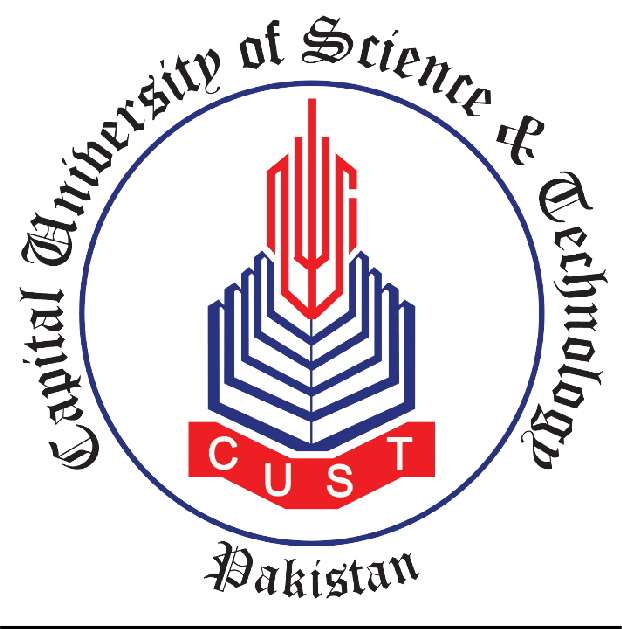
**FYP Library**

Mahnoor. BCS191012

M. Kamran BCS191037

M. Asim Amir BCS191022



**Spring-2022 Supervised By**

**Sir Hafiz Anas Bilal**

**Department of Computer Science**

**Capital University of Science & Technology, Islamabad**

Submission from the Final-Year

**Supervisor’s Signature**

PROJECT REPORT

|  |  |
| --- | --- |
|  | V 1.0 |

|  |  |
| --- | --- |
|  | 03 |

|  |  |
| --- | --- |
|  | FYP Library |

|  |  |
| --- | --- |
|  | Sir Hafiz Anas Bilal |

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Mahnoor** | BCS191012 | [Mahnoorkhan0468@gmail.com](mailto:Mahnoorkhan0468@gmail.com) |
| **M. Kamran** | BCS191037 | [Kamran11106@gmail.com](mailto:Kamran11106@gmail.com) |
| **M. Asim Amir** | BCS191022 | [Malixxasim111@gmail.com](mailto:Malixxasim111@gmail.com) |

Table of content

[Chapter 1 9](#_Toc117117284)

[1.1 Introduction 9](#_Toc117117285)

[1.2 Problem statement 9](#_Toc117117286)

[1.3 Aim of the Project 9](#_Toc117117287)

[1.4 FYP Library 10](#_Toc117117288)

[1.5 Business Scope 11](#_Toc117117289)

[1.6 Existing example 11](#_Toc117117290)

[1.7 Useful Tools and Technologies 11](#_Toc117117291)

[1.8 Work Breakdown Structure 12](#_Toc117117292)

[1.9 Project Timeline 13](#_Toc117117293)

[Chapter 2 14](#_Toc117117294)

[Requirement Specification and Analysis 14](#_Toc117117295)

[2.1 Requirement Specification 14](#_Toc117117296)

[2.1.1 Functional Requirements 14](#_Toc117117297)

[2.1.2 Non-Functional Requirements 15](#_Toc117117298)

[2.2 System Use case Modeling 16](#_Toc117117299)

[2.2.1 Use case diagram of super admin 16](#_Toc117117300)

[2.2.2 Use case diagram of admin 17](#_Toc117117301)

[2.2.3 Use case diagram of student 18](#_Toc117117302)

[2.3 Use cases Descriptions 19](#_Toc117117303)

[2.3.1 Add Admin 19](#_Toc117117304)

[2.3.2 Search admin 20](#_Toc117117305)

[2.3.3 Delete admin 21](#_Toc117117306)

[2.3.4 Update admin 22](#_Toc117117307)

[2.3.5 List of admin 23](#_Toc117117308)

[2.3.6 Generate Accounts 24](#_Toc117117309)

[2.3.7 Login 25](#_Toc117117310)

[2.3.8 View the list of current semester Final Year Projects. 25](#_Toc117117311)

[2.3.9 Admin can add projects 26](#_Toc117117312)

[2.3.10 Search projects 27](#_Toc117117313)

[2.3.11 Delete projects. 28](#_Toc117117314)

[2.3.12 Modify project 29](#_Toc117117315)

[2.3.13 Student can view list of projects 30](#_Toc117117316)

[2.3.14 Add own project information 32](#_Toc117117317)

[2.3.15 Upload Project Report 33](#_Toc117117318)

[2.3.16 Logout 34](#_Toc117117319)

[2.4 System sequence diagram 35](#_Toc117117320)

[2.5 Domain Model 43](#_Toc117117321)

[Chapter 3 44](#_Toc117117322)

[System design 44](#_Toc117117323)

[3.1 Software Architecture 44](#_Toc117117324)

[3.2 Class Diagram 45](#_Toc117117325)

[3.3 Sequence Diagram 46](#_Toc117117326)

[3.4 Database schema 56](#_Toc117117327)

[3.5 User Interface Design 57](#_Toc117117328)

[Chapter 4 61](#_Toc117117329)

[Software Development 61](#_Toc117117330)

[4.1 Coding Standards 61](#_Toc117117331)

[4.2 Development Environment 61](#_Toc117117332)

[4.2.1 Visual Studio code 61](#_Toc117117333)

[4.2.2 Apache MYSQL 61](#_Toc117117334)

[4.3 Code Snippet 61](#_Toc117117335)

[Chapter 5 65](#_Toc117117336)

[Software Testing 65](#_Toc117117337)

[5.1 Testing Methodology 65](#_Toc117117338)

[5.2 Test Cases 65](#_Toc117117339)

[5.3.1 Test Case 1 65](#_Toc117117340)

[Test case description 65](#_Toc117117341)

[Expected result of the test case 65](#_Toc117117342)

[Actual result of the test case 65](#_Toc117117343)

[5.3.2 Test Case 2 66](#_Toc117117344)

[Test case description 66](#_Toc117117345)

[Expected result of the test case 66](#_Toc117117346)

[Actual result of the test case 66](#_Toc117117347)

[5.3.3 Test Case 3 67](#_Toc117117348)

[Test case description 67](#_Toc117117349)

[Expected result of the test case 67](#_Toc117117350)

[Actual result of the test case 67](#_Toc117117351)

**List of Figures**

[Figure 1 Work breakdown structure 12](#_bookmark12)

[Figure 2 Project timeline 13](#_bookmark14)

[Figure 3 Use case diagram of super admin 16](#_bookmark24)

[Figure 4 Use case of admin 17](#_bookmark26)

[Figure 5 Use case diagram of student 18](#_bookmark28)

[Figure 6 Super admin login 35](#_bookmark63)

[Figure 7 Add admin 35](#_bookmark64)

[Figure 8 Delete admin 35](#_bookmark65)

[Figure 9 Update admin information 36](#_bookmark66)

[Figure 10 Search admin 36](#_bookmark67)

[Figure 11 list of admin 36](#_bookmark68)

[Figure 12 Super admin logout 37](#_bookmark69)

[Figure 13 Admin login 37](#_bookmark70)

[Figure 14 Current project 37](#_bookmark71)

[Figure 15 Past projects 38](#_bookmark72)

[Figure 16 Add project 38](#_bookmark73)

[Figure 17 Search projects 38](#_bookmark74)

[Figure 18 Delete projects 39](#_bookmark75)

[Figure 19 Update project information 39](#_bookmark76)

[Figure 20 Admin logout 39](#_bookmark77)

[Figure 21 Student login 40](#_bookmark78)

[Figure 22 Search project 40](#_bookmark79)

[Figure 23 Project list 40](#_bookmark80)

[Figure 24 Current project list 41](#_bookmark81)

[Figure 25 Upload report 41](#_bookmark82)

[Figure 26 upload project information 41](#_bookmark83)

[Figure 27 student logout 41](#_bookmark84)

[Figure 28 Domain model 42](#_bookmark86)

[Figure 29 Software Architecture 43](#_bookmark90)

[Figure 30 Class diagram 44](#_bookmark92)

[Figure 31 Super admin login 45](#_bookmark94)

[Figure 32 Add admin 45](#_bookmark95)

[Figure 33 Search admin 46](#_bookmark96)

[Figure 34 Update admin 46](#_bookmark97)

[Figure 35 Delete admin 47](#_bookmark98)

[Figure 36 list of admin 47](#_bookmark99)

[Figure 37 Super Admin logout 47](#_bookmark100)

[Figure 38 Admin login 48](#_bookmark101)

[Figure 39 Request list of project 48](#_bookmark102)

[Figure 40 Search project 49](#_bookmark103)

[Figure 41 Add project 49](#_bookmark104)

[Figure 42 Delete project 50](#_bookmark105)

[Figure 43 Update project 50](#_bookmark106)

[Figure 44 Admin logout 51](#_bookmark107)

[Figure 45 Student login 51](#_bookmark108)

[Figure 46 Request list of projects 52](#_bookmark109)

[Figure 47 Search project 52](#_bookmark110)

[Figure 48 Upload project information 52](#_bookmark111)

[Figure 49 Upload report to check plagiarism 53](#_bookmark112)

[Figure 50 Student logout 53](#_bookmark113)

[Figure 51 Database Schema 54](#_bookmark115)

[Figure 52 Sign in GUI 55](#_bookmark117)

[Figure 53 List of projects 56](#_bookmark118)

[Figure 54 CRUD operation 56](#_bookmark119)

[Figure 55 Add project 57](#_bookmark120)

[Figure 56 past projects 57](#_bookmark121)

[Figure 57 view report 58](#_bookmark122)

[Figure 58 upload report for plagiarism checking 58](#_bookmark123)

[Figure 59 user login and logout 60](#_bookmark131)

[Figure 60 admin CRUD operation 60](#_bookmark132)

[Figure 61 add Fyp Project 61](#_bookmark133)

[Figure 62 super admin successful login 61](#_bookmark134)

[Figure 63 add admin 62](#_bookmark135)

[Figure 64 URLs.py 62](#_bookmark136)

[Figure 65 Database link 62](#_bookmark137)

**List of Tables**

[Table 1 Functional requirements 14](#_bookmark19)

[Table 2 Non-Functional requirements 16](#_bookmark21)

[Table 3 Add new admin 19](#_bookmark31)

[Table 4 Search admin 20](#_bookmark33)

[Table 5 Delete admin 21](#_bookmark35)

[Table 6 Update admin 22](#_bookmark37)

[Table 7 List of admin 23](#_bookmark39)

[Table 8 Admin can generate accounts 24](#_bookmark41)

[Table 9 Login to the system 25](#_bookmark43)

[Table 10 View current semester projects 26](#_bookmark45)

[Table 11 Admin can add projects 27](#_bookmark47)

[Table 12 Search projects by title 28](#_bookmark49)

[Table 13 Delete projects 28](#_bookmark51)

[Table 14 Modify project 29](#_bookmark53)

[Table 15 can view all projects 30](#_bookmark55)

[Table 16 Add own project information 31](#_bookmark57)

[Table 17 Upload project report 32](#_bookmark59)

[Table 18 Logout to the system 33](#_bookmark61)

[Table 19 user successful login 64](#_bookmark143)

[Table 20 failed user login 64](#_bookmark145)

[Table 21 Add Admin 65](#_bookmark147)

# Chapter 1

# 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.

# 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.

# 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.

# 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. Features
* Account Creation.
* Login.
* FYP Repository.
* Searching of projects.
* Plagiarism checking on projects.
  + 1. Super Admin
* Will generate accounts.
* Can login\logout.
* Can view the profile.
* Will be able to add department admin, supervisor, student, project and department.
* Will be able to view the information of department admin, supervisor, student, project and department
* Will be able to search the information of department admin, supervisor, student, project and department.
* Will be able to modify the information of department admin, supervisor, student, project and department.
  + 1. Admin `
* Will generate accounts of supervisor, student.
* Can login\logout.
* Can view the profile.
* Will be able to add supervisor, student, and project.
* Will be able to view the information supervisor, student and project.
* Will be able to search the information of supervisor, student, project.
* Will be able to modify the information of supervisor, student, project.

* + 1. Students
* Will be able to login.
* Will be able to logout.
* Will be able to view profile.
* Will be able to add Final year project.
* Will be able to view the information of past Final Year Projects.
* Will be able to search the information of project.
* Will be able to upload report.
* Will be able to check plagiarism on report.

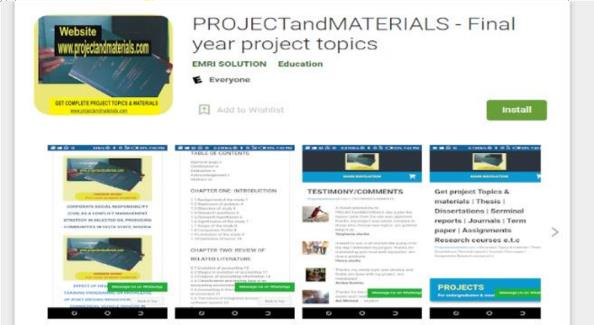
* + 1. Supervisors
* Will be able to login.
* Will be able to logout.
* Will be able to view profile.
* Will be able to add Final year project.
* Will be able to view the information of past Final Year Projects.
* Will be able to search the information of project.
* Will be able to upload report.
* Will be able to check plagiarism on report.

# 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.

# Existing example



Projectandmaterails.com is an Online Academic Library, the “NO: 1” Ultimate source to your Research work which includes: Project Topics and Materials

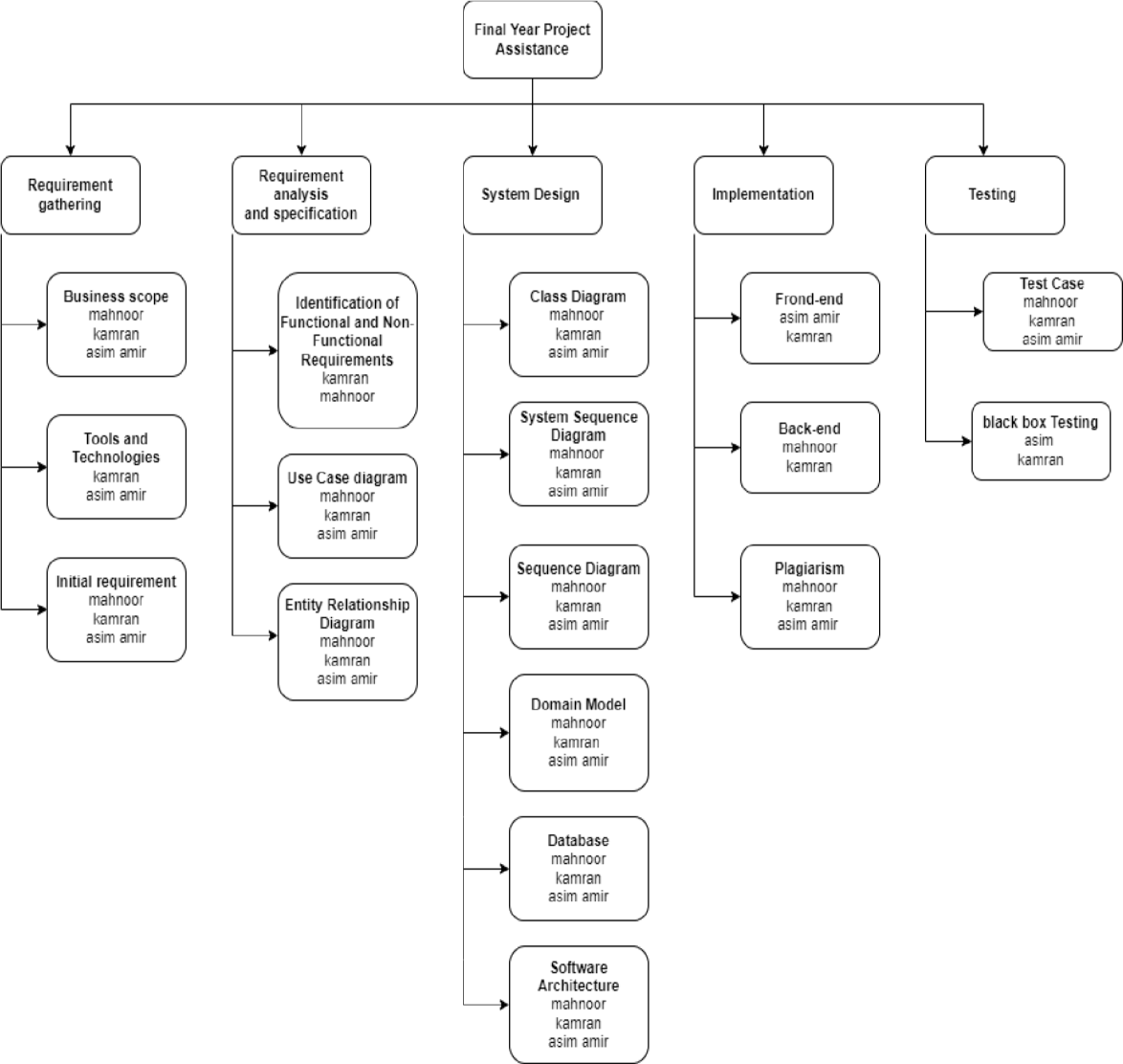
# Useful Tools and Technologies

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**

# Work Breakdown Structure

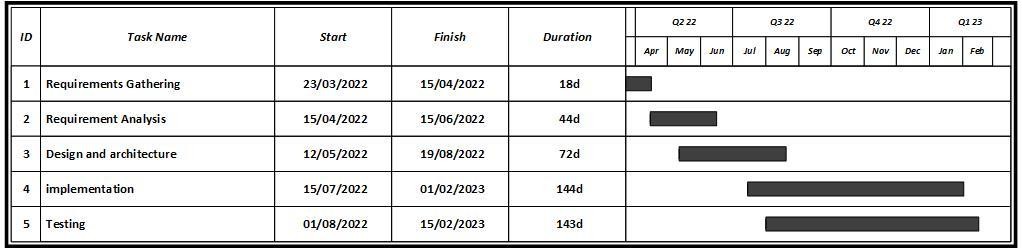
The figure below shows the work breakdown structure of the complete project. It involves the activities which are required to complete the project.



*Figure 1 Work breakdown structure*

# Project Timeline

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



*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.

# 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.

## 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 | Approved |
| **2.** | Super admin can logout | core | Approved |
| **3.** | Super admin can assign credentials to department admin, supervisor and student. | core | Approved |

|  |  |  |  |
| --- | --- | --- | --- |
| **4.** | Super admin can perform CRUD operations on admin, supervisor and student. | core | Approved |
| **5.** | Super admin can view the information of department admin, supervisor and student. | core | Approved |
|  | Super Admin can check plagiarism on reports. |  |  |
| **6.** | Admin can login. | core | Approved |
| **7.** | Admin can logout. | core | Approved |
| **8.** | Admin can generate students and supervisor accounts. | core | Approved |
| **9.** | Admin can view the information of supervisor and student and projects. | core | Pending |
|  | Admin can perform CRUD operations on supervisor , student and projects. |  |  |
| **10.** | Admin can view profile. | optional | Pending |
| **11.** | Admin can check plagiarism on reports. | Core | Pending |
| **13.** | Student can login. | Core | Pending |
| **14.** | Student can logout. | Core | Pending |
| **15.** | Student can view profile. | Core | Pending |
| **16.** | Student can view the information of Past Final year projects. | Core | Pending |
| **17.** | Student can add own information of Final Year Project. | Core | Pending |
| **18.** | Student can generate plagiarism report. | Core | Pending |
| **19.** | Supervisor can login. | Core | Pending |
| **20.** | Supervisor can logout. | Core | Pending |
|  | Supervisor can view profile. |  |  |
|  | Supervisor can add Final Year Project. |  |  |
|  | Supervisor can view the information of past Final Year Projects. |  |  |
|  | Supervisor can search the information of projects. |  |  |
|  | Supervisor can upload report. |  |  |
|  | Supervisor can check plagiarism on report. |  |  |

## 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:

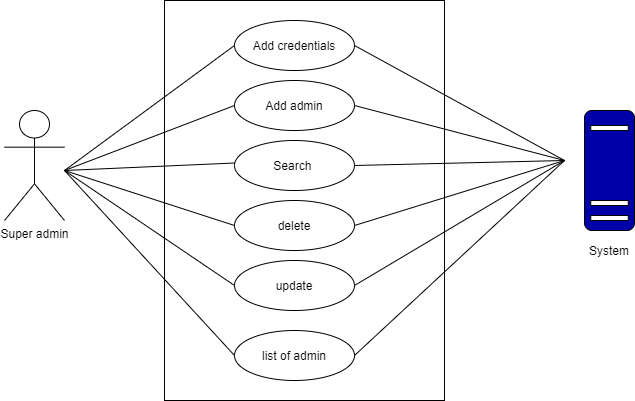
*Table 2 Non-Functional requirements*

|  |  |  |
| --- | --- | --- |
| **Sr.** | **Non-Functional Requirements** | **Category** |
| **1.** | Interactive UI as per guidelines. | Learnability |
| **2.** | Correct records retrieval with user id. | Reliability |
| **3.** | Real-time data display for accurate results. | Accuracy |
| **4.** | Applications alerts on unplanned activity. | Security |

# System Use case Modeling

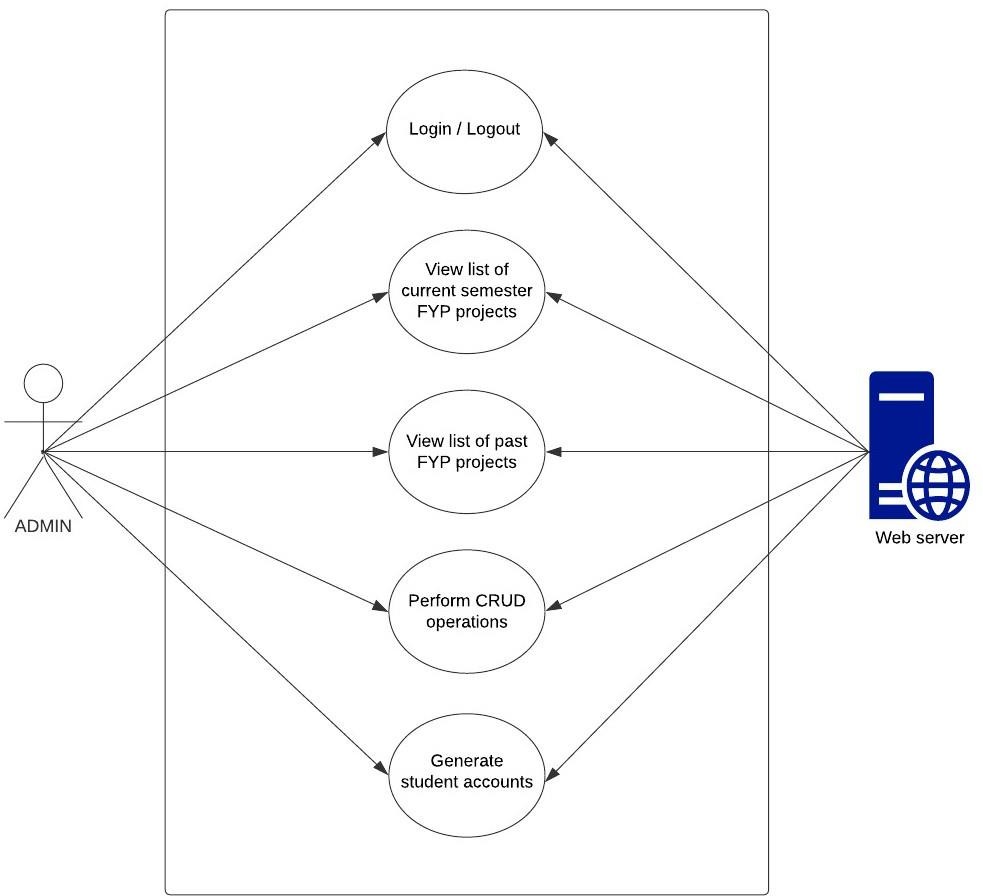
Use Case Diagram of the overall system in which graphic depiction of the interactions among the elements of a system and the relationships between and among the actors. We show use cases with respect to particular actors related with actions.

## Use case diagram of super admin



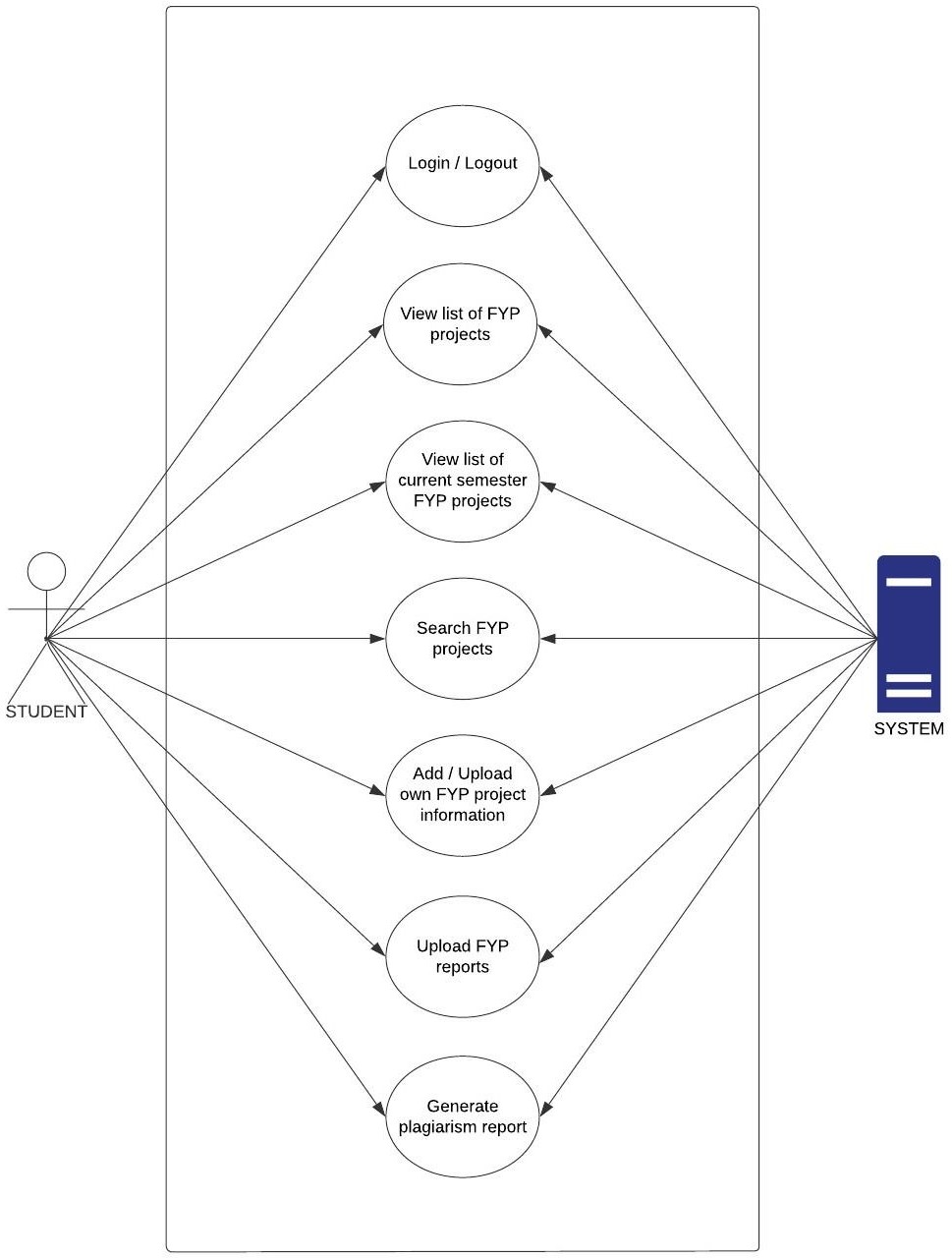
*Figure 3 Use case diagram of super admin*

## Use case diagram of admin



*Figure 4 Use case of admin*

## Use case diagram of student



*Figure 5 Use case diagram of student*

# 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.

## Add 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 3 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.  6. Super admin will add admin information.  7. Enter submit button. | | System  2. Validate login credentials.  4. Open Portal Homepage.  8. System add admin in database. | | |
| **Alternative Flows:** | | Admin already exists. | | | | |
| **Exceptions:** | | 1. Internet connection is not available. 2. Database is not responding. | | | | |

## 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 4 Search admin*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 02 | | | | | |
| **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 login credentials.  3. Successful Login.  5. Click on Search button.  6. Search by name/ id and press enter. | | System  2. Validate login credentials.  4. Open Portal Homepage.  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 | | | | |

## Delete admin

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

*Table 5 Delete admin*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 03 | | | | |
| **Use Case Name:** | Delete admin | | | | |
| **Created By:** | kamran | | **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 delete an admin. | | | |
| **Trigger:** | | Click on delete button. | | | |
| **Preconditions:** | | Admin must be login into the system. | | | |
| **Post conditions:** | | Admin will be deleted. | | | |
| **Normal Flow:** | | Super admin  1. super admin must enter login credentials.  3. Successful Login.  5. Click on delete button. | | System  2. Validate login credentials.  4. Open Portal Homepage.  6. System display delete status. | |
| **Alternative Flows:** | | N/A | | | |
| **Exceptions:** | | 1. Internet connection is not available. 2. Database is not responding. 3. Admin doesn’t exists. | | | |

## 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 6 Update admin*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 04 | | | | | |
| **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. Super admin must be logged in. 2. Admin must exist. | | | | |
| **Post conditions:** | | Admin information must be updated. | | | | |
| **Normal Flow:** | | Super admin  1. Super admin must enter login credentials.  3. Successful Login.  5. Click on update button.  7. Enter updated information.  8. Press enter. | | System  2. Validate login credentials.  4. Open Portal Homepage.  6. Show update form.  8.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. | | | | |

## 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 7 List of admin*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 05 | | | | | |
| **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.  3. Successful Login.  5. Click on list of admin button. | | System  2. Validate login credentials.  4. Open Portal Homepage.  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. | | | | |

## 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 8 Admin can generate accounts*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 06 | | | | |
| **Use Case Name:** | 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:** | | Admin | | | |
| **Description:** | | Admin can generate students’ accounts which are not registered yet. | | | |
| **Trigger:** | | Click on the generate accounts button. | | | |
| **Preconditions:** | | Admin must login to the system. | | | |
| **Post conditions:** | | Accounts generated Successfully. | | | |
| **Normal Flow:** | | Admin  1. Admin must enter login credentials.  3. Successful Login.  5. Admin press on generate account button. | | System  2. Validate login credentials.  4. Open Portal Homepage.  6. Fetch data from server and generate accounts. | |
| **Alternative Flows:** | | NA | | | |
| **Exceptions:** | | 1. Internet connection is not available. 2. Database is not responding. 3. Accounts already exists. | | | |

## 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 9 Login to the system*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 07 | | | | |
| **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:** | | Super admin, admin, student  1. User must enter 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. | | | |

## View the list of current semester Final Year Projects.

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. After that they will see the current semester project button on the open portal homepage through which they will see the list of current semester final year projects as shown:

*Table 10 View current semester projects*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 08 | | | | | |
| **Use Case Name:** | View the list of current semester Final Year Projects. | | | | | |
| **Created By:** | M. Asim Amir | | **Last Updated By:** | | N/A | |
| **Date Created:** | 2/4/2022 | | **Last Revision Date:** | | | 4/4/2022 |
| **Actors:** | | Admin, Student | | | | |
| **Description:** | | Users will be able to view the list of current semester Final Year projects from their accounts. | | | | |
| **Trigger:** | | Click on view current semester projects button. | | | | |
| **Preconditions:** | | User must login to the system. | | | | |
| **Post conditions:** | | User must view the list of current semester Final Year Projects. | | | | |
| **Normal Flow:** | | Admin, student  1. User must enter 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:** | | 1. Projects are not registered. 2. System will display empty fields. | | | | |
| **Exceptions:** | | 1. Internet connection is not available. 2. Database is not responding. | | | | |

## Admin can 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 11 Admin can add projects*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 09 | | | | | |
| **Use Case Name:** | Admin 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:** | | Admin. | | | | |
| **Description:** | | Admin can add projects for students whose records are not in the database. | | | | |
| **Trigger:** | | Click on Add project button. | | | | |
| **Preconditions:** | | Admin must login to the system. | | | | |
| **Post conditions:** | | Project added successfully. | | | | |
| **Normal Flow:** | | Admin  1. Admin must enter login credentials.  3. Successful Login.  5. Click on Add project.  6. Add project information and press enter. | | System  2. Validate login credentials.  4. Open Portal Homepage.  7. System add project in database. | | |
| **Alternative Flows:** | | Project already exists. | | | | |
| **Exceptions:** | | 1. Internet connection is not available. 2. Database is not responding. | | | | |

## 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 12 Search projects by title*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 10 | | | | | |
| **Use Case Name:** | Search project by title | | | | | |
| **Created By:** | M. Kamran | | **Last Updated By:** | | N/A | |
| **Date Created:** | 2/4/2022 | | **Last Revision Date:** | | | 4/4/2022 |
| **Actors:** | | 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:** | | Admin, student  1. User must login credentials.  3. Successful Login.  5. Click on Search button.  6. Enter project name.  7. Press enter. | | System  2. Validate login credentials.  4. Open Portal Homepage.  8. System will show the project. | | |
| **Alternative Flows:** | | NA | | | | |
| **Exceptions:** | | 1. Internet connection is not available. 2. Database is not responding. 3. Project is not available. | | | | |

## Delete projects.

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 delete through which admin can delete projects and system will display the delete projects status as shown below:

*Table 13 Delete projects*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 11 | | | | |
| **Use Case Name:** | Delete projects | | | | |
| **Created By:** | M. Asim Amir | | **Last Updated By:** | | N/A |
| **Date Created:** | 2/4/2022 | | **Last Revision Date:** | | 4/4/2022 |
| **Actors:** | | Admin | | | |
| **Description:** | | Admin will be able to delete project information. | | | |
| **Trigger:** | | Click on delete button. | | | |
| **Preconditions:** | | Admin must login to the system. | | | |
| **Post conditions:** | | Must delete project information. | | | |
| **Normal Flow:** | | Admin  1. Admin must enter login credentials.  3. Successful Login.  5. Click on delete button. | | System  2. Validate login credentials.  4. Open Portal Homepage.  6. System display delete status. | |
| **Alternative Flows:** | | Project does not exists. | | | |
| **Exceptions:** | | 1. Internet connection is not available. 2. Database is not responding. | | | |

## 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 14 Modify project*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 12 | | | | | |
| **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:** | | Admin | | | | |
| **Description:** | | Admin will be able to update project information. | | | | |
| **Trigger:** | | Click on update button. | | | | |
| **Preconditions:** | | 1. Admin must be logged in. 2. Project must exist. | | | | |
| **Post conditions:** | | Must updated information. | | | | |
| **Normal Flow:** | | Admin  1. Admin must enter login credentials.  3. Successful Login.  5. Click on update button.  7. Update information of project and press enter. | | System  2. Validate login credentials.  4. Open Portal Homepage.   1. Show update form page. 2. System will store updated   Information in the database. | | |
| **Alternative Flows:** | | NA | | | | |
| **Exceptions:** | | 1. Internet connection is not available. 2. Database is not responding. 3. Project does not exist. | | | | |

## Student can 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 15 can view all projects*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 13 | | | | |
| **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:** | | Admin, Student. | | | |
| **Description:** | | User will be able to view the list of past Final Year projects. | | | |
| **Trigger:** | | Click on all projects button. | | | |
| **Preconditions:** | | User must be logged in. | | | |
| **Post conditions:** | | Must view details of projects. | | | |
| **Normal Flow:** | | Admin, student  1. User must enter 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. | | | |

## Add own project information

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 they can add project information by clicking on the button of add projects and after that system will get project title and check plagiarism on title, department, supervisor name, group members, tools, and store that in database as shown below:

*Table 16 Add own project information*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 14 | | | | |
| **Use Case Name:** | Add project information. | | | | |
| **Created By:** | M. Kamran | | **Last Updated By:** | | N/A |
| **Date Created:** | 4/4/2022 | | **Last Revision Date:** | | 5/4/2022 |
| **Actors:** | | Student. | | | |
| **Description:** | | Student can add details/information of own project. | | | |
| **Trigger:** | | Click on add project information button. | | | |
| **Preconditions:** | | Student must be logged in. | | | |
| **Post conditions:** | | Display projects information. | | | |
| **Normal Flow:** | | Student  1. Student must enter login credentials.  3. Successful Login.  5. Click on add project information button.  7. Enter project information  8. Press enter | | System  2. Validate login credentials.  4. Open Portal Homepage.  6. Show add form page.  9. System store project information in the database. | |
| **Alternative Flows:** | | Show Error message that the project already exists. | | | |
| **Exceptions:** | | 1. Internet connection is not available. 2. Database is not responding. | | | |

## 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 if the plagiarism will be less than 20% report will be uploaded in database otherwise shoe error and as shown:

*Table 17 Upload project report*

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | 15 | | |
| **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:** | | Student  1. Student must enter login credentials.  3. Successful Login.  5. Click on Report Upload button.  7. Upload file and press enter. | | System  2. Validate login credentials.  4. Open Portal Homepage.   1. Show dialog box to upload file.   8. System will check plagiarism on file. | |
| **Alternative Flows:** | | Show Error message that File cannot uploaded. | | | |
| **Exceptions:** | | 1. Internet connection is not available. 2. Database is not responding. | | | |

## Logout

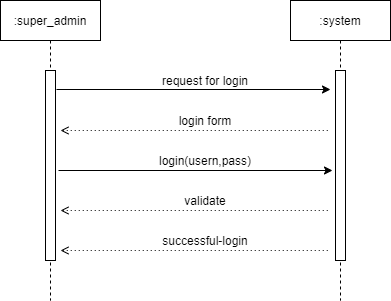
Here is the Use Case in which super admin, admin and student can log out by clicking on log out button and system will allow them to log out of the system as shown:

*Table 18 Logout to the system*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | 16 | | | | |
| **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:** | | Super admin, admin, student  1. User will click on logout button to request for logging out.  3. Successfully log out. | | System  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. | | | |

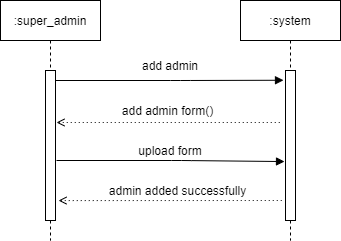
# System sequence diagram

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.



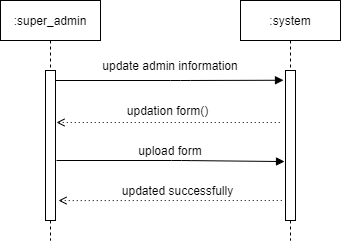
*Figure 7 Add admin*

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



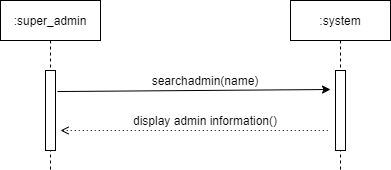
*Figure 8 Delete admin*

Super admin has also the authority to delete an admin. If the admin leaves super admin can delete that admin.



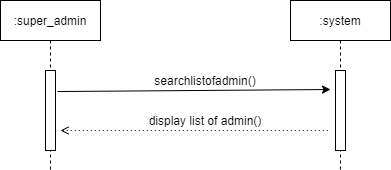
*Figure 9 Update admin information*

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.



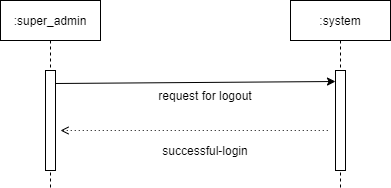
*Figure 10 Search admin*

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.



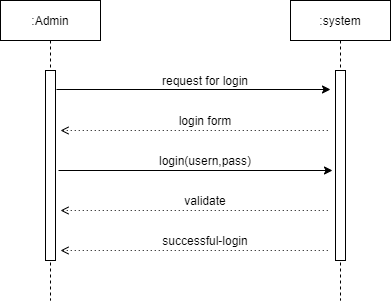
*Figure 11 list of admin*

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



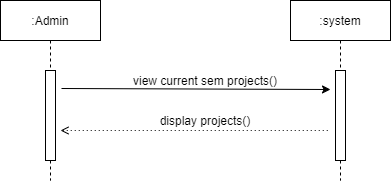
*Figure 12 Super admin logout*

Super admin can request to logout the system.



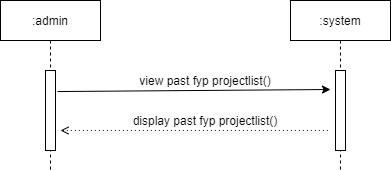
*Figure 13 Admin login*

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.



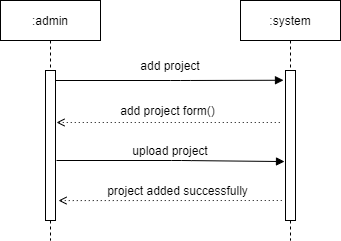
*Figure 14 Current project*

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



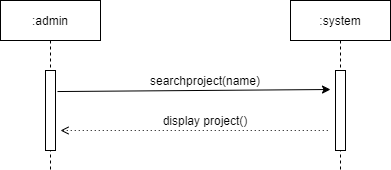
*Figure 15 Past projects*

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



*Figure 16 Add project*

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



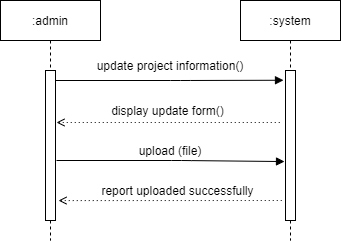
*Figure 17 Search projects*

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.



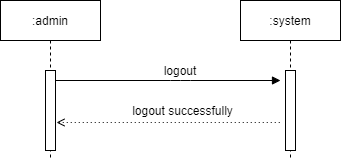
*Figure 18 Delete projects*

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



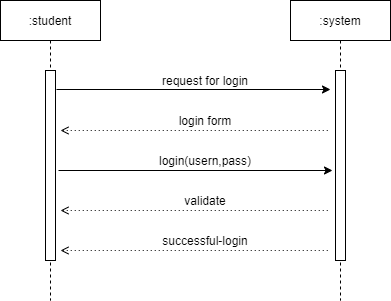
*Figure 19 Update project information*

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.



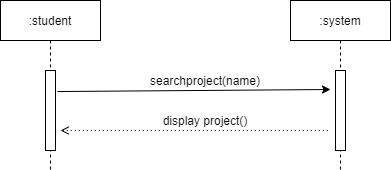
*Figure 20 Admin logout*

Admin can logout from the system.



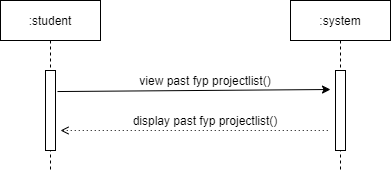
*Figure 21 Student login*

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.



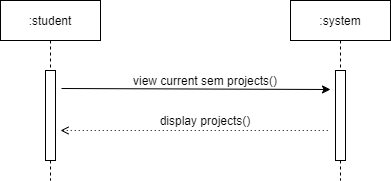
*Figure 22 Search project*

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 23 Project list*

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



*Figure 24 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.



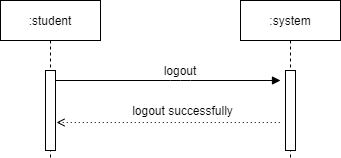
*Figure 25 Upload report*

Student can upload the report for plagiarism check.



*Figure 26 upload project information*

Student can upload the information about the project.

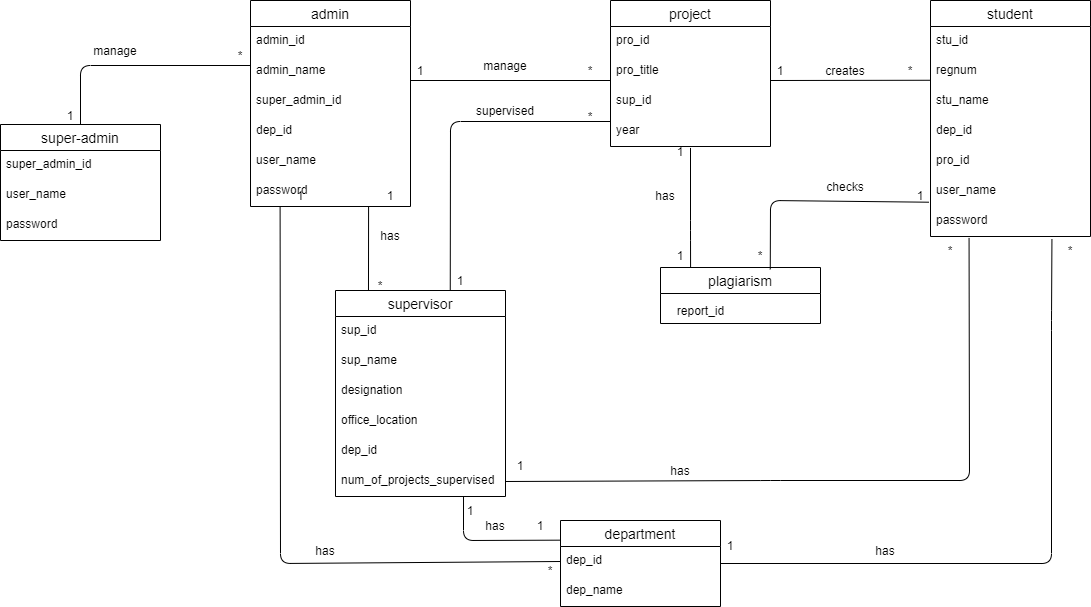


*Figure 27 student logout*

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 28 Domain model*

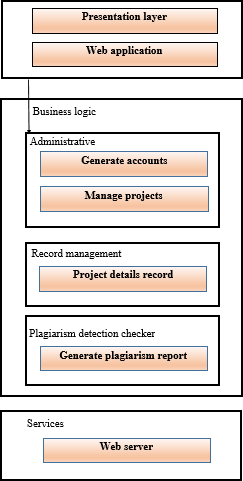
# Chapter 3

# 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.

# Software Architecture

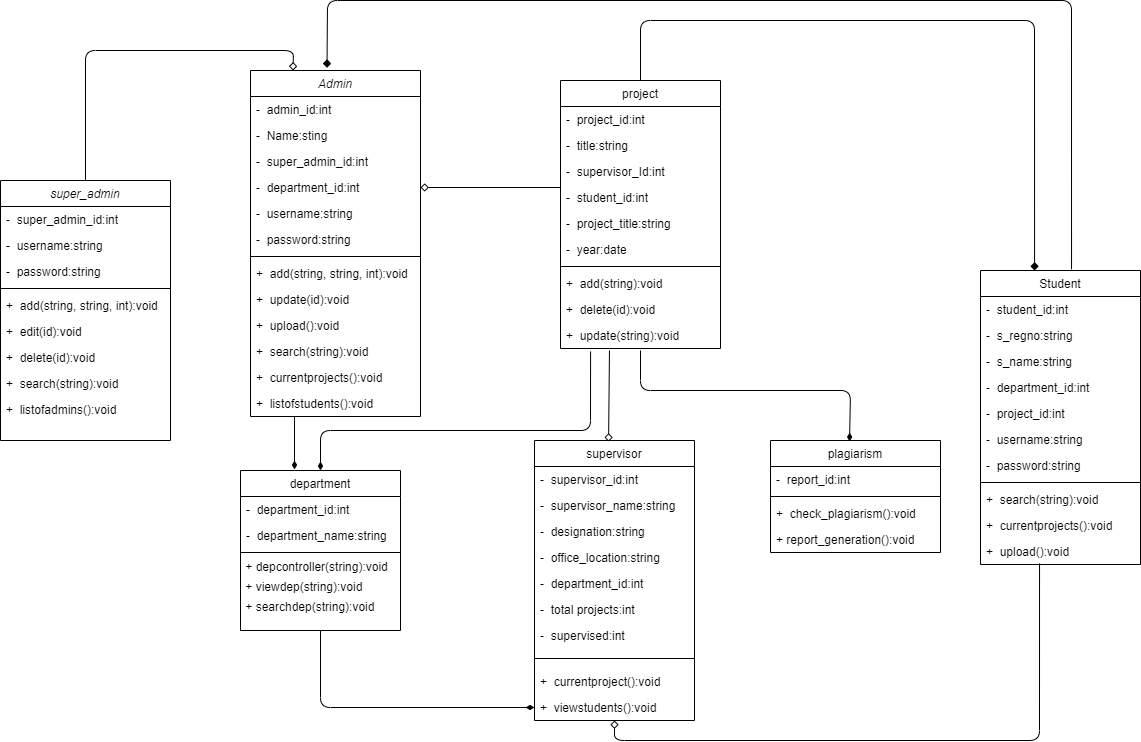
While dealing with software architecture we know that it describes the organization or structure of a system, where the system represents a collection of components that accomplish a specific function or set of functions.



*Figure 29 Software Architecture*

# Class Diagram

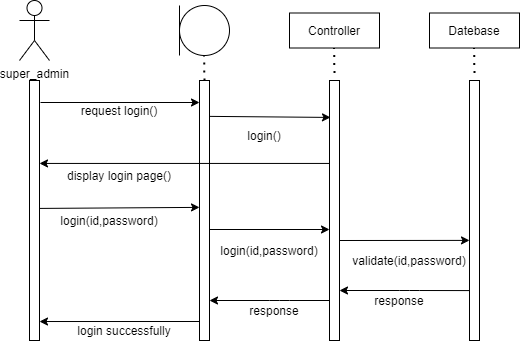
The class diagram describes the attributes and operations of a class and the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object- oriented languages:



*Figure 30 Class diagram*

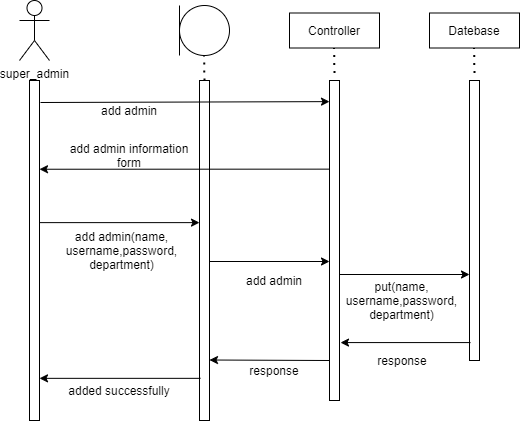
# Sequence Diagram

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.



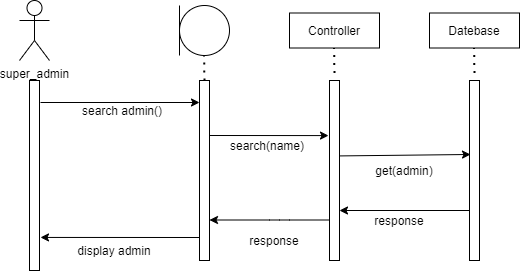
*Figure 31 Super admin login*

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.



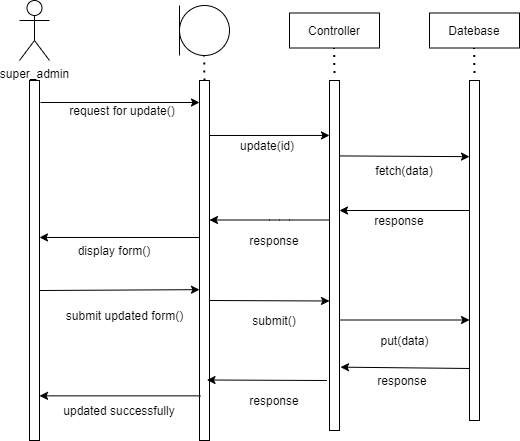
*Figure 32 Add admin*

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.



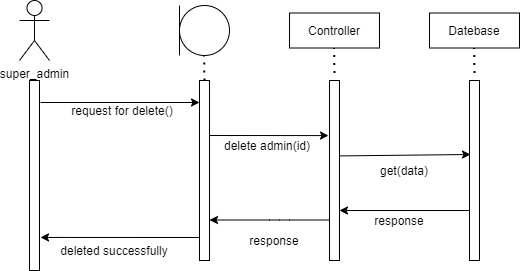
*Figure 33 Search admin*

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.



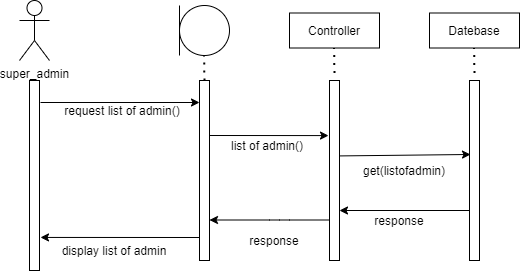
*Figure 34 Update admin*

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.



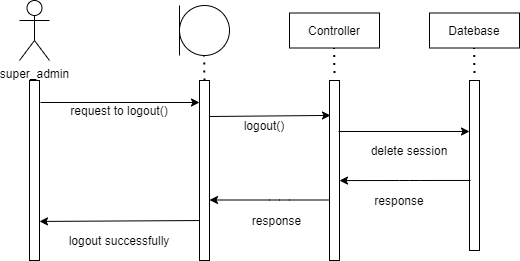
*Figure 35 Delete admin*

Super admin can delete an admin. The controller will get the data and delete it.



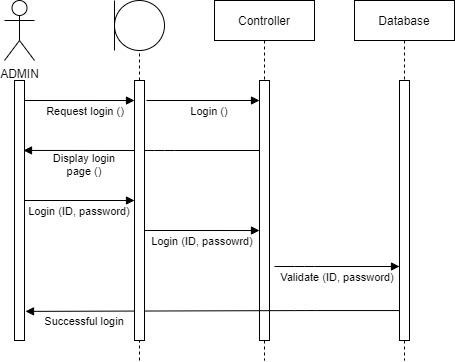
*Figure 36 list of admin*

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.



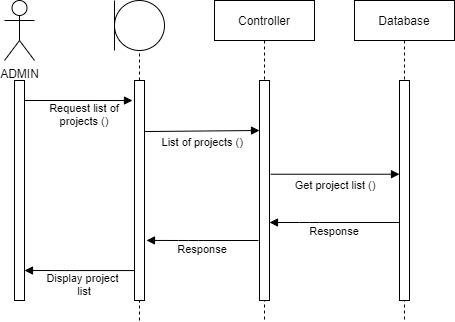
*Figure 37 Super Admin logout*

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



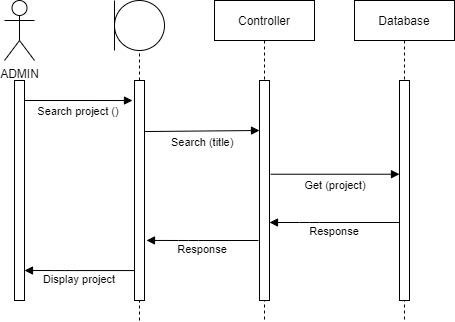
*Figure 38 Admin login*

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.



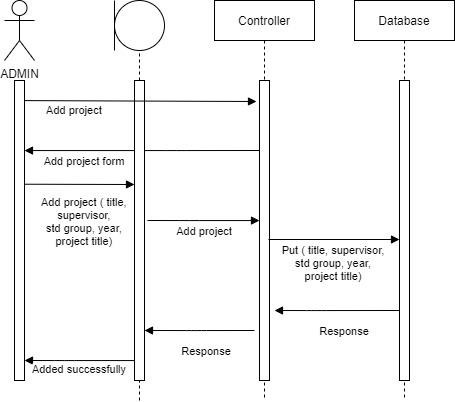
*Figure 39 Request list of project*

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.



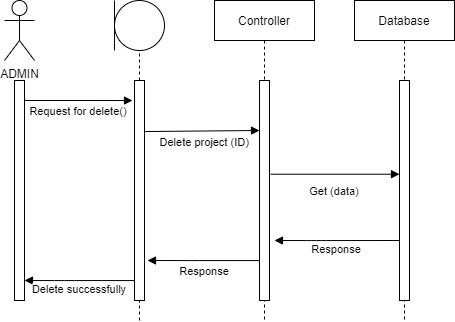
*Figure 40 Search project*

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.



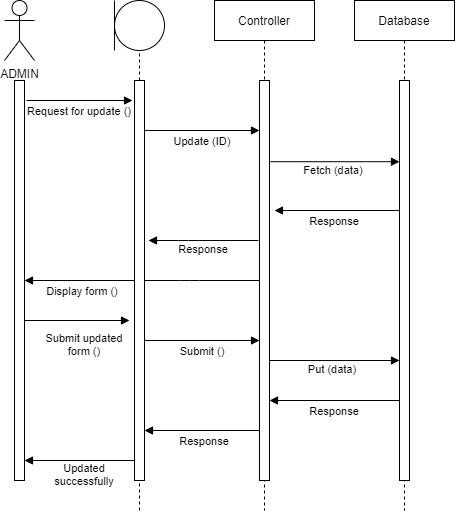
*Figure 41 Add project*

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.



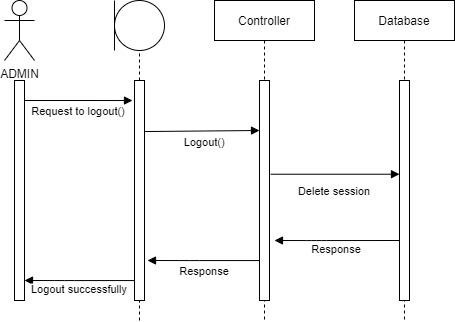
*Figure 42 Delete project*

Admin can request to delete a project. The controller will get the data from the database and will delete it.



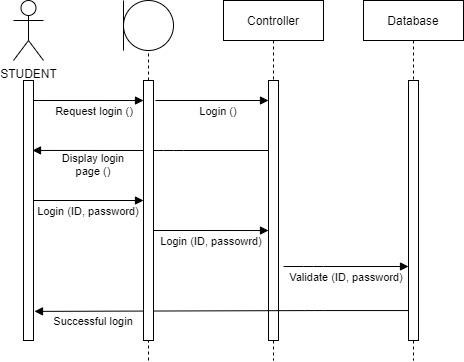
*Figure 43 Update project*

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.



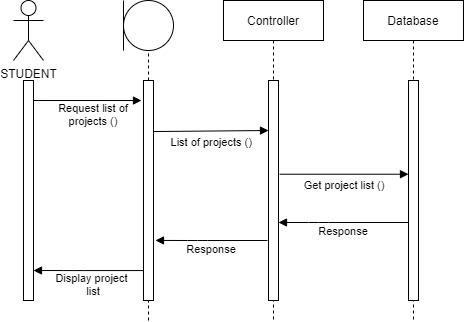
*Figure 44 Admin logout*

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



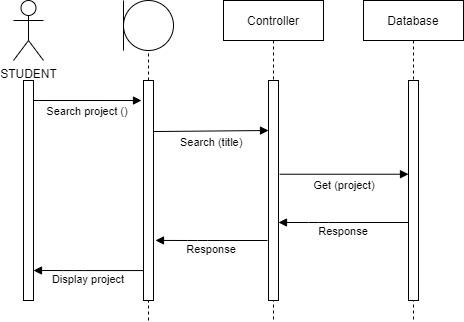
*Figure 45 Student login*

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.



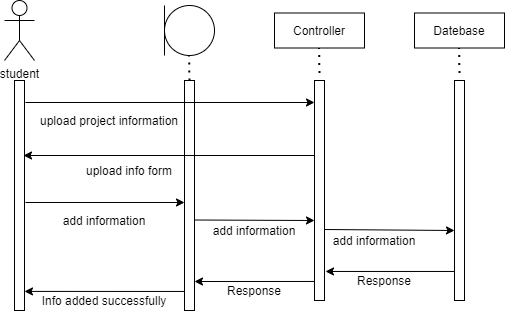
*Figure 46 Request list of projects*

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.



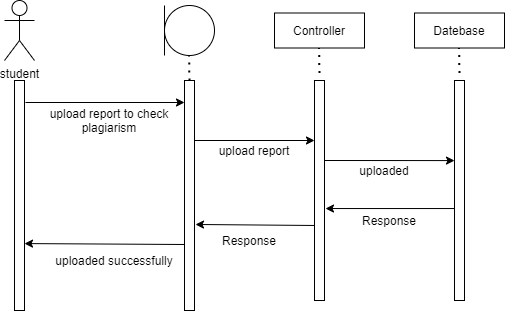
*Figure 47 Search project*

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.



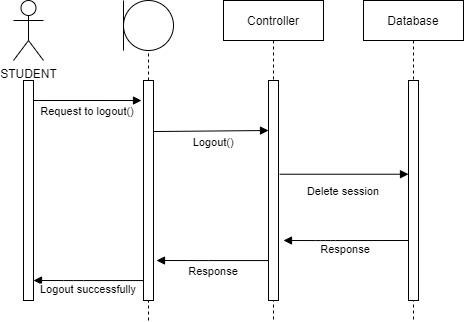
*Figure 48 Upload project information*

Student can upload the information about the project. An upload information form will be displayed and the student will add the information. The controller will add the information in the database.



*Figure 49 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.

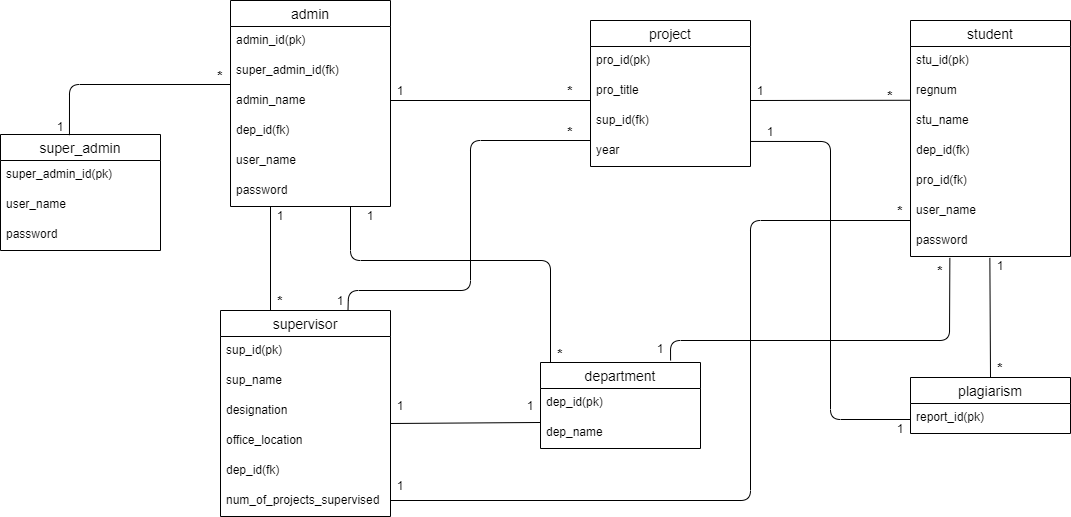


*Figure 50 Student logout*

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

# 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.



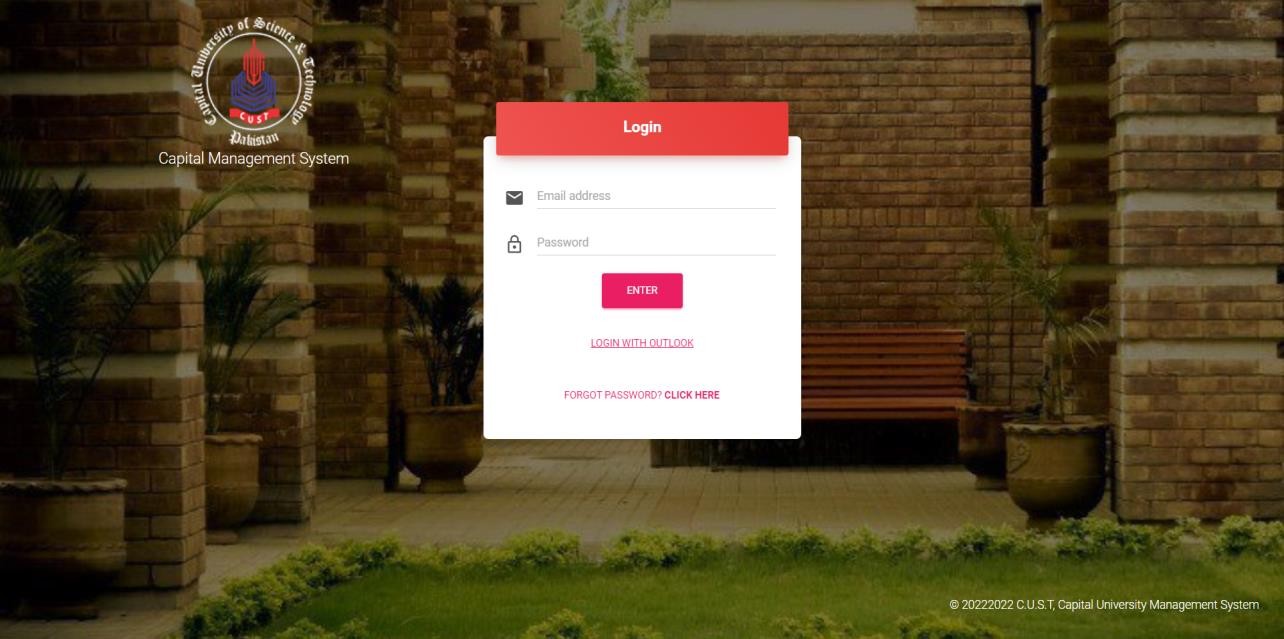
*Figure 51 Database Schema*

# 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.

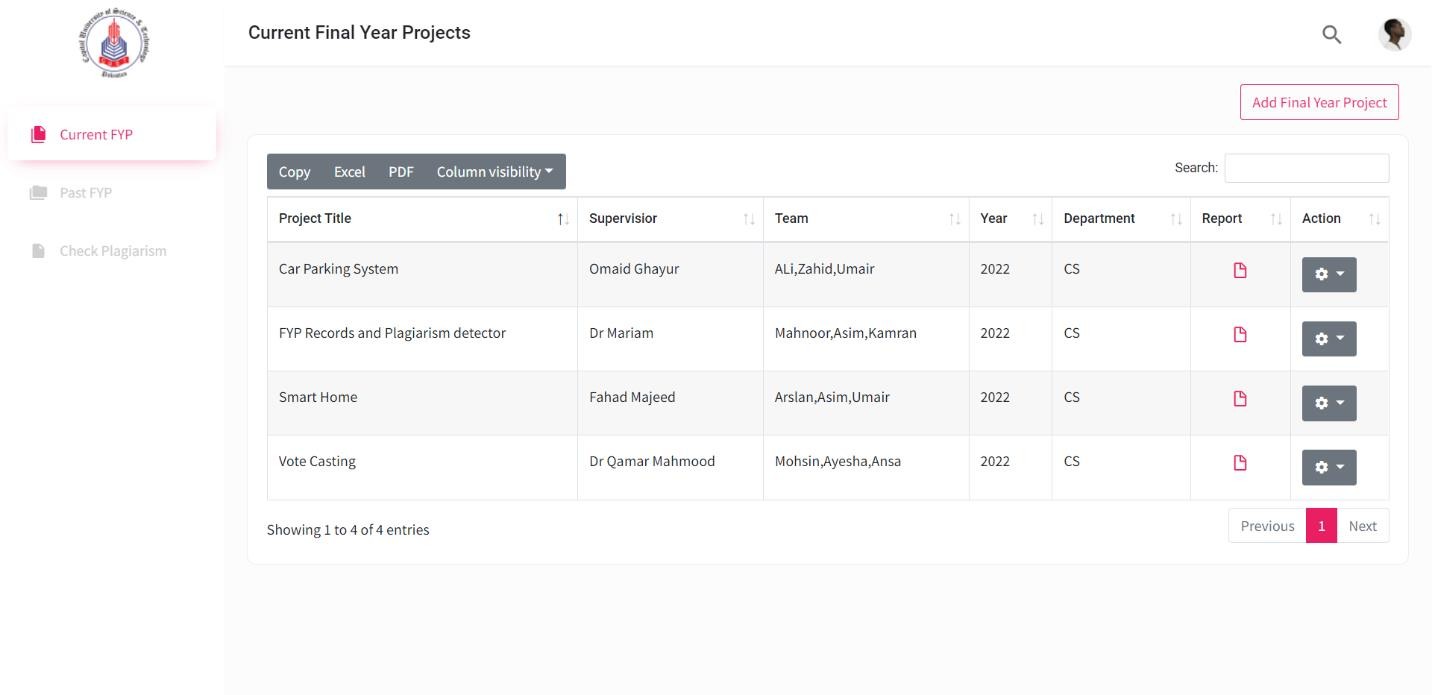
**User Sign in:**

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.



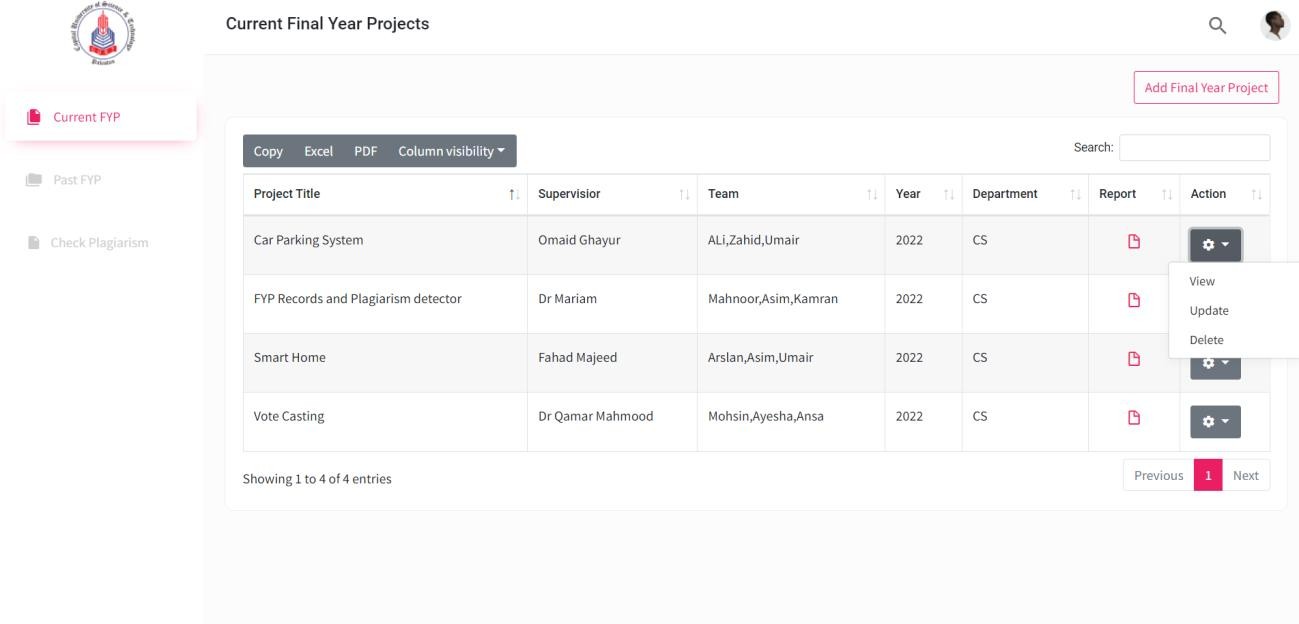
*Figure 52 Sign in GUI*

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:



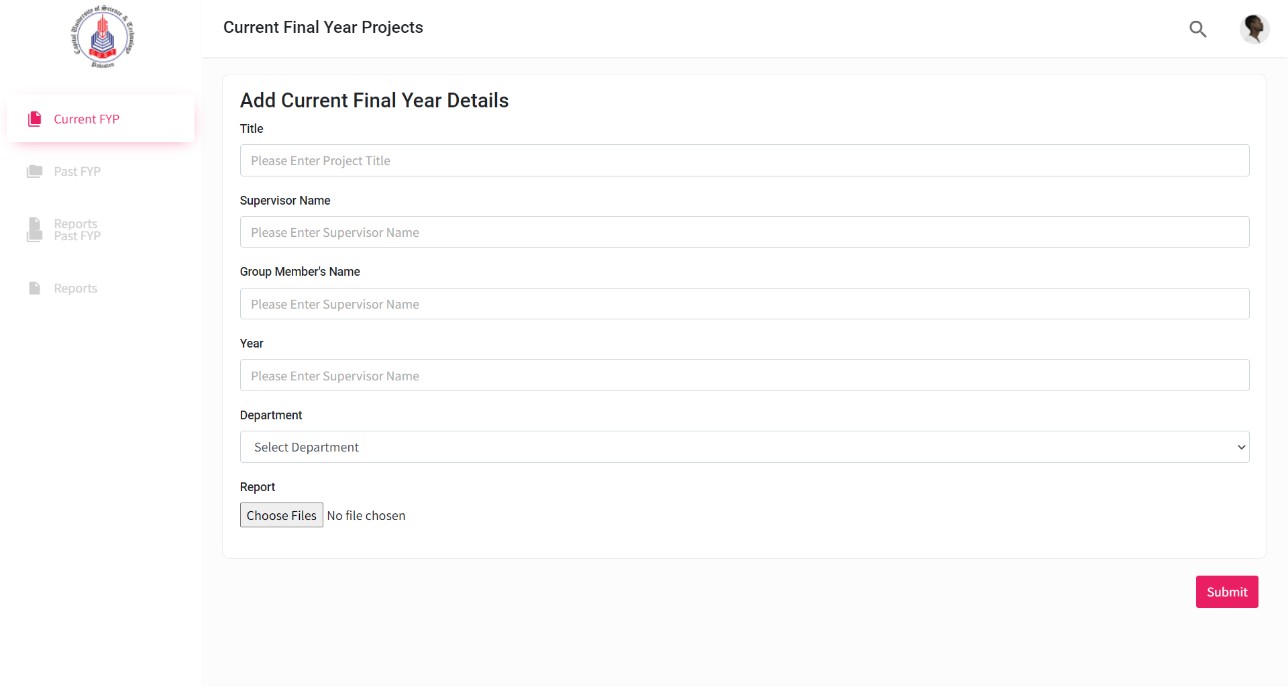
*Figure 53 List of projects*

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:



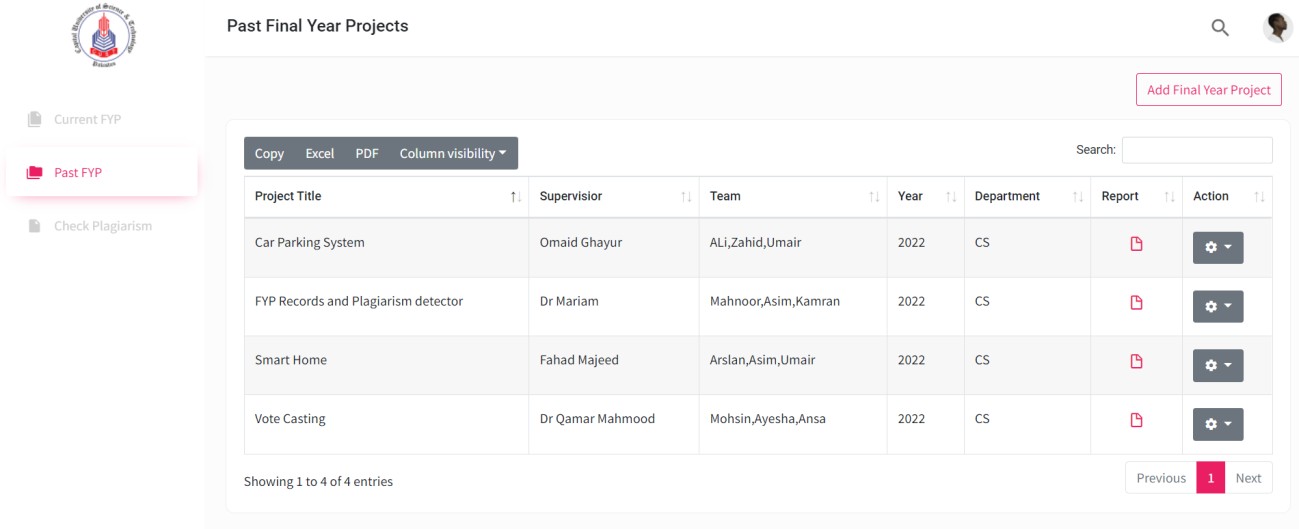
*Figure 54 CRUD operation*

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



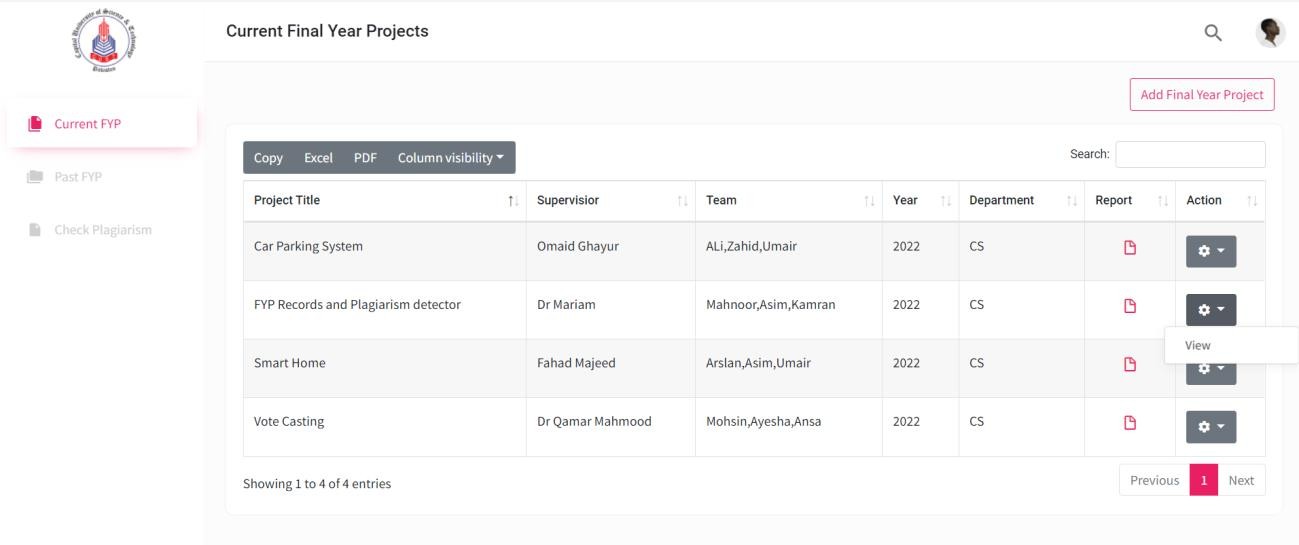
*Figure 55 Add project*

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:



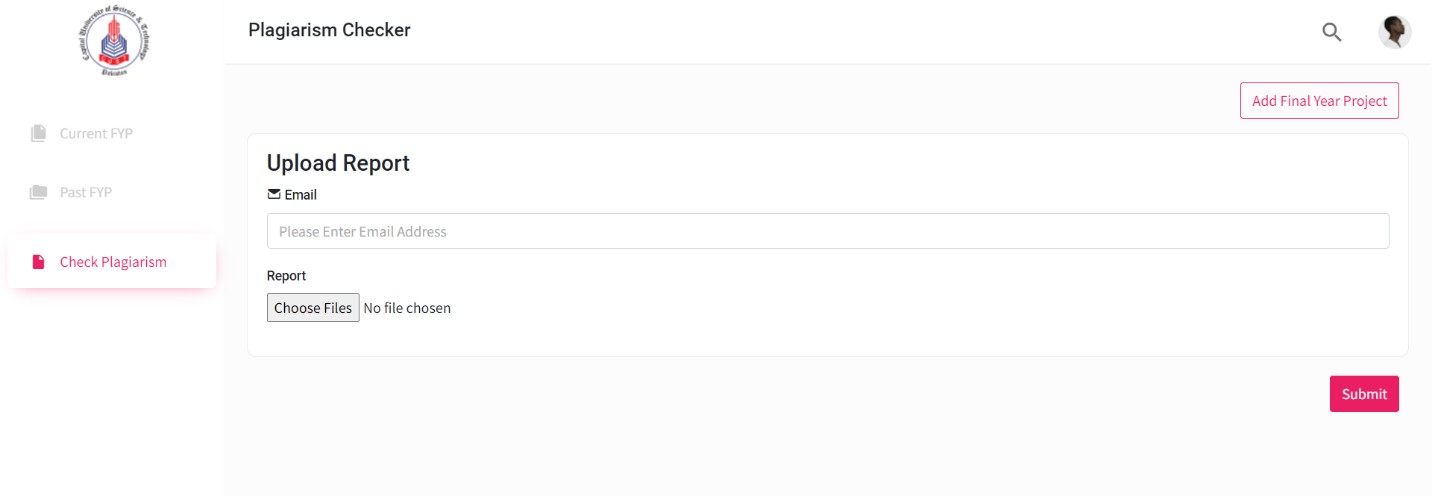
*Figure 56 past projects*

The student is able to able to view the report



*Figure 57 view report*

The GUI for report uploading for plagiarism checking:



*Figure 58 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.

# Coding Standards

The indention, 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.

# 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.

## Visual Studio code

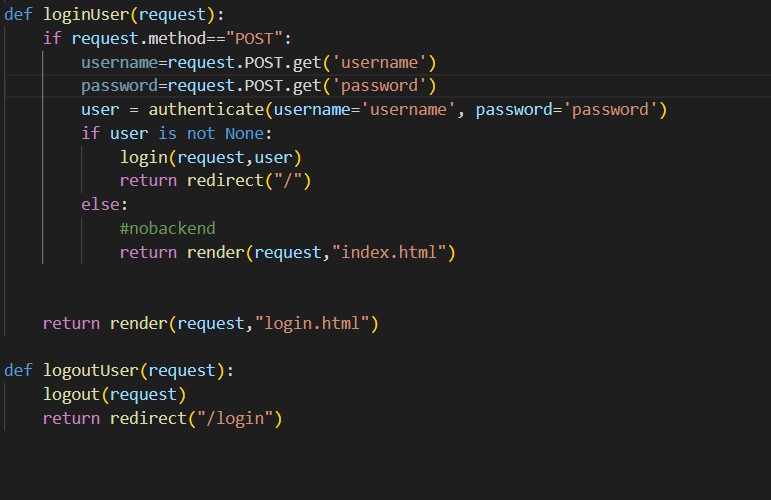
In general, the build process is very simple in this tool by building your code, you can quickly identify compile time errors such as incorrect syntax, misspelled keywords and type mismatches. You can also detect and correct runtime errors such as semantic errors and errors by building and running debug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

## Apache MYSQL

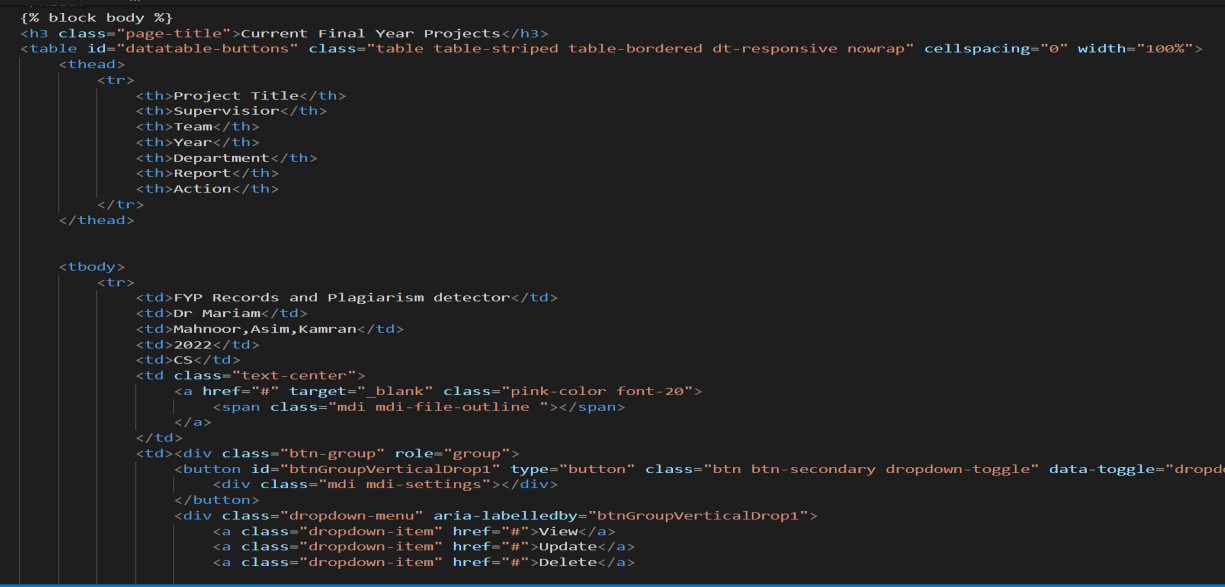
Apache is the web server that processes requests and serves web assets and content via HTTP. MySQL is the database that stores all your information in an easily queried format. We connected MySQL and Django as all the data will be stored in MySQL.

# Code Snippet

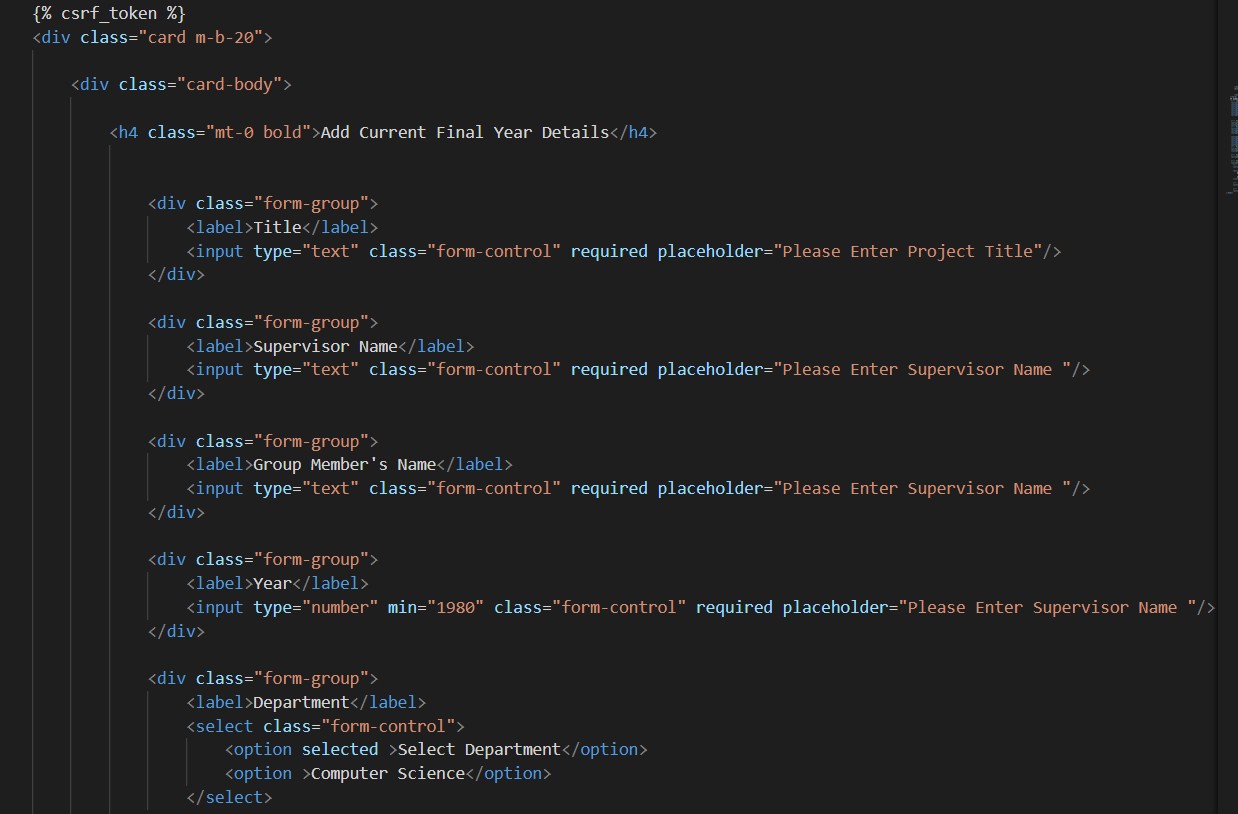
Here are the code snippets of major functionality of the system which show how the system interacts and performs various operations .It will discuss the logic of the system that will be implemented in the code. Some snippets code which are given below:



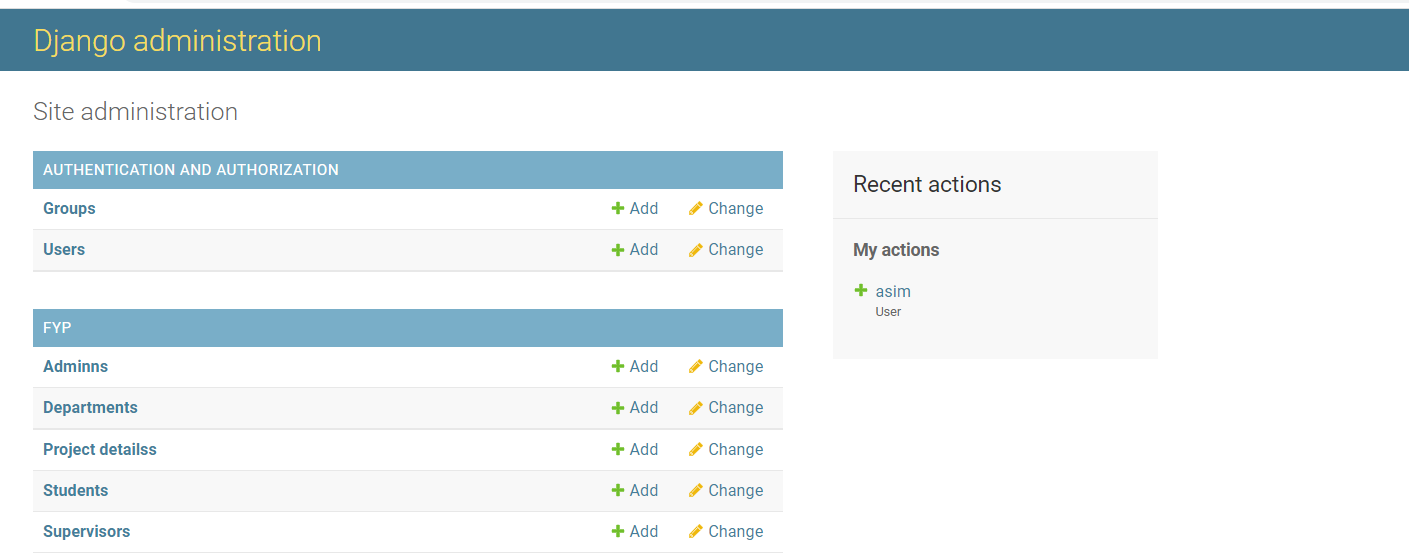
*Figure 59 user login and logout*



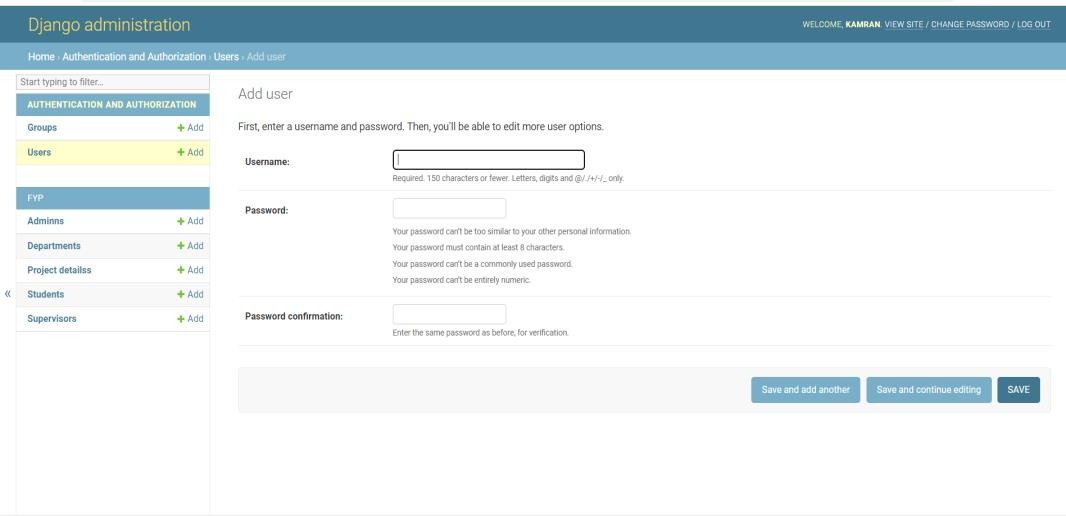
*Figure 60 admin CRUD operation*



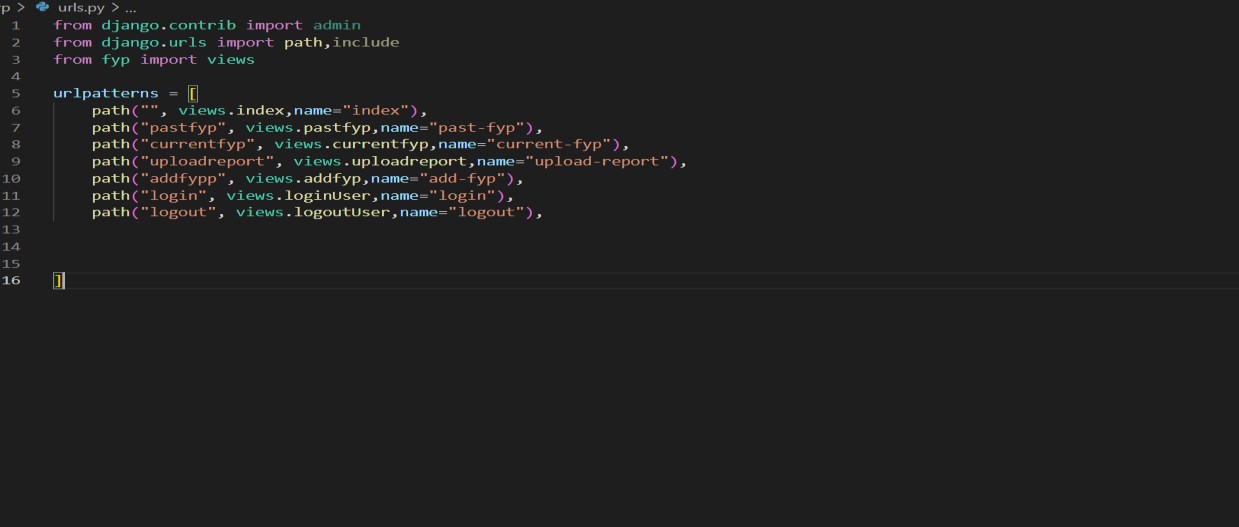
*Figure 61 add Fyp Project*



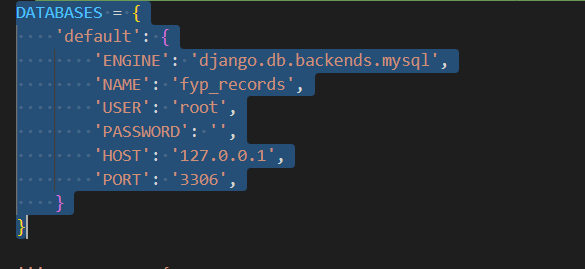
*Figure 62 super admin successful login*



*Figure 63 add admin*



*Figure 64 URLs.py*



*Figure 65 Database link*

# 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.

# 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.

# 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.

## 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 19 user successful login*

|  |  |
| --- | --- |
| Date: 29 July 2022 | *Test ID:*1 |
| *System:* FYP records and plagiarism | *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 | |

## 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 20 failed user login*

|  |  |
| --- | --- |
| Date: 29 July 2022 | *Test ID:*2 |
| *System:* FYP records and plagiarism | *Test Type:* Unit testing |
| *Objective:* Sign in user | |
| *Version:*1 | |
| *Input:* Sign in credentials(username, and password) | |

|  |
| --- |
| *Expected Result:* User will not be successfully login. |
| *Actual Result:* cannot login |

## 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 21 Add Admin*

|  |  |
| --- | --- |
| Date: 29 July 2022 | *Test ID:*2 |
| *System:* FYP records and plagiarism | *Test Type:* Unit testing |
| *Objective:* add admin | |
| *Version:*1 | |
| *Input:* add credentials of admin. | |
| *Expected Result:* User will be successfully added. | |
| *Actual Result:* passed. | |