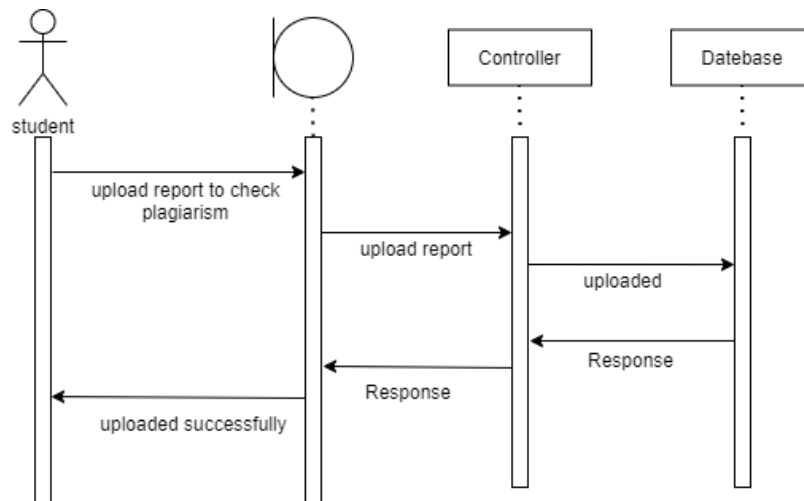


Figure 58 Upload report to check plagiarism

Student will upload the report on the system for plagiarism check. The controller will upload the report into the database for checking similarity.

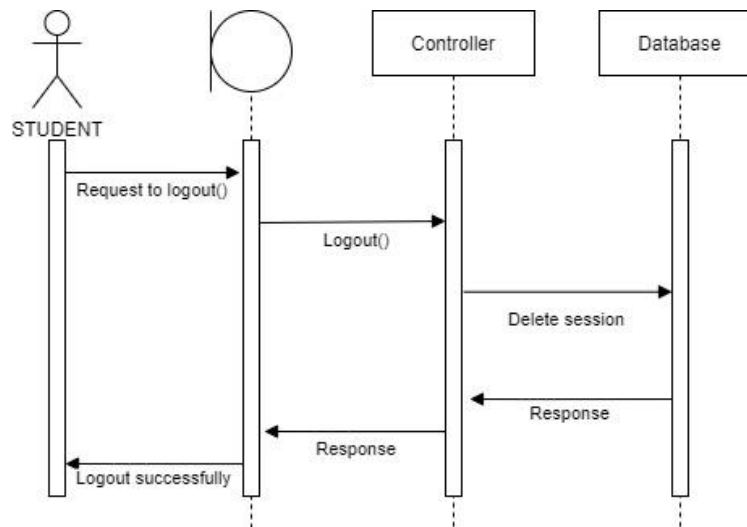


Figure 59 Student logout

Student can request for logout. The session will be deleted and the user will be signed out.

3.4 Database schema

A database schema represents the logical configuration of all or part of a relational database. It can exist both as a visual representation and as a set of rules known as integrity constraints that govern a database. These rules are expressed in a data definition language, such as SQL.

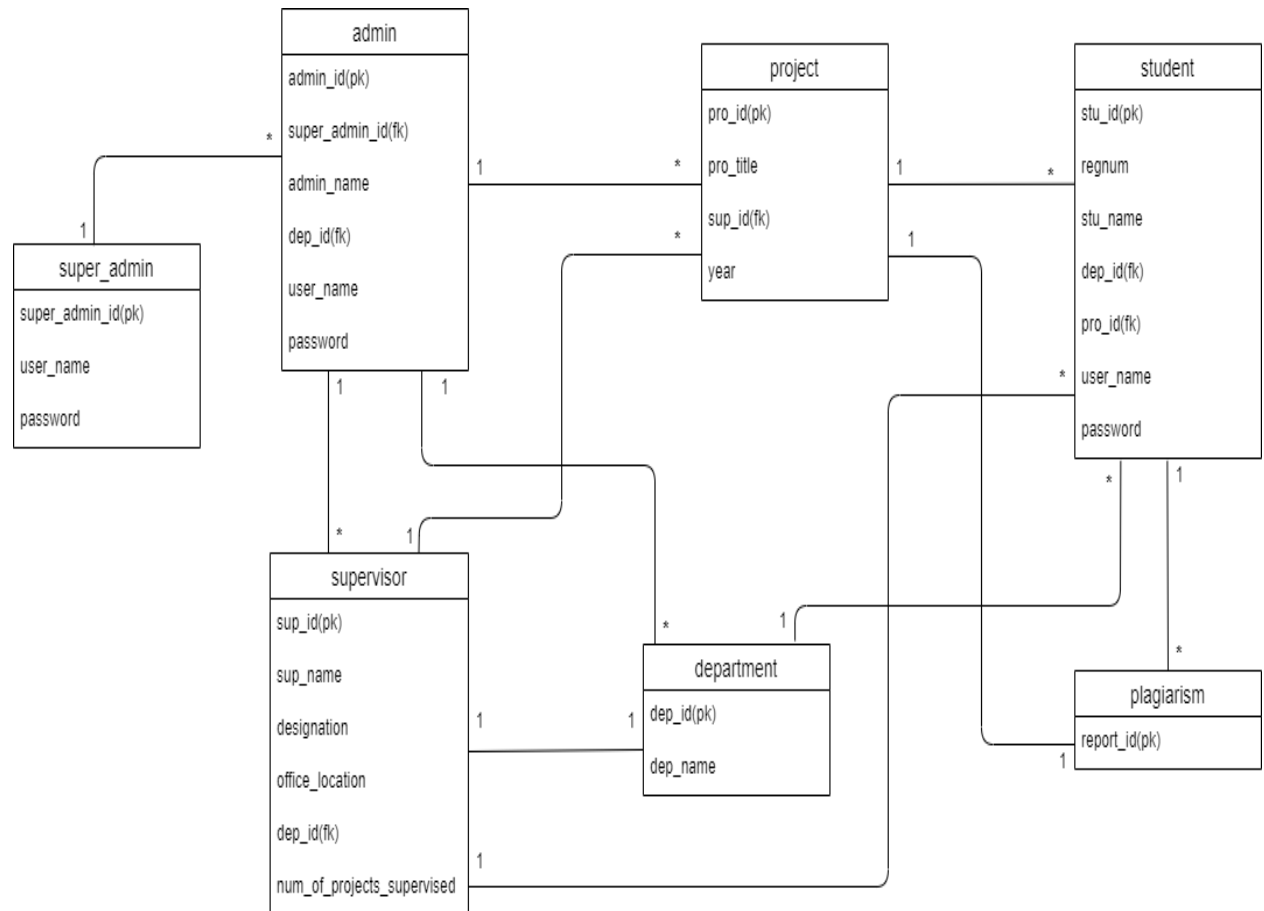


Figure 60 Database Schema

3.5 User Interface Design

User Interface (UI) Design focuses on anticipating what users might need to do and ensuring that the interface has elements that are easy to access, understand, and use to facilitate those actions. UI brings together concepts from interaction design, visual design, and information architecture.

User Sign in:

This is the login screen for the user in which user enter username and password. User have to enter his/her unique username in “username” column. In “password” field user, enter his/her password.

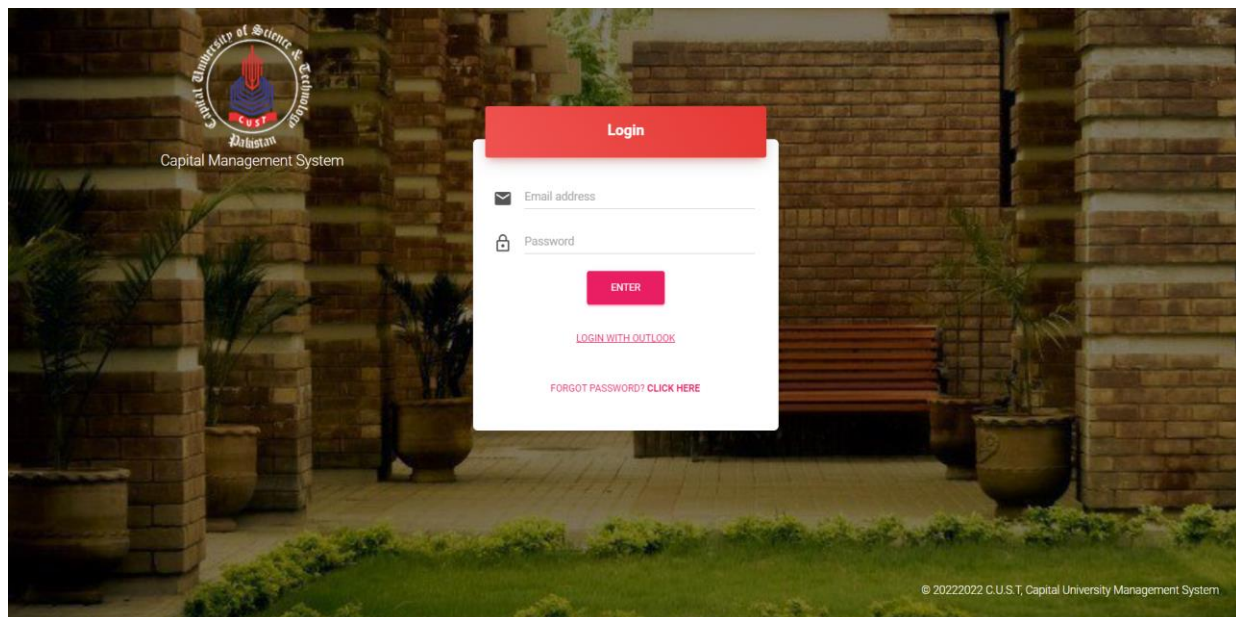



Figure 61 Sign in GUI

After signing to the system the admin will be able to see the list of current projects. The GUI page consist of project title, supervisor, team, year, department, report etc. as shown below:



Current FYP

Past FYP

Check Plagiarism

Current Final Year Projects

Copy

Excel

PDF

Column visibility

Search:

Project Title	Supervisor	Team	Year	Department	Report	Action
Car Parking System	Omaid Ghayur	Ali,Zahid,Umaid	2022	CS		
FYP Records and Plagiarism detector	Dr Mariam	Mahnoor,Asim,Kamran	2022	CS		
Smart Home	Fahad Majeed	Arslan,Asim,Umaid	2022	CS		
Vote Casting	Dr Qamar Mahmood	Mohsin,Ayesha,Ansa	2022	CS		

Showing 1 to 4 of 4 entries

Previous


1

Next

Add Final Year Project

Figure 62 List of projects

If there is a wrong input the admin can view, update, and delete project. The admin can also add a project which is recently done:



Current FYP

Past FYP

Check Plagiarism

Current Final Year Projects

Copy

Excel

PDF

Column visibility

Search:

Project Title	Supervisor	Team	Year	Department	Report	Action
Car Parking System	Omaid Ghayur	Ali,Zahid,Umaid	2022	CS		
FYP Records and Plagiarism detector	Dr Mariam	Mahnoor,Asim,Kamran	2022	CS		
Smart Home	Fahad Majeed	Arslan,Asim,Umaid	2022	CS		
Vote Casting	Dr Qamar Mahmood	Mohsin,Ayesha,Ansa	2022	CS		

Showing 1 to 4 of 4 entries

Previous

1

Next

Add Final Year Project


View

Update

Delete

Figure 63 CRUD operation

The model to add a new project in the list of project is shown below:



Current FYP

Past FYP

Reports
Past FYP

Reports

Current Final Year Projects

Add Current Final Year Details

Title

Supervisor Name

Group Member's Name


Year

Department

Report
 No file chosen

Figure 64 Add project

To see the pass final year projects user have to select Past FYP from the list. The past projects will be shown, the GUI for Past projects is given below:



Current FYP

Past FYP

Check Plagiarism

Past Final Year Projects

Copy Excel PDF Column visibility

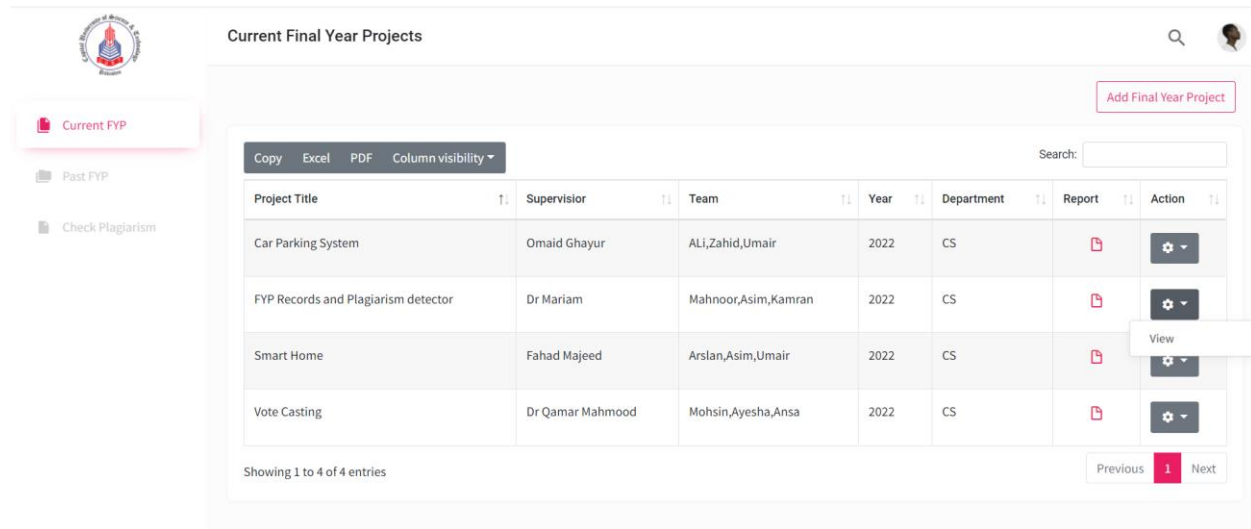
Search:

Project Title	Supervisor	Team	Year	Department	Report	Action
Car Parking System	Omaid Ghayur	ALI,Zahid,Umaid	2022	CS		
FYP Records and Plagiarism detector	Dr Mariam	Mahnoor,Asim,Kamran	2022	CS		
Smart Home	Fahad Majeed	Arslan,Asim,Umaid	2022	CS		
Vote Casting	Dr Qamar Mahmood	Mohsin,Ayesha,Ansa	2022	CS		

Showing 1 to 4 of 4 entries

Figure 65 past projects

The student is able to view the report

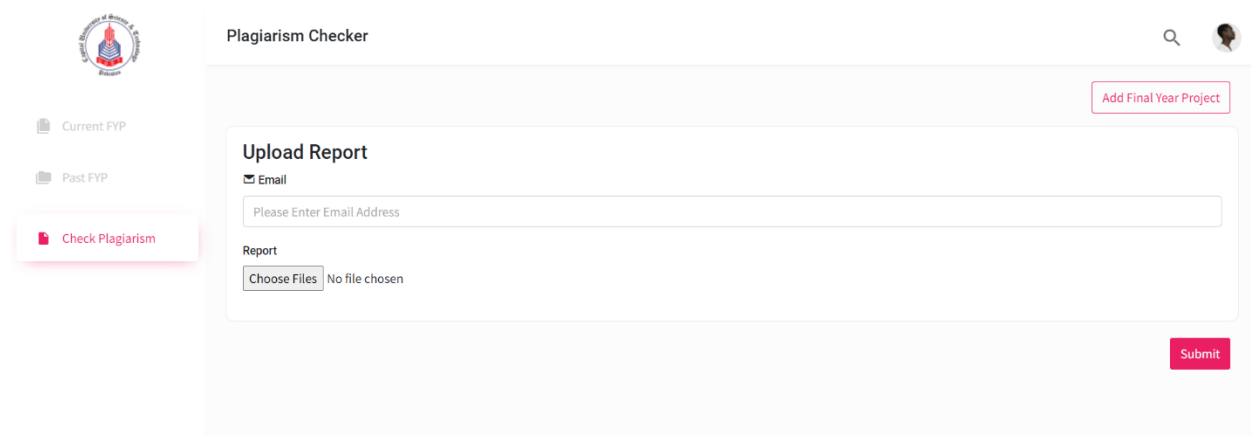


The screenshot shows a web application interface for 'Current Final Year Projects'. On the left is a sidebar with a logo and three menu items: 'Current FYP' (highlighted), 'Past FYP', and 'Check Plagiarism'. The main content area has a title 'Current Final Year Projects' and a search bar. Below the title is a table with columns: Project Title, Supervisor, Team, Year, Department, Report, and Action. The table contains four entries. The 'Report' column has a red document icon, and the 'Action' column has a gear icon. A 'View' button is visible next to the 'Smart Home' project. At the bottom of the table, it says 'Showing 1 to 4 of 4 entries' and has 'Previous', '1', and 'Next' navigation links.

Project Title	Supervisor	Team	Year	Department	Report	Action
Car Parking System	Omaid Ghayur	Ali,Zahid,Umaid	2022	CS		
FYP Records and Plagiarism detector	Dr Mariam	Mahnoor,Asim,Kamran	2022	CS		
Smart Home	Fahad Majeed	Arslan,Asim,Umaid	2022	CS		
Vote Casting	Dr Qamar Mahmood	Mohsin,Ayesha,Ansa	2022	CS		

Figure 66 view report

The GUI for report uploading for plagiarism checking:



The screenshot shows a web application interface for 'Plagiarism Checker'. On the left is a sidebar with a logo and three menu items: 'Current FYP', 'Past FYP', and 'Check Plagiarism' (highlighted). The main content area has a title 'Plagiarism Checker' and a search bar. Below the title is a form titled 'Upload Report'. The form has two sections: 'Email' with a text input field labeled 'Please Enter Email Address', and 'Report' with a 'Choose Files' button and the text 'No file chosen'. A 'Submit' button is at the bottom right of the form.

Figure 67 upload report for plagiarism checking

Chapter 4

Software Development

The Implementation section is similar to the Specification and Design section in that it describes the system, but it does so at a finer level of detail, down to the code level. This section is about the realization of the concepts and ideas developed earlier.

4.1 Coding Standards

The indentation, declaration, naming convention used while coding the project as follows:

- Indentation is used properly in all of our coding.
- Camel Case is used for attributes declaration.
- Pascal Case for method declaration.

4.2 Development Environment

The development environment provides developers an interface and helpful see of the advancement handle which incorporates composing code, testing the same and bundling the construct so that it can be deployed.

4.2.1 Visual Studio code

In general, the build process is very simple in this tool by building your code, you can quickly identify compile time errors such as incorrect syntax, misspelled keywords and type mismatches. You can also detect and correct runtime errors such as semantic errors and errors by building and running debug versions of the code.

4.2.2 Apache MYSQL

Apache is the web server that processes requests and serves web assets and content via HTTP. MySQL is the database that stores all your information in an easily queried format. We connected MySQL and Django as all the data will be stored in MySQL.

4.3 Code Snippet

Here are the code snippets of major functionality of the system which show how the system interacts and performs various operations. It will discuss the logic of the system that will be implemented in the code. Some snippets code which are given below:

```
urlpatterns = [
    path('_views.ShowLoginPage', name="ShowLoginPage"),
    path('doLogin', views.DoLogin, name="doLogin"),
    path('baseClass', hodviews.BaseClass, name="baseClass"),
    path('add_admin', hodviews.AddAdmin, name="add_admin"),
    path('manage_admin', hodviews.Manage_Admin, name="manage_admin"),
    path('admin_save', hodviews.Admin_Save, name="admin_save"),
    path('edit_admin/<str:admin_id>', hodviews.Edit_Admin, name="edit_admin"),
    path('edit_admin_save', hodviews.Edit_admin_Save, name="edit_admin_save"),
    path('addsupervisor', hodviews.AddSupervisor, name="addsupervisor"),
    path('supervisor_save', hodviews.Add_Supervisor_Save, name="supervisor_save"),
    path('manage_supervisor', hodviews.Manage_Supervisor, name="manage_supervisor"),
    path('edit_supervisor/<str:supervisor_id>', hodviews.Edit_Supervisor, name="edit_supervisor"),
    path('edit_supervisor_save', hodviews.Edit_Supervisor_Save, name="edit_supervisor_save"),
    path('addstudent', hodviews.AddStudent, name="addstudent"),
    path('add_student_save', hodviews.Add_Student_Save, name="add_student_save"),
    path('manage_student', hodviews.Manage_Student, name="manage_student"),
    path('edit_student/<str:student_id>', hodviews.Edit_Student, name="edit_student"),
    path('edit_student_save', hodviews.Edit_Student_Save, name="edit_student_save"),
    path('adddepartment', hodviews.AddDepartment, name="adddepartment"),
    path('department_save', hodviews.Department_Save, name="department_save"),
    path('manage_department', hodviews.Manage_Department, name="manage_department"),
    path('edit_department/<str:department_id>', hodviews.Edit_Department, name="edit_department"),
    path('edit_department_save', hodviews.Edit_Department_Save, name="edit_department_save"),
    path('currentfyp', hodviews.CurrentFyp, name="currentfyp"),
    path('add_fyp', hodviews.Add_Fyp, name="add_fyp"),
    path('fyp_save', hodviews.Fyp_Save, name="fyp_save"),
    path('select_supervisor', hodviews.Select_Supervisor, name="select_supervisor"),
    path('pastfyp', hodviews.PastFyp, name="pastfyp"),
]
```

Figure 68 url patterns

```
from django.contrib.auth import logout
from django.core.files.storage import FileSystemStorage
from django.http import HttpResponseRedirect, HttpResponseRedirect
from django.shortcuts import render

from fyp_repository_app.models import Departments, CustomUser, Admindep, Supervisor, Student_Projects

def BaseClass(request):
    return render(request, "hod_templates/base_class.html")

def AddAdmin(request):
    departments=Departments.objects.all()
    return render(request, "hod_templates/addadmin.html", {"departments":departments})

def Admin_Save(request):
    if request.method != "POST":
        return HttpResponse("Method Not Allowed")
    else:
        first_name = request.POST.get("first_name")
        last_name = request.POST.get("last_name")
        username = request.POST.get("username")
        email = request.POST.get("email")
        password = request.POST.get("password")
        address = request.POST.get("address")
        gender = request.POST.get("gender")
        office_loc = request.POST.get("office_loc")
        profile_pic = request.FILES['profile_pic']
        fs = FileSystemStorage()
        filename = fs.save(profile_pic.name, profile_pic)
```

Figure 69 base class

```

def AddDepartment(request):
    return render(request, "hod_templates/add_department.html")

def Department_Save(request):
    if request.method!="POST":
        return HttpResponse("Method Not Allowed")
    else:
        try:
            dep_id = request.POST.get("department_id")
            department_model=Departments(name=dep_id)
            department_model.save()
            return HttpResponseRedirect("/adddepartment")
        except:
            return HttpResponse("Failed to Add Department")

def Manage_Department(request):
    departments=Departments.objects.all()
    return render(request,"hod_templates/manage_department.html",{ "departments":departments})

def Edit_Department(request,department_id):
    department=Departments.objects.get(id=department_id)
    return render(request, "hod_templates/edit_department.html", {"department": department})

```

Figure 70 department

```

filename = fs.save(profile_pic.name, profile_pic)
profile_pic_url = fs.url(filename)
dep_id = request.POST.get("dep_id")
try:
    user = CustomUser.objects.create_user(username=username, email=email, password=password,
                                           last_name=last_name, first_name=first_name, user_type=3)

    user.admindep.address = address
    user.admindep.gender = gender
    user.admindep.office_location = office_loc
    user.admindep.profile_pic = profile_pic_url
    dep_obj = Departments.objects.get(id=dep_id)
    user.admindep.dep_id = dep_obj
    user.save()
    return HttpResponseRedirect("/add_admin")

except:
    return HttpResponse("Failed to add Admin")

def Manage_Admin(request):
    admins=Admindep.objects.all()
    return render(request, "hod_templates/manage_admin.html",{ "adminss":admins})

def Edit_Admin(request,admin_id):
    departments=Departments.objects.all()
    admins=Admindep.objects.get(admin=admin_id)
    return render(request, "hod_templates/edit_admin.html", {"admins": admins, "departments":departments})

```

Figure 71 manage admin

```
def Edit_Department_Save(request):  
    if request.method != "POST":  
        return HttpResponse("Method Not Allowed")  
    else:  
        dep_id = request.POST.get("dep_id")  
        dep_name = request.POST.get("dep_name")  
        try:  
            departments=Departments.objects.get(id=dep_id)  
            departments.name=dep_name  
            departments.save()  
            return HttpResponseRedirect("/manage_department")  
        except:  
            return HttpResponse("Failed to update Department")
```

Figure 72 edit department

Chapter 5

Software Testing

Software Testing is the most crucial part of the Software Development Process. It is the investigation or evaluation of a software component, improving them, and finding bugs and defects. Testing is usually done by executing a system in such a way that it identifies any gaps, errors, or missing requirements in contrary to the actual requirements.

5.1 Testing Methodology

It is essential to have a testing plan in place to ensure that the product delivered is robust and stable, and is delivered on a predictable timeline.

Unit testing is performed by utilizing the black box testing strategy. Unit testing is the primary level computer program testing and is performed earlier to integration testing. It is normally performed by software developers themselves. Codes are more re-usable. In order to make unit testing possible, codes need to be modular. It means that code is easy to re-use. Unit testing increases confidence in maintaining code. If unit tests are written very well and if they are run every time any code is changed, we will be able to catch any defects introduced due to change.

5.2 Test Cases

Test case is a detail of the inputs, execution conditions, testing method, and expected results that define a single test to be executed to realize a specific program testing objective, such as to work out a specific program way or to comply with a particular requirement.

5.2.1 Test Case 1

Test case description

There is an option for user to get sign in to the system. For this user has to enter username, and password.

Expected result of the test case

The expected result of the test case that when user will be successfully login.

Actual result of the test case

Actual result of test case is passed.

Table 21 user successful login

Date: 29 July 2022	Test ID:1
System: FYP Library	Test Type: Unit testing
Objective: Sign in user	
Version:1	
Input: Sign in credentials(username, and password)	
Expected Result: User will be successfully login.	
Actual Result: passed	

5.2.2 Test Case 2

Test case description

There is an option for user to get sign in to the system. For this user has to enter username, and password.

Expected result of the test case

The expected result of the test case that when user will not be successfully login.

Actual result of the test case

Actual result of test case is failed login.

Table 22 failed user login

Date: 29 July 2022	Test ID:2
System: FYP Library	Test Type: Unit testing
Objective: Sign in user	
Version:2	
Input: Sign in credentials(username, and password)	

<i>Expected Result:</i> User will not be successfully login.
<i>Actual Result:</i> cannot login

5.2.3 Test Case 3

Test case description

There is an option for user to get sign in to the system. After a successful login super admin can add admin.

Expected result of the test case

The expected result of the test case that when super admin add admin they will be successfully added.

Actual result of the test case

Actual result of test case is passed.

Table 23 add admin

Date: 29 July 2022	Test ID:3
System: FYP Library	Test Type: Unit testing
Objective: add admin	
Version:1	
Input: add login information of admin.	
Expected Result: User will be successfully added.	
Actual Result: passed.	

5.2.4 Test Case 4

Test case description

There is an option for user to get sign in to the system. After a successful login super admin can add admin.

Expected result of the test case

The expected result of the test case that when super admin add admin, admin won't be added successfully added.

Actual result of the test case

Actual result of test case is admin already exists.

Table 24 add Admin

Date: 29 July 2022	Test ID:4
System: FYP Library	Test Type: Unit testing
Objective: add admin	
Version:2	
Input: add login information of admin.	
Expected Result: User cannot be added.	
Actual Result: passed.	

5.2.5 Test Case 5

Test case description

There is an option for user to get sign in to the system. After a successful login super admin can add project.

Expected result of the test case

The expected result of the test case that when super admin add project it will be successfully added.

Actual result of the test case

Actual result of test case is passed.

Table 25 add project

Date: 29 July 2022	Test ID:5
System: FYP Library	Test Type: Unit testing
Objective: add project	
Version:1	
Input: add project	
Expected Result: project will be successfully added.	
Actual Result: passed.	

5.2.6 Test Case 6

Test case description

There is an option for user to get sign in to the system. After a successful login super admin cannot add project.

Expected result of the test case

The expected result of the test case that when super admin add project it will not be successfully added.

Actual result of the test case

Actual result of test case is failed.

Table 26 add project

Date: 1 June 2022	Test ID:5
System: FYP Library	Test Type: Unit testing
Objective: add project	

<i>Version:</i> 2
<i>Input:</i> add project
<i>Expected Result:</i> project cannot be added.
<i>Actual Result:</i> failed

