THE KELKAR EDUCATION TRUST'S VINAYAK GANESH VAZE COLLEGE OF ARTS, SCIENCE AND COMMERCE

(Autonomous)

MULUND, MAHARASHTRA, 400081

DEPARTMETN OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the projectentitled," Student Problem Solver System",

is bonafied work of Steven Thomas bearing

Roll No: 4045 4051 submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATIONTECHNOLOGY.

11/03/24

Internal Guide

K.E.T.'S VAZE COLLEGE 1984

Poornima B

Head Of Department

External Guide 11/3/24

Date: 11 MAR 2024

Student Problem Solver System

THE APPROVAL PROJECT PROPOSAL

Name:	Steven Thomas
Title:	Student Problem Solver System
Name of Guide:	Mrs. Nanda Rupnar
Is this your first submission of project?	Yes
Control Id:	2019060868
Roll no:	4045A051

ACKNOWLEDGEMENT

I am glad to say that, I have satisfactorily reached my intentions to make this documentation. However, it would not have been possible without the kind support and help of many individuals. I would like to extend my sincere thanks to all of them.

I am highly indebted of my guide, **Ms. Nanda Rupnar** for her guidance and constant supervision as well as for providing necessary information regarding the project.

I would also like to extend my gratitude towards **Principal (Prof) Dr. Preeta Nilesh** and the Head of the Department, **Ms. Pournima Bhangale** for providing me with all the facilities that was required.

Then I would like to thank my parents who have helped me with their valuable suggestions and guidance which has been very helpful.

Last but not the least I would like to thank my classmates who have helped me a lot. Directly or indirectly their contribution was indispensable, and will always be remembered.

This opportunity has given me a valuable experience about software development for which I shall be thankful for the years to come.

Steven Thomas

DECLARATION

I hereby declare that the project entitled, **"Student Problem Solver System"** done at **V.G Vaze College, Mulund**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

to be submitted as final semester project as part of our curriculum.

Name: Steven Thomas

Signature:

TABLE OF CONTENTS

List Of Tables

List Of Figures

SYNOPSIS

1. Title: Student Problems management system

2. Statement about the problem

Student's face problems which they cannot complaint or state due to fear of revealing identity. Student can share their problems faced in the college, anonymously in this community where the problems most faced by the students is most visible to the handler who is representing the college.

Thus, making it easy for students for stating their problems and handler to solve the students most facing problems.

3. Why this topic?

Internet is providing the channel for students and college representative to communicate through WhatsApp etc. but the core problem of the student is losing their identity.

Thus, this software makes sure the identity of student is not revealed thus the raw problems of students can reach out to the college.

To make sure no invalid post is posted it is verified through the handler before it is posted publicly.

4. Objective and scope

- Making it easier for the college to see into student's most faced problems in the college
- To make sure student's identity is anonymous
- Giving the handler choice which problems can be solvable and making it priority to solve accordingly
- Scope restricted to a particular college.

5. Methodology

To develop this system. I am going to use incremental model. It is used because working on a little new technology so time required to learn and less documentation required in this model.

6. Proposed Architecture

To create the system, we will use the two-tier architecture, the client -server architecture. To achieve faster interaction between client and server, as this system involves constant client server interaction.

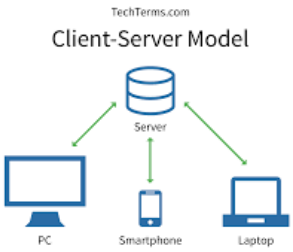


Fig 1.1 client server model

7. Requirement

7.1 Software Requirement

Frontend: HTML5 Javascript CSS

Backend: PHP

Database: MySQL

7.2 Hardware Requirement

Processor: 11th Gen Intel(R) Core (TM) i3-1115G4 @ 3.00GHz 3.00 GHz, 64-bit operating system, x64-based processor

Memory: 2 GB or more

Display: FHD display with a resolution of 1920 x 1080

7.3 Platform

Platform used is visual studio code for faster development.

8. Contribution

This website will be of a great help for the students. Most of the problems would reach out to the college and would be solved, thus making a healthy environment for the students.

Also saves the time and energy of physical meeting or writing a complaint in letter. Thus, making an overall better college experience.

9. Conclusion

This system will let the students post their problems without any fear or shyness, due to its anonymity.

It has voting system making the most faced issue reach out.

Reduces the time of the complaint reaching the handler.

Reduces the unnecessary usage of the resources.

Thus, more problems would reach out to the college and more problems would be solved.

Chapter 1

Introduction

1.1 Background

A student faces many problems in his/her college life. Many problems are not even addresses to the college due to student revealing their identity shyness or the process being tedious. Currently, there is no existing identical system. A similar system would be Reddit. The main goal of reddit is, through the voting system the best content is to be reached to the user. The reddit is for anything (memes, any matter, or problems).

The software being developed would be specifically for the college and students. In the Student Problem Management System, the most faced or critical problem can be addressed and voted by the students which would be taken up by the college handler.

1.2 Objectives

The objectives of the system are as follows:

- Providing a community where students can address by posting their problems in a quicker way without any shyness and fear of revealing identity.
- Providing a community where students can vote over the most faced or critical problems.
- Providing a community for college where they can view and take up the most faced problems by the students and prioritizing it to solve it accordingly.
- Reducing the time of the problems reaching out to the college thus making it a merit to the college.
- Reducing the resources usage in the usual physical process.

1.3 Purpose, scope, and applicability

1.3.1 Purpose

Providing a community where students can address by posting their problems in a quicker way without any shyness and fear of revealing identity and where the college they can view and take up the most faced problems by the students and prioritizing it to solve it accordingly.

1.3.2 Scope

Making students of a college address by posting their problems at this software. Thus, the scope of this software to work efficiently is for a particular college.

It consists of some limitations for user like notifications which can be upgraded in future.

1.3.3 Applicability

- This software is for the students of a college to help them address their problems without any shyness and fear of revealing identity.
- This software is for the college to take up the problems faced by the students and try to provide solutions to solvable problems in a swift way.
- Thus, a software for a college which consists of students posting their problems and handler (college representative) taking up the problems, which will reduce the overall time and resources used in its usual manual process.

Chapter 2

Survey of Technologies

1. HTML

HTML is a markup language. It provides the structures of a website so that the web browser knows what to show.

2. CSS

CSS (Cascading Style Sheets) is a simple design language used to stylize elements written in a markup language such as HTML. It simplifies the process of making web pages presentable by handling the look and feel part of a web page and allowing users to control multiple web pages all at once.

3. Programming Languages

- **JavaScript**

JavaScript is a lightweight, interpreted, or just in time compiled programming language with first class function. While it is most well-known as the scripting language for webpages, many web browsers environments also use it, such as Node.js, Apache CouchDB, and adobe acrobat. JavaScript is a prototype based, multi-paradigm, single threaded, dynamic language, supporting object oriented, imperative, and declarative(e.g., functional programming) styles. It is used by all web browsers, meteor, and lots of other frameworks. Jquery an popular part of the JavaScript library that simplifies and enhances web development by providing a concise and efficient way to manipulate HTML documents, handle events, create animations, and interact with web services. It is designed to make JavaScript coding more accessible and cross-browser compatible.

- **Coffee Script**

It is a kind of "dialect" of java script. It is viewed as simpler and easier on your eyes as a developer but it compiles or converts back into java script. Coffee script is an attempt to expose the good part of java script in a simple way. The code compiles one to one into equivalent JS and there is no interpretation at the runtime. You can use any existing java script library seamlessly from coffee script and vice-versa. The compiled output is readable, pretty-printed and tends to run as fast or faster than the equivalent handwritten java script.

- **Python**

It is one of the highly used programming languages. It is used by the Django framework and used in a lot of mathematical calculations.

- **Ruby**

Used by the ruby or rails framework. It is a dynamic, open-source programming languages with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to understand.

- **PHP**

PHP is a popular general-purpose scripting language that is especially suited to web development. Fast, flexible, and pragmatic, PHP powers everything from your blog to most popular websites in the world.

- **Java**

Used by android(Google) and a lot of desktop applications. Java is a commonly used language for the web development, especially on the server-side. Java web applications are distributed applications that run on the internet. Web applications that run on the internet. Web development with java allows us to create dynamic web pages where user can interact with the interface.

4. Frameworks

- **Node.js**

It is a server-side java script framework. It is a JavaScript runtime built on chrome's V8 JavaScript engine. As an asynchronous event-driven java script runtime, Node.js is designed to build scalable network applications.

- **Django**

Django is a high-level python web framework that encounters rapid development and clean, pragmatic design. Built by experienced developers, it takes cares of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It is free and open source.

- **Bootstrap**

A UI(user interface) framework for building with HTML/CSS/JavaScript.

- **Foundation**

A UI(user interface) framework for building with HTML/CSS/JavaScript.

- **WordPress**

A CMS(content management system) built on php. Currently, about 20% of all websites run on this framework.

- **.NET**

It is a full-stack framework built by Microsoft. It includes a large class library called Framework class library and provides language interoperability across several programming languages.

- **Angular.js**

It is a front-end java script framework. AngularJS lets you extend HTML vocabulary for your applications. The resulting environment is extraordinary expressive, readable, and quick to develop.

5. Databases

- **MongoDB**

It is an open-sourced NoSQL database and is currently the only database supported by Meteor. It is document data model naturally supports JSON and its expressive query language is simpler for developers to learn and use. Functionality such as automatic failover, horizontal scaling, and the ability to assign data to a location are built-in.

- **Redis**

It is the most popular key-value store. It is lighting fast for retrieving data but does not allow for much depth in the data storage. It has built in replication, Lua scripting, LRU eviction, transactions, and different levels of on disk persistence, and provides high availability via Redis Sentinel and automatic partitioning with Redis Cluster.

- **MySQL**

MySQL server is an open-source relational database management system which is a major support for web-based applications. MySQL server is used for data operations like querying, sorting, filtering, grouping, modifying, and joining the tables.

- **SQL server**

Microsoft SQL Server or MS SQL Server for short is the query language provided for data definition and manipulation. SQL Server is a Relational Database Management Systems which was developed and marketed by the Microsoft company. SQL and SQL servers are built as two layers where the SQL server is on the top for interacting with the relational databases.

6. Data Formats

It is the structure of how the data is stored.

- **JSON:**

It is quickly becoming the most popular data format. It is a lightweight data interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate.

- **XML:**

It was the main data format early in the web days and predominantly used by Microsoft systems.

- **CSV:**

Its data is formatted with commas. Excel data is typically formatted this way.

The technologies/system which will be used for the creation of the systems are:

1) HTML5 – as it is short and crisp syntax hence it became very easy to write and manage HTML5 code. It will be used make user-friendly front-end.

2) CSS -it will be used to make the website presentable and the style applied is consistency visible across variety of sites.

3) JAVASCRIPT-it will be used to validate, manipulate, and compute data and as it could support all modern browser and produce an equivalent result. It simplifies and enhances web development by providing a concise and efficient way to manipulate HTML documents, handle events, create animations, and interact with web services. It is designed to make JavaScript coding more accessible and cross-browser compatible.

4)PHP-as it is open source and adaptable at multiple platforms. PHP has a relatively simple and intuitive syntax, making it easy to learn for beginners. It also integrates seamlessly with HTML and MySQL, allowing developers seamlessly with HTML, allowing developers to embed code within webpages.

5)MySQL-as easy to use and higher efficiency. It is easy to learn and used for storing data in database server and retrieve and modify according to the functionality of the software. This language helps to add, delete, and update data making it easy to use.

6)JSON-used for proper access of data as it is lightweight and easy to use.

Chapter 3

Requirements and Analysis

3.1 Problem Definition

The system is built so that students could state their problems in this software without revealing their identity in less tedious manner. The student could post their problems and could vote on the problems they find mutual or which they find right. The handler first verifies the posts before being posted to be voted and can take up the most faced problems and coordinate to the college management to solve it.

3.2 Requirement Specifications

3.2.1 Requirement Gathering

Various ways to gather requirement:

1. Interview and surveys: Conducting interview and surveys with users to gather their preferences and requirements for the software.

2. User stories: Utilize user stories, which are concise descriptions of a specific user's requirement or need, to capture functional and non-functional requirements from the user's perspective.

3. Prototyping and Mock-ups: Create protypes or mock-ups of the software to allow users to visualize and interact with the proposed solution. Their feedback can help refine and clarify requirements.

I have used the survey, and user-story method.

The information on requirement was gathered by asking some questions and taking some opinions of students and mentors of college. The questionnaire was prepared and passed by using Google forms, the link of the form is:

1. Student form: Software Requirement Form-Student (google.com)(https://forms.gle/imxrodxJzd6VsadeA)

The questions that were asked to students are as follows:

1.Does your problems faced in college get resolved?

2.Do you find it difficult to address the problems to the college?

3.Does it take long time from the problem being identified and getting resolved?

4.Reasons of not telling problems faced in the college to the college:

5.In which mode would u prefer to address the problem to college

6.Would it become easier if a software is provided in which your identity is not revealed and the above reasons are resolved and you can state your problem to your college?

7. Whether students should be able to vote on the problems and based on votes, can decide the priority of which problems is to be taken up without revealing any identity.

Based on these above questions the responses of the students who filled the form was recorded and they are all shown below:

	Timestamp	Email a	1.Does your problems fa	2.Do you find it difficult to	3.Does it take long time f	4.Reasons of not telling	5.In which mode would	6.Would it become ea	7. Whether students shou
2	27/04/2023 11:06:43	sakshis	No	Sometimes	Yes	Problems never getting	Online	Yes	Yes
3	27/04/2023 19:38:11	christin	No	Yes	Yes	Fear of Revealing Identi	Any mode	Yes	Yes
4	27/04/2023 23:01:33	adityap	No	Yes	Yes	Shyness, Fear of Revea	Any mode	Yes	Yes
5	28/04/2023 12:19:32	kumbha	Sometimes	Sometimes	No	Fear of getting scold	Any mode	Yes	Yes
6	28/04/2023 12:23:31	omkarg	Sometimes	Yes	Yes	Problems never getting	Online	Yes	Yes
7	28/04/2023 12:26:26	hrushis	No	Yes	Yes	Fear of Revealing Identi	Online	Yes	Yes
8	28/04/2023 12:32:43	sameek	Yes	No	Sometimes	Shyness	Any mode	Yes	Yes
9	28/04/2023 12:59:50	sherem	Sometimes	Yes	Yes	Shyness, Fear of Revea	Online	Yes	Maybe
10	28/04/2023 13:00:14	sanikac	Sometimes	Yes	Yes	Problems never getting	Online	Yes	Yes
11	28/04/2023 13:09:14	prathan	Sometimes	Most of the times	Yes	Fear of getting scold	Online	Yes	Yes
12	28/04/2023 13:15:33	nandan	Sometimes	Sometimes	Yes	Fear of Revealing Identi	Any mode	Yes	Yes
13	28/04/2023 13:17:35	sristiha	Sometimes	Yes	Most of the times	Shyness, Fear of Revea	Any mode	Yes	Yes
14	28/04/2023 13:22:10	ranapiy	Sometimes	No	Yes	Fear of getting scold	Offline	Maybe	Yes
15	28/04/2023 13:22:10	gauripa	No	Yes	Yes	Problems never getting	Any mode	Yes	Yes
16	28/04/2023 13:53:08	aryanjb	No	Sometimes	Yes	Fear of Revealing Identi	Offline	Yes	Yes
17	28/04/2023 14:08:24	tanvinir	Yes	Sometimes	No	Fear of getting scold	Online	Yes	Yes
18	28/04/2023 14:46:22	anushk	Sometimes	Most of the times	Most of the times	Shyness, Fear of Revea	Online	Yes	Yes
19	28/04/2023 15:15:37	sid.cha	Most of the times	No	Sometimes	No such reasons	Any mode	Yes	Yes
20	28/04/2023 15:18:15	orevdel	No	No	Yes	Problems never getting	Online	Yes	No
21	28/04/2023 15:24:54	abhidny	No	Sometimes	Yes	Fear of Revealing Identi	Any mode	Yes	Yes
22	28/04/2023 20:41:33	vandan	No	Yes	Sometimes	Shyness, Fear of Revea	Any mode	Yes	Yes
23	28/04/2023 22:25:05	sakshis	Sometimes	Yes	Yes	Fear of Revealing Identi	Online	Yes	Yes
24	28/04/2023 22:42:18	jatinmis	Sometimes	Sometimes	Most of the times	Problems never getting	Any mode	Yes	Maybe
25	28/04/2023 23:26:38	varunsu	No	Yes	Yes	Shyness, Fear of Revea	Online	Yes	Yes
26	29/04/2023 01:53:36	tcelucu	No	Yes	Yes	Problems never getting	Any mode	Maybe	Maybe
27	29/04/2023 08:19:09	VAIDEH	Sometimes	Yes	Yes	Tedious process	Any mode	Yes	Yes
28	29/04/2023 10:28:51	anuppa	Yes	No	Sometimes	Fear of Revealing Identi	Online	Yes	Yes
29	29/04/2023 11:12:33	gauravt	No	Yes	Yes	Problems never getting	Any mode	Yes	Yes
30	30/04/2023 16:57:36	AKSHA	Sometimes	Sometimes	Yes	Fear of getting scold	Any mode	Yes	Yes
31	01/05/2023 12:41:36	anisha.	Yes	Sometimes	No	Shyness, Fear of Revea	Online	Yes	Yes

Figure 3.2.1.1: Google Form Result

The response to the problems that they normally seek (with the percentage according to survey):

1.The problems faced by the student doesn't get solved (42.4%) or get solved sometimes (42.4%)

2.The problems faced by the students are difficult to address for 45.5% and sometimes difficult for 33.3%.

3.The problems faced in the college takes long time from being identified and getting resolved is found to be true to 66.7% students

4.The top 4 reasons of not telling the problems faced in college by the students were-

i. Fear of getting scold (51.8%)

ii. Problems never getting solved (45.5%)

iii. Fear of Revealing Identity (42.4%)

iv. Tedious process (30.3%)

v. 51.5% students are comfortable addressing their problems online.

vi.93.9% students are in favourable of wanting this software.

vii.87.9% students wanted the voting feature to the problems they face.

The response about any features that they wished there was in the website are as follows:

1. only one vote per student

2. Identity of any person should not be revealed to teachers

3.Information about the previous problems which were solved

4.History of the post posted to be provided to the student's profile.

5.Categorization of the problems.

3.2.2 Requirement Analysis

3.2.2.1 Functional Requirements

1.Identity of Student to be anonymous

2.Problems could be posted and managed.

3.Problems could be viewed and voted.

4.View profile and recent posted.

5.Handler can verify and remove posts sent to verification

6.Admin can manage(add,delete,edit) details of all the users.

7.Admin can manage(add,delete) the posts.

8.Support page through which any issues can be communicated to admin directly.

3.2.2.2 Non-Functional Requirements

1. Security

It is achieved in login page by validating through username and password. In the registration page first the student's control-id is validated then the email is authorised by through OTP verification.

2. Availability

It is achieved by creating backup database in case of any corruption or damage to the database. Regular backup and monitoring of database are to be done.

3. User-feedback

Any issues faced by the user will be handle through the contact us page or just mailing us the issue through the mail button set in every page footer.

4. Efficiency

It is achieved by using lightweight data format-JSON while fetching data through the ajax in the jQuery. And MySQL and PHP are used for the efficient server interaction.

5. Reliability

By making the software available and efficient it becomes reliable to the users to use whenever required. Every action is given an alert (user message) thus making every action user friendly. Thus, making the software reliable.

3.2.2.3 System Requirements

1. Registration

The student-user must first register its account in order to use the site. Without registering the user will be able to use the website.

- Input: User Details (controlid, rollno, email, otp (to the email), username, password)
- Output: Registration successful (redirect to the main page (index page))
- Pre-Condition: No existing user with same credentials and it should correspond to the details of the college student data table.
- Post-Condition: User able to login in with his/her account

2. Login

After registering the student-user will have to login in the system by the username/id and the password and after the credentials is verified the user will be logged in the system. For handler and admin, they would be given their username and password before only, registration not required.

- Input: username and password
- Output: successful login
- Pre-Condition: user should be registered (for the students). For handler and admin, they should be added to database through either admin page or manually added to database.
- Post-Condition: moved to user home page

3. Post

After login the student-user can post problems faced by them through the menu post tab.

- Input: post tab
- Output: show the posting page
- Pre-Condition: valid data should be entered
- Post-Condition: post is sent to handler for verification.

For Handler, in this tab they can verify or remove the post to be verified.

For Admin, they can verify or delete the removed post and remove the to be verified post

4. View and Vote

After login the student-user can view and vote the posts according to their opinions.

- Input: home tab or login
- Output: show the home page
- Pre-Condition: user must be logged in
- Post-Condition: post is sent to handler for verification.

For Handler, in this tab they can only view the voting done by the other student users and take up the problem and try to solve it by taking required actions.

For Admin, they can remove the verified post.

5. Profile

After login the all the user can view their profile in which they can view their details. And can edit details.

- Input: profile tab
- Output: show the profile page and option to edit details
- Pre-Condition: user must be logged in
- Post-Condition: all profile details are shown

6. Support Page

After login the all the user can navigate to contact us to state the software related issues.

- Input: contact us tab
- Output: show the contact us page
- Pre-Condition: user must be logged in
- Post-Condition: contact us form through which mail could be sent.

6. User Update Page

After login the only the admin can navigate to this page where he can manage (add,delete and edit) the users-student, handler and admin.

- Input: User Updater
- Output: show the User Update page
- Pre-Condition: user must be logged in.
- Post-Condition: user-specific tab to be selected according to which users can managed.

3.3 Task Table and Gantt Chart

Task No.	Task name	Starting date	Ending date	Actual Starting date	Actual Ending Date
T1	Synopsis	1-April-23	15-April-23	1-April-23	12-Apr-23
T2	Approval	1-April-23	15-April-23	3-April-23	12-April-23
T3	Chapter-1 Introduction	17-April-23	24-April-23	19-April-23	20-April-23
T4	Chapter-2 Survey of Technologies	13-June -23	19-June-23	17-June-23	19-June-23
T5	Chapter-3 Requirement and Analysis 3.1 Problem Definition 3.2 Requirement Specification	13-June-23	19-June-23	17-June-23	19-June-23
T6	Task Table and Gnat chart	1-July-23	6-October-23	1-July-23	6-October-23
T7	Chapter -4 System Design 4.1 Business Rule 4.2 Module Diagram 4.3 ER diagram	14-July-23	18-August-23	17-July-23	10-August
T8	4.4 Schema	1-August-23	10-August-23	3-August-23	5-August-23
T9	4.5 Data Flow Diagram	10-August-23	31-August-23	12-August-23	31-August-23

T10	4.6 Class Diagram	10-August-23	31-August-23	12-August-23	31-August-23
T11	4.7 Use case and Scenario	28-August-23	13-September-23	28-August-23	9-September-23
T12	4.8 Sequence Diagram	13-September-23	15-September-23	13-September-23	15-September-23
T13	4.9 Activity Diagram	16-September-23	03-October-23	17-September-23	03-October-23
T14	4.10 State Chart Diagram	16-September-23	03-October-23	18-September-23	03-October-23
T15	4.11 UI Design	16-September-23	03-October-23	19-September-23	03-October-23
T16	4.12 Test Cases	01-October-23	05-October-23	04-October-23	05-October-23
T17	**Re-engineering** Chapter 1	1-November-23	4-November-23	1-November-23	1-November-23
T18	Chapter 2	1-November-23	4-November-23	1-November-23	1-November-23
T29	Chapter 3	1-November-23	4-November-23	1-November-23	1-November-23
T20	Chapter 4	1-November-23	4-November-23	1-November-23	1-November-23
T21	**Unit 1: Home & Registration & Login Page** Design Module Home	10-November-23	11-November-23	10-November-23	11-November-23
T22	Debugging Module Home	10-November-23	11-November-23	10-November-23	11-November-23
T23	Testing Module Home	11-November-23	11-November-23	11-November-23	11-November-23
T24	Design Module Register	11-November-23	12-November-23	11-November-23	12-November-23

T25	Debugging Module Register	12-November-23	12-November-23	12-November-23	13-November-23
T26	Testing Module Register	12-November-23	13-November-23	13-November-23	14-November-23
T2	Design Module Login	13-November-23	13-November-23	14-November-23	14-November-23
T2	Debugging Module Login	14-November-23	15-November-23	15-November-23	16-November-23
T2	Testing Module Login	14-November-23	15-November-23	15-November-23	16-November-23
T3	Integrating All completed Unit	15-November-23	16-November-23	16-November-23	17-November-23
T3	Debugging	16-November-23	16-November-23	17-November-23	17-November-23
T3	Testing	16-November-23	16-November-23	17-November-23	17-November-23
T3	**Unit 2: Student Home Page (View and Vote Post)** Design Student Home Page	17-November-23	18-November-23	18-November-23	19-November-23
T3	Debugging Student Home Page	18-November-23	18-November-23	19-November-23	19-November-23
T3	Testing Student Home Page	18-November-23	18-November-23	19-November-23	19-November-23

T3	Integrating All completed Unit	19-November-23	20-November-23	20-November-23	21-November-23
T3	Debugging	20-November-23	20-November-23	21-November-23	21-November-23
T3	Testing	20-November-23	20-November-23	21-November-23	21-November-23
T3	**Unit 3: Send Post** Design Send Post	21-November-23	22-November-23	22-November-23	23-November-23
T4	Debugging Send Post	22-November-23	22-November-23	23-November-23	23-November-23
T4	Testing Send Post	22-November-23	22-November-23	23-November-23	23-November-23
T4	Integrating All completed Unit	23-November-23	24-November-23	24-November-23	25-November-23
T4	Debugging	24-November-23	24-November-23	25-November-23	25-November-23
T4	Testing	24-November-23	24-November-23	25-November-23	25-November-23
T4	**Unit 4: Profile Page** Design Profile Page	25-November-23	26-November-23	26-November-23	27-November-23
T4	Debugging Profile Page	26-November-23	26-November-23	27-November-23	27-November-23
T4	Testing Profile Page	26-November-23	26-November-23	27-November-23	27-November-23
T4	Integrating All completed Unit	27-November-23	28-November-23	28-November-23	29-November-23

T4	Debugging	28-November-23	28-November-23	29-November-23	29-November-23
T5	Testing	28-November-23	28-November-23	29-November-23	29-November-23
T5	**Unit 5: Contact Us** Design Contact Us Page	29-November-23	30-November-23	30-November-23	1-December-23
T5	Debugging Contact Us Page	30-November-23	30-November-23	1-December-23	1-December-23
T5	Testing Contact Us Page	30-November-23	30-November-23	1-December-23	1-December-23
T5	Integrating All completed Unit	1-December-23	2-December-23	2-December-23	3-December-23
T5	Debugging	2-December-23	2-December-23	3-December-23	3-December-23
T5	Testing	2-December-23	2-December-23	3-December-23	3-December-23
T5	**Unit 6: Handler Home Page (View Post)** Design Handler Home Page	3-December-23	4-December-23	4-December-23	5-December-23
T5	Debugging Handler Home Page	4-December-23	4-December-23	5-December-23	5-December-23

T5	Testing Handler Home Page	4-December-23	4-December-23	5-December-23	5-December-23
T6	Integrating All completed Unit	5-December-23	6-December-23	6-December-23	7-December-23
T6	Debugging	6-December-23	6-December-23	7-December-23	7-December-23
T7	Testing	6-December-23	6-December-23	7-December-23	8-December-23
T7	**Unit 7: Verify and Remove Post Page** Design Verify and Remove Post	7-December-23	8-December-23	9-December-23	10-December-23
T7	Debugging Verify and Remove Post	8-December-23	8-December-23	10-Decemeber-23	10-Decemeber-23
T7	Testing Verify and Remove Post	8-December-23	8-December-23	10-Decemeber-23	10-Decemeber-23
T7	Integrating All completed Unit	9-December-23	10-December-23	11-Decemeber-23	12-Decemeber-23
T7	Debugging	10-December-23	10-December-23	12-Decemeber-23	12-Decemeber-23
T7	Testing	10-December-23	10-December-23	12-Decemeber-23	12-Decemeber-23
T7	**Unit 8: Admin Home Page (View and Remove Post)**	11-December-23	12-December-23	13-Decemeber-23	14-Decemeber-23

	Design Admin Home Page				
T7	Debugging Admin Home Page	12-December-23	12-December-23	14-Decemeber-23	14-Decemeber-23
T7	Testing Admin Home Page	12-December-23	12-December-23	14-Decemeber-23	14-Decemeber-23
T8	Integrating All completed Unit	13-December-23	14-December-23	15-Decemeber-23	16-Decemeber-23
T8	Debugging	14-December-23	14-December-23	16-Decemeber-23	16-Decemeber-23
T8	Testing	14-December-23	14-December-23	16-Decemeber-23	16-Decemeber-23
T8	**Unit 9: Verify, Remove and Delete Post Page** Design Verify and Remove Post	15-December-23	16-December-23	17-Decemeber-23	18-Decemeber-23
T8	Debugging Verify and Remove Post	16-December-23	16-December-23	18-Decemeber-23	18-Decemeber-23
T8	Testing Verify and Remove Post	16-December-23	16-December-23	18-Decemeber-23	18-Decemeber-23
T8	Integrating All completed Unit	17-December-23	18-December-23	19-Decemeber-23	20-Decemeber-23
T8	Debugging	18-December-23	18-December-23	20-Decemeber-23	20-Decemeber-23

T8	Testing	18-December-23	18-December-23	20-Decemeber-23	20-Decemeber-23
T8	**Unit 10: Manage User Page (View and Remove Post)** Design Manage User Page	19-December-23	20-December-23	21-Decemeber-23	22-Decemeber-23
T9	Debugging Manage User Page	20-December-23	20-December-23	22-Decemeber-23	22-Decemeber-23
T9	Testing Manage User Page	20-December-23	20-December-23	22-Decemeber-23	22-Decemeber-23
T9	Integrating All completed Unit	21-December-23	21-December-23	23-Decemeber-23	23-Decemeber-23
T9	Debugging	21-December-23	21-December-23	23-Decemeber-23	23-Decemeber-23
T9	Testing	22-December-23	22-December-23	24-Decemeber-23	24-Decemeber-23

Table 3.3.1: Task Table

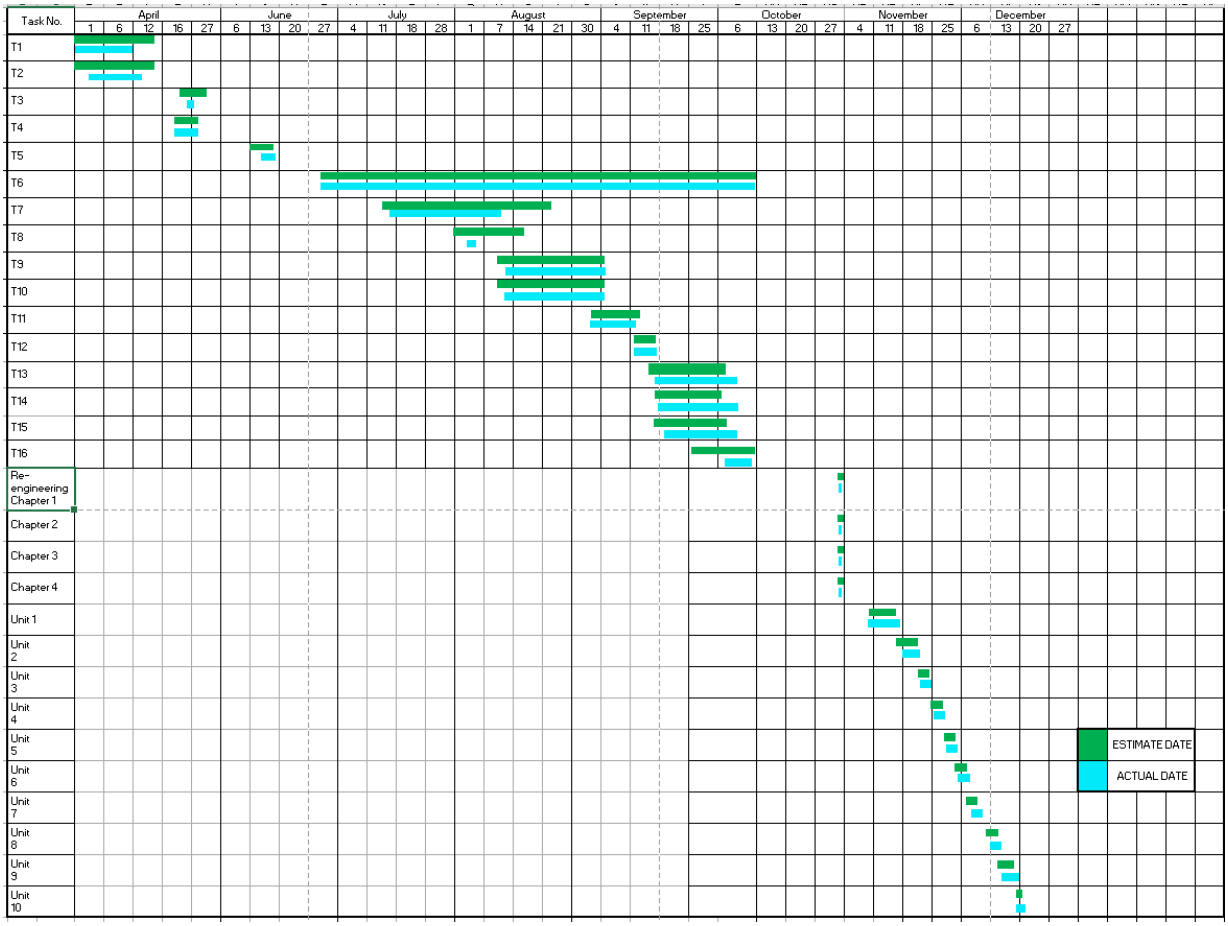


Figure 3.3.1: Gantt Chart

Chapter 4

System Design

4.1 Business Rule

- When the web application is opened the user can either sign in if they have already registered or must sign up.
- To sign up the user must be a student of college having a valid control id, roll no and email given to college. Roll no and email should match with the control id. Control id should be a 10-digit number while roll no should be 8 digit alpha numeric. An OTP would be sent while registering for authentication of mail.
- While registering the user will have to create username and its length would be between 8 and 20 and minimum 1 letter, 1 number required and other than underscore (_) no special character is allowed and password which will be composed of minimum 1 capital letter, 1 number, 1 special character, and the length of the password should be greater than 8.
- If user have already registered (for students) or have credentials (admin, handler) can login by entering their username and password. The admin and handler would be given credentials in hand (no registration required).
- After login the respective users would get their functionality according to their designation (student, handler, and admin).
  Student is allowed to view, vote and post problems. View profile and Contact admin and Logout.
  Handler is allowed to view, verify, and remove problems. View profile and Contact admin and Logout.
  Admin is allowed to view, verify, remove, and delete problems and can manage users (create, view update, and delete). View profile and Contact admin and Logout.

4.2 Module Diagram

Module diagram is a diagram which is used for showing the allocation of classes and objects to module in the physical design of a system. Module Diagram indicates the partitioning of the system architecture. Through this diagram, it is possible to understand the general physical architecture of a system. The two essential elements of a module diagram are modules and their dependencies.

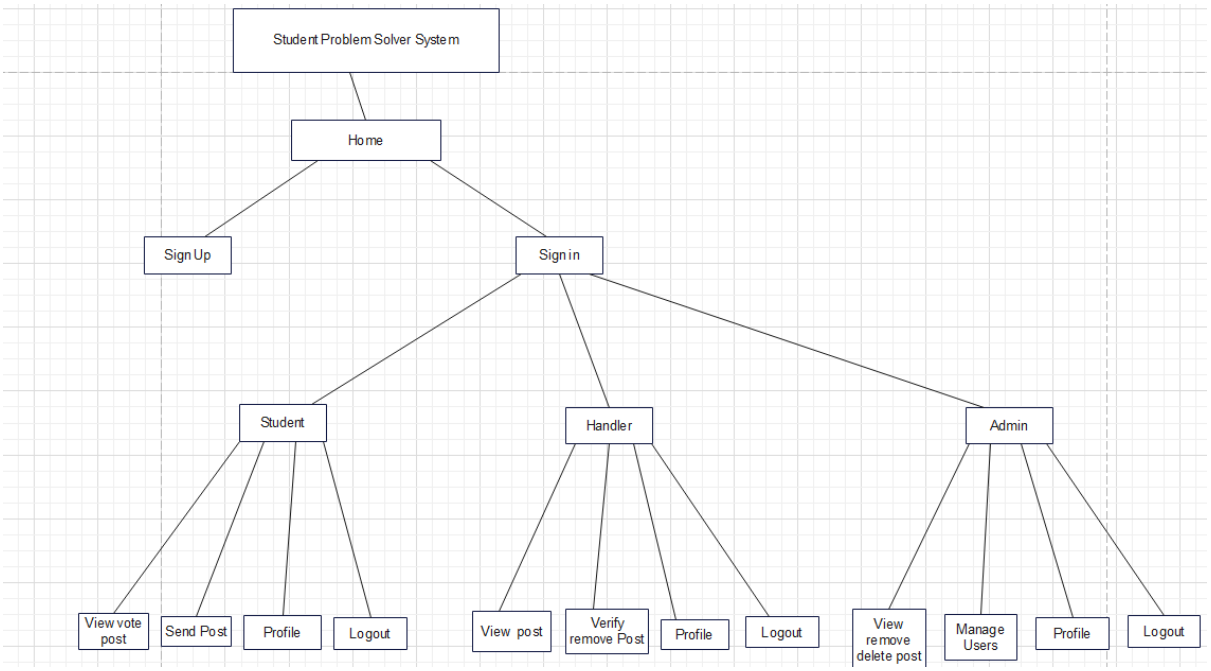


Figure 4.2: Module diagram

4.3 ER Diagram

An Entity-Relationship (ER) diagram is a visual representation used to model the relationships between various entities (objects, concepts, or things) within a system or database. It depicts how these entities are related to each other and helps to define the structure of the data and the way data is stored and retrieved. ER diagrams use symbols such as rectangles for entities, diamonds for relationships, and lines to connect them, allowing for a clear visualization of data relationships and dependencies.

Symbols:

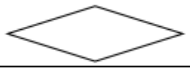
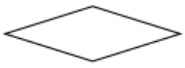
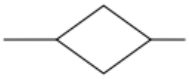
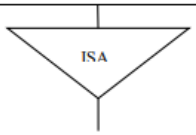
Name	Symbol	Description
Rectangle		Entity Set
Double Rectangle		Weak entity set
Eclipse		Attribute
Double Eclipse		Multivalued Attribute
Dotted Eclipse		Derived Attribute
Diamond		Relationship set
Double Diamond		Multi Valued attributes
Line		Links attribute to entity set or entity set to a relationship set
Double Line		Represents Total participation of entity
Many to many		Many to many relationship
Many to One		Many to One Relationship
One to One		One to One
IS A		IS A(Specialization or Generalization)

Table 4.3.1: Entity Relationship Diagram symbols

4.3.1 Entity sets

1.Students

This Entity represent the students in the college.

Attributes:

1.Control id(int)-pk

Represent the control id of the student

2.roll no(varchar)

Represent the roll-no of the student

3.email(varchar)

Represent the email of the student

4.registered_no(int)

It Represent whether a student is registered or not.

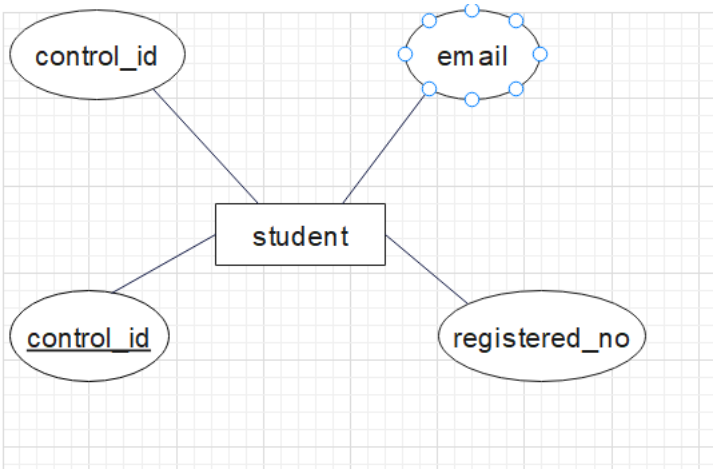


Figure 4.3.1.1: Student Entity

2.Stud_Account

This Entity represent the students who have registered to the website.

Attributes:

1.Stu_username(varchar)-ck,pk

Represent the username of the student who registered.

2.password(varchar)

Represent the password of the student who registered.

3.public_username(varchar)

Represent the public username of the student who registered.

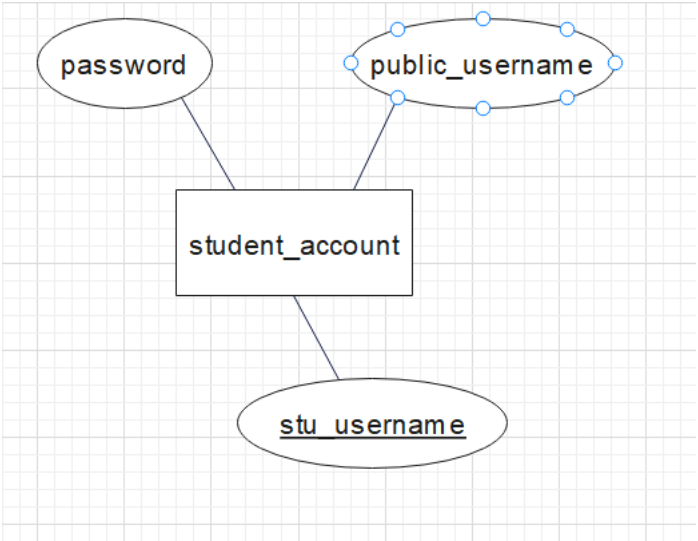


Figure 4.3.1.2: Student_account Entity

3.Post

This entity represents the Post that is posted by the student

Attributes:

1. post_id(int)-pk

Represent the an unique id given to each post

2.post_votes(int)

Represent the number of votes given to a post

3.post_status(varchar)

Represent the status of a post. Whether it is to be verified, verified, or removed.

4.post_title(varchar)

Represent the title given to a post.

5.post_desc(varchar)

Represent the description given to a post.

6.post_date(date)

Represent the date at which the post was send to be verified by the handler.

7.verified_date(date)

Represent the date at which the post was verified by the handler. And date at which it is available to the student to be viewed publicly and voted.

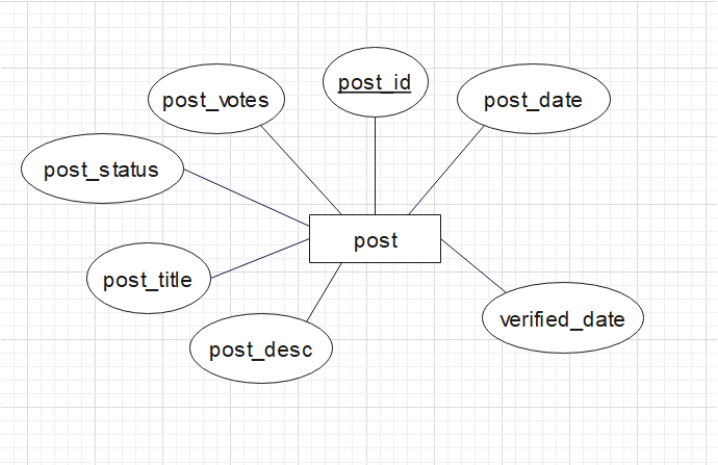


Figure 4.3.1.3: Post Entity

4.3.2 Relationship Sets

1.stud_account send post

Mapping constraint: many to many

Participation: partial (from student) total(from post)

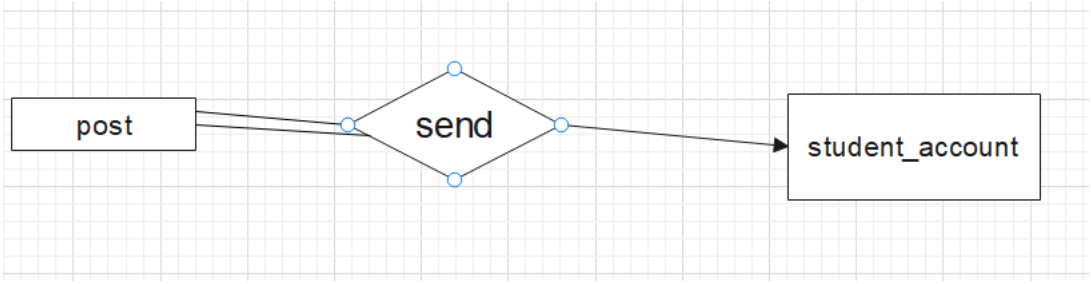


Figure 4.3.2.1: student_account send post

2.student create stud_account

Mapping constraint: one to one

Participation: partial (from student) total(from stud_account)

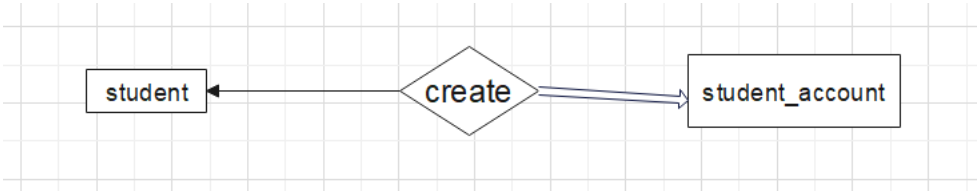


Figure 4.3.2.2: student create student_account

3.student_account votes post

Mapping constraint: many to many

Participation: partial (from student) partial(from post)

Vote have its own attribute called vote_id

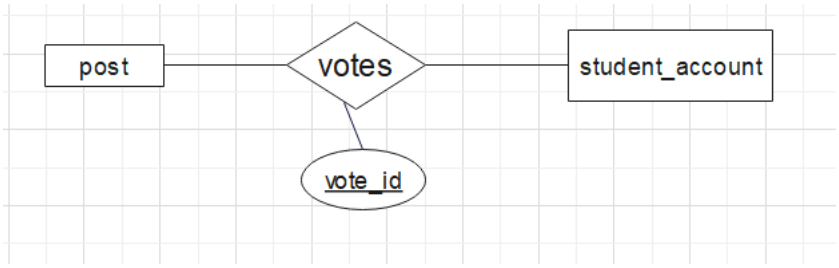


Figure 4.3.2.3: student_account votes post

4.3.3 ER diagram

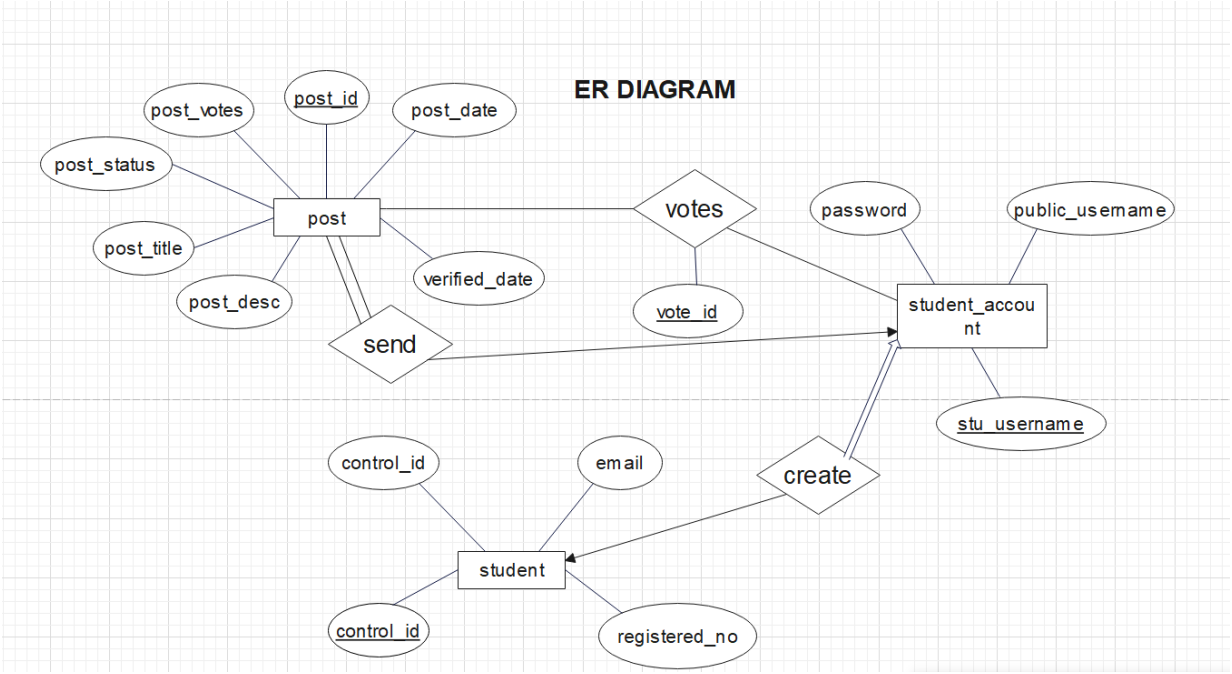


Figure 4.3.3.1: Entity Relationship Diagram

4.4 Schema Diagram

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

Symbols:

Name	Symbol	Description
Table		A table is a collection of related table held in table format within a database.
Relational		In a relational database system, a on-to-one table relationship links two tables based on a Primary Key column in the child which is also a Foreign Key referencing the Primary Key of the parent table row. Therefore, we can say that the child table share the Primary Key with the parent table.

Table 4.4.1: Schema Diagram Symbols

Schema Diagram:

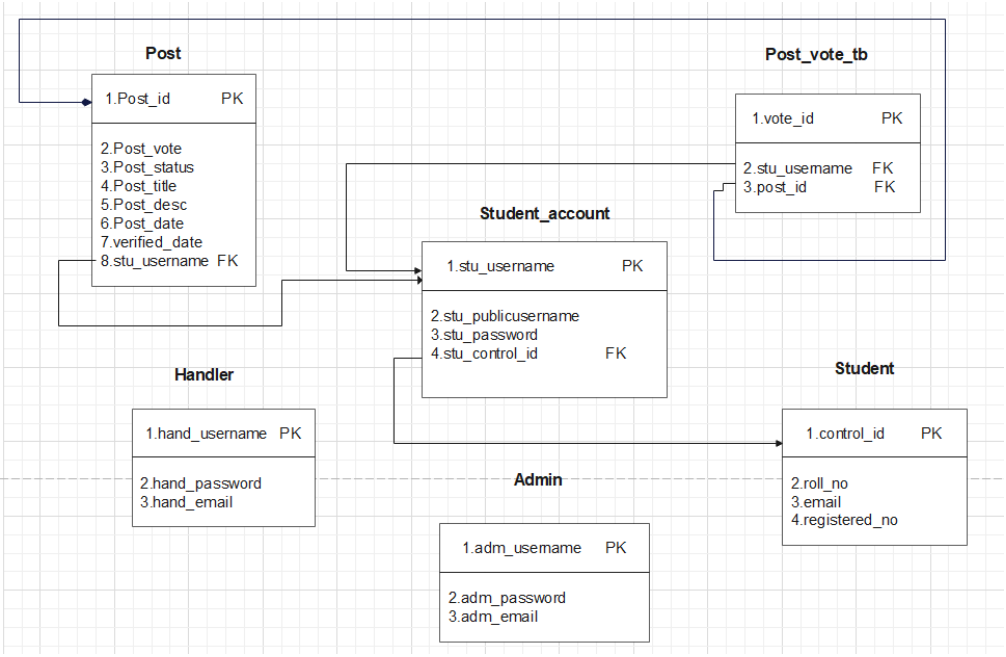


Figure 4.4.1: Schema Diagram

4.5 Data Flow Diagram (DFD)

A data-flow diagram is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

Symbols:

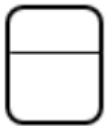
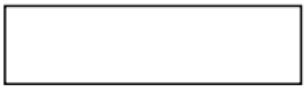
Name	Symbol	Description
Process		A process transforms incoming data flow into outgoing data flow.
Data Store		Data stores are repositories of data in the system.
Data Flow		Data flows are pipelines through which packets of information flow. Label the arrows with the name of the data that moves through it.
External Entity		External entities are objects outside the system, with which the system communicates

Table 4.5.1: Data Flow Diagram Symbols

Level 0(Context level DFD):

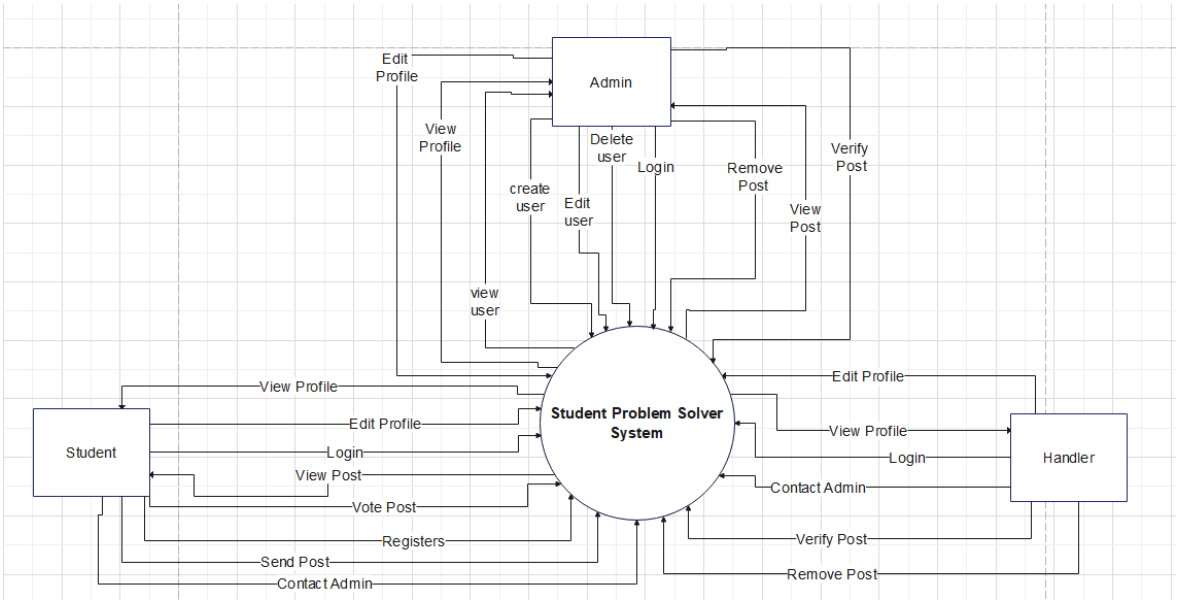


Figure 4.5.1: DFD Level 0

Level 1 DFD:

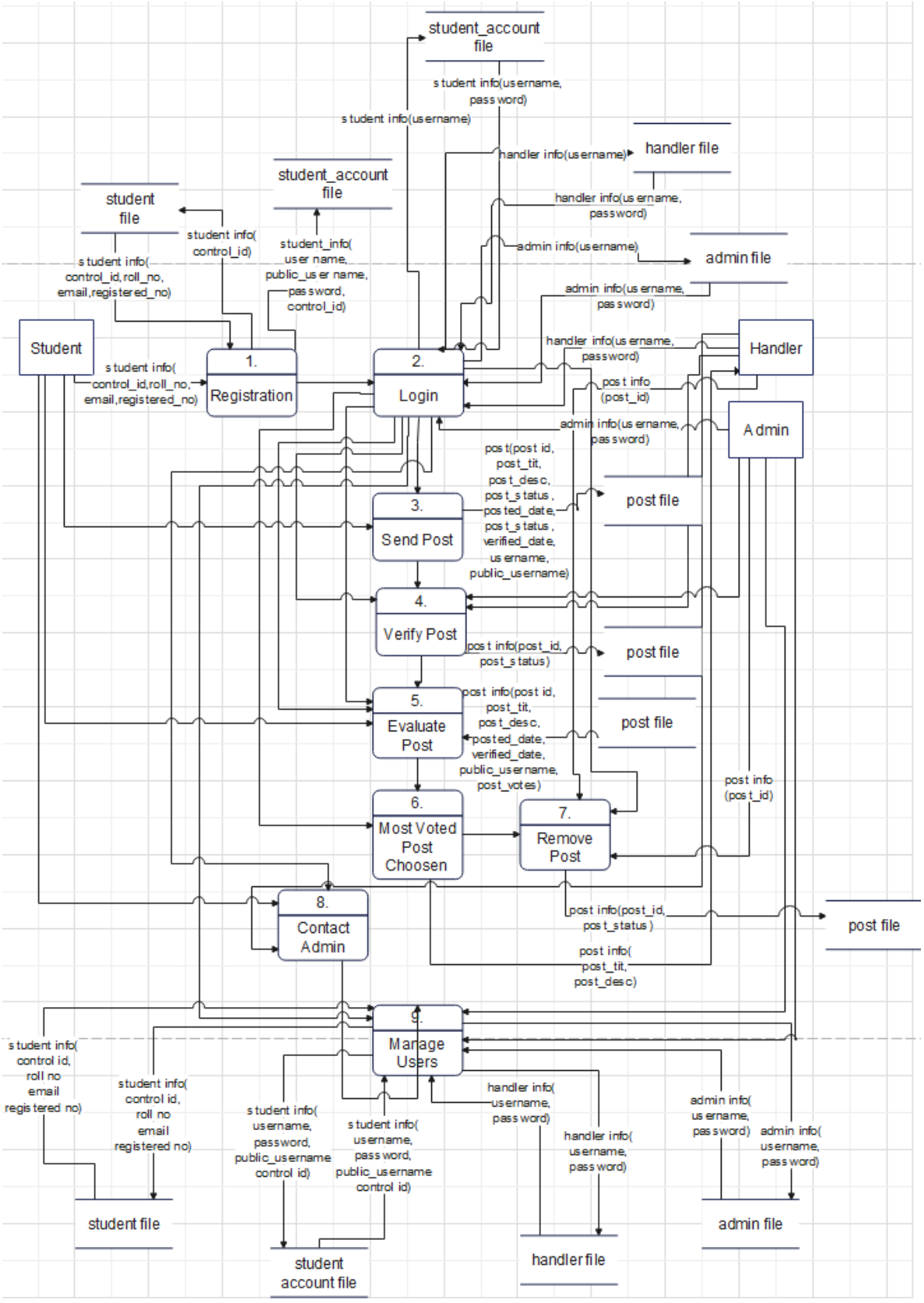


Figure 4.5.2: DFD Level 1

Level 2 DFD for Login:

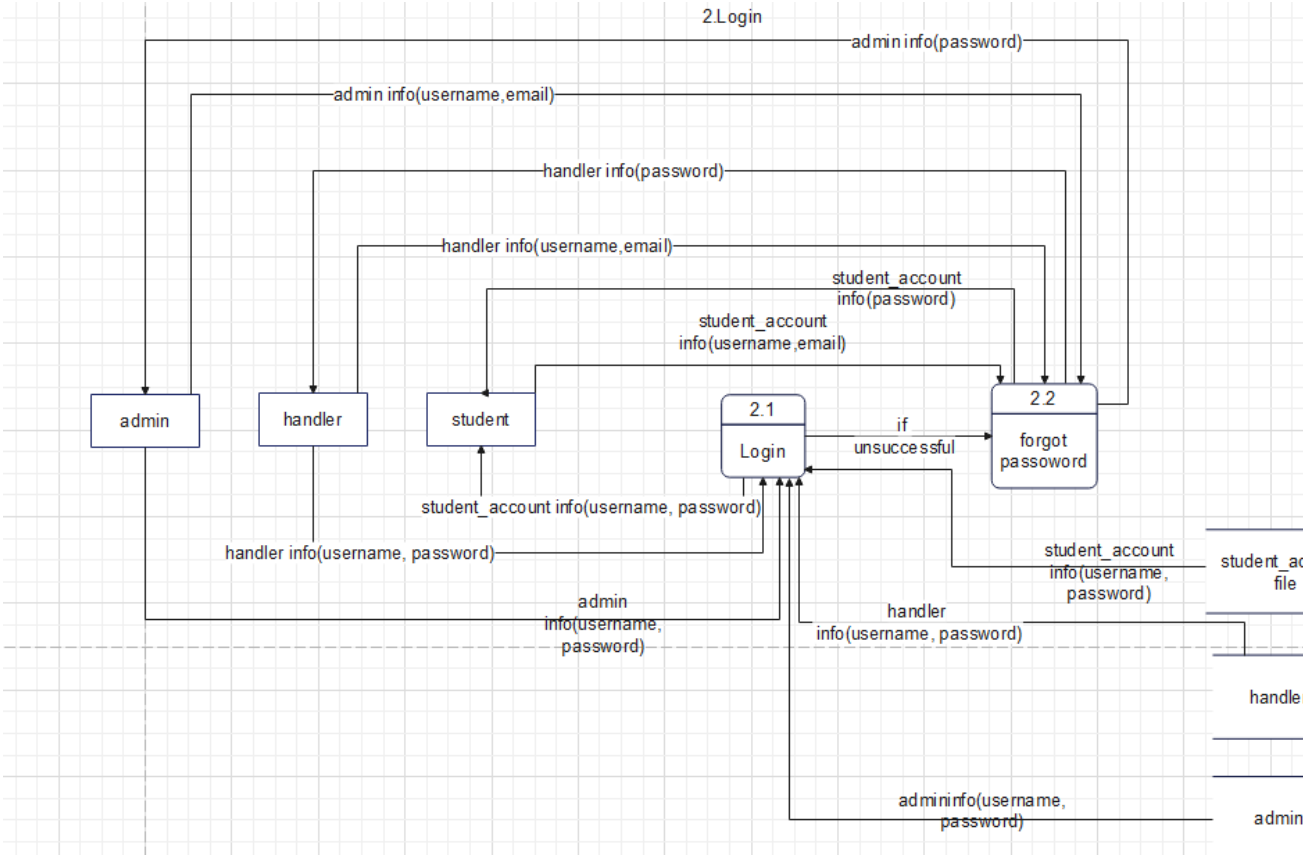


Figure 4.5.3: DFD Level 2 for Login

Level 2 DFD for Registration:

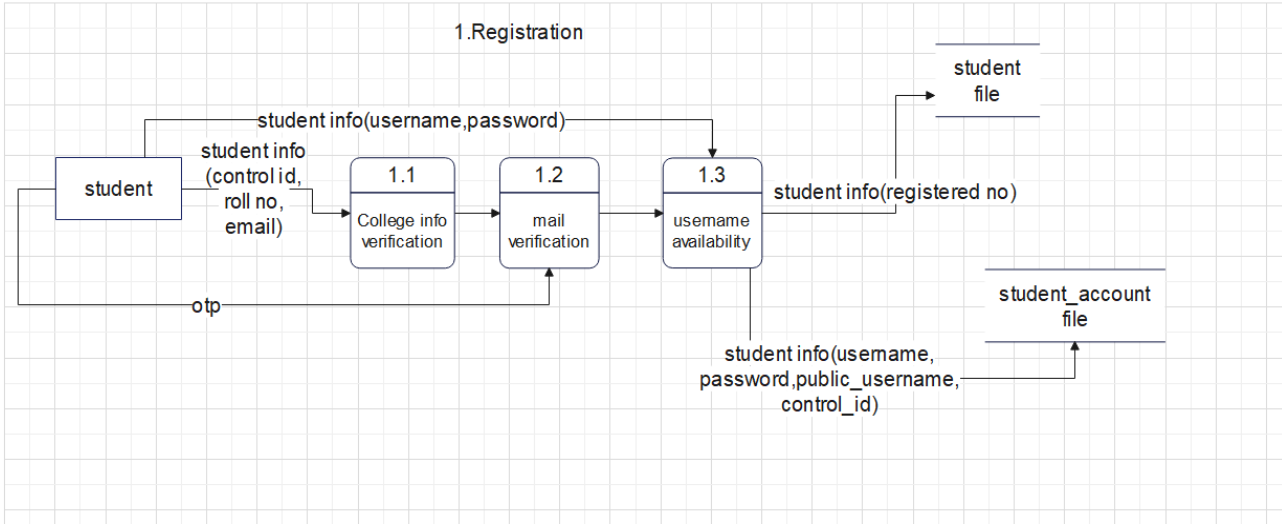


Figure 4.5.4: DFD Level 2 for Registration

Level 2 DFD for Verify Post:

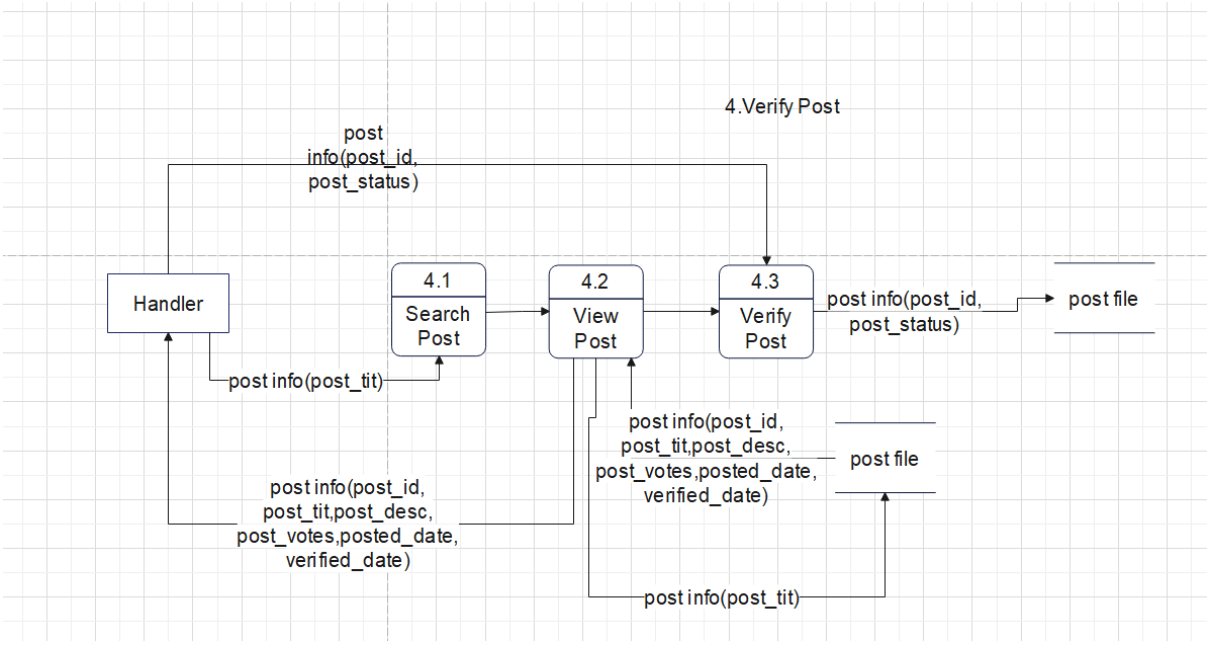


Figure 4.5.5: DFD Level 2 for Verify Post

Level 2 DFD for Evaluate Post:

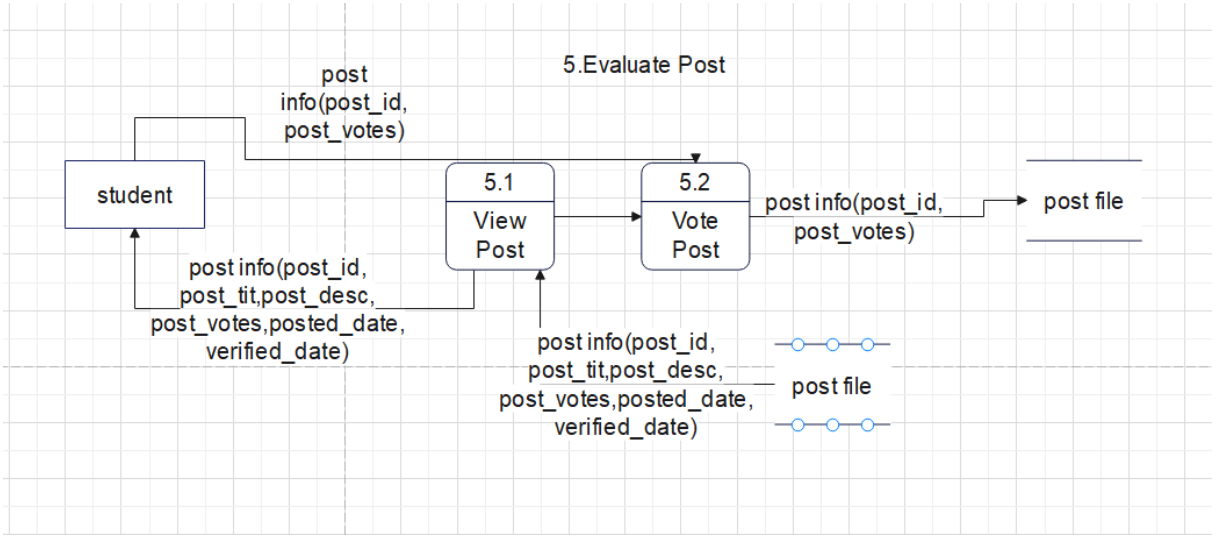


Figure 4.5.6: DFD Level 2 for Evaluate Post

Level 2 DFD for Remove Post:

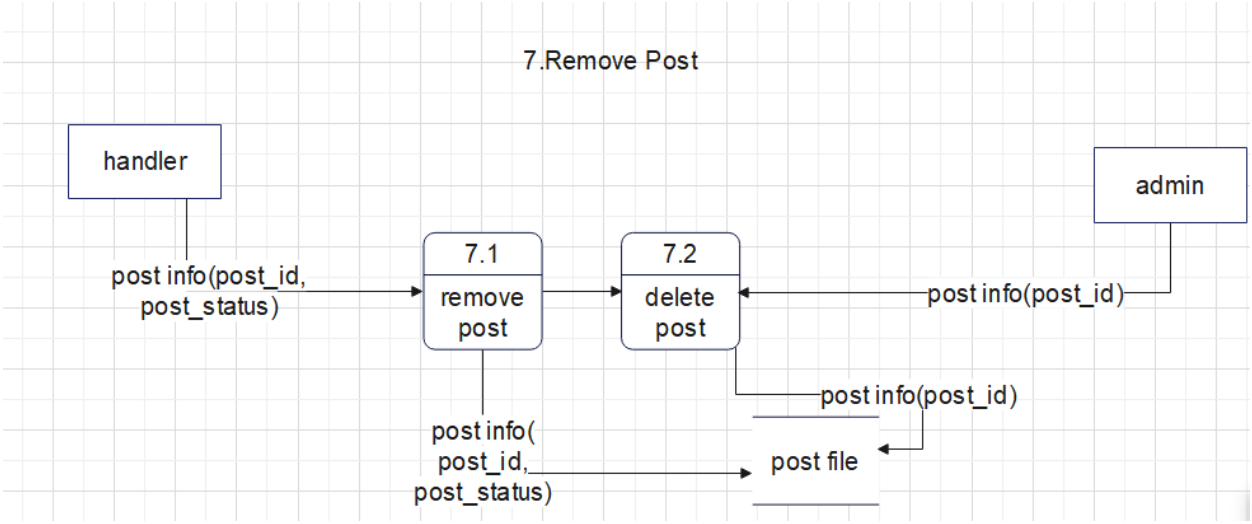


Figure 4.5.7: DFD Level 2 for Remove Post

Level 2 DFD for Manage Users:

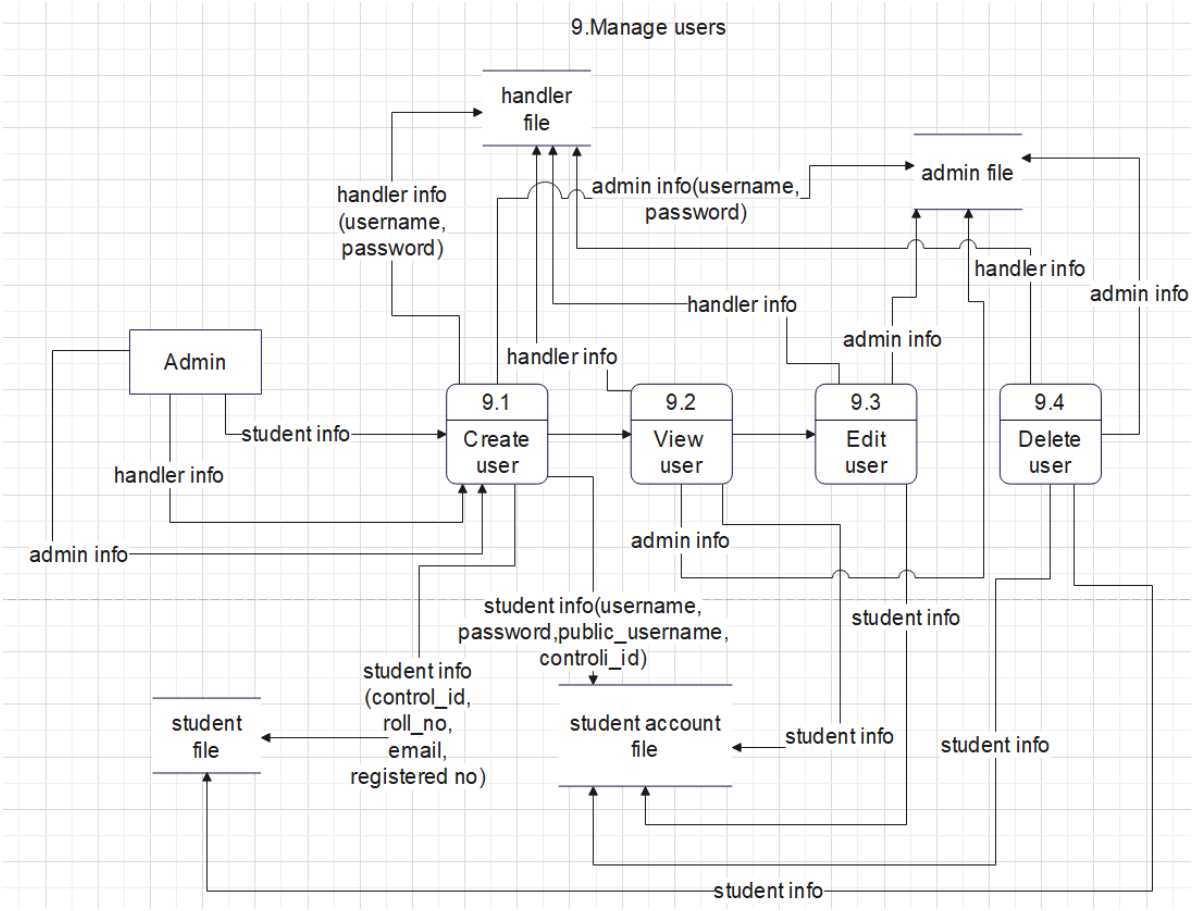


Figure 4.5.8: DFD Level 2 for Manage Users

Level 2 DFD for Manage Profile:

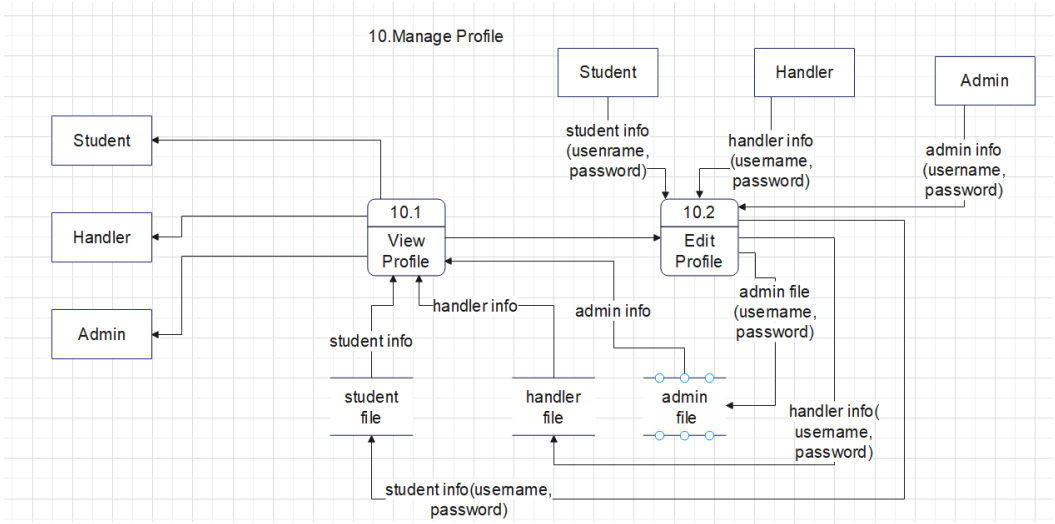


Figure 4.5.9: DFD Level 2 for Manage Profile

4.6 Class diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. 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 modelling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

Symbols:

Name	Symbol	Description
Class	Class Attributes Operations	Classes and interfaces in UML show architecture and features of the designed system.
Association	———	Represents the static relationship shared among the objects of two classes.

Table 4.6.1: Class Diagram Symbols

Class Diagram not applicable to this system

4.7 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

Symbols:

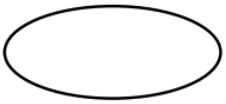
Name	Symbol	Reference
Actor		Actor represents a user or another system that will interact with the system you are modelling.
Use case		A use case is an external view of the system that represents some action the user might perform in order to complete a task.

Table 4.7.1: Use Case Diagram Symbols

4.7.1 Diagram

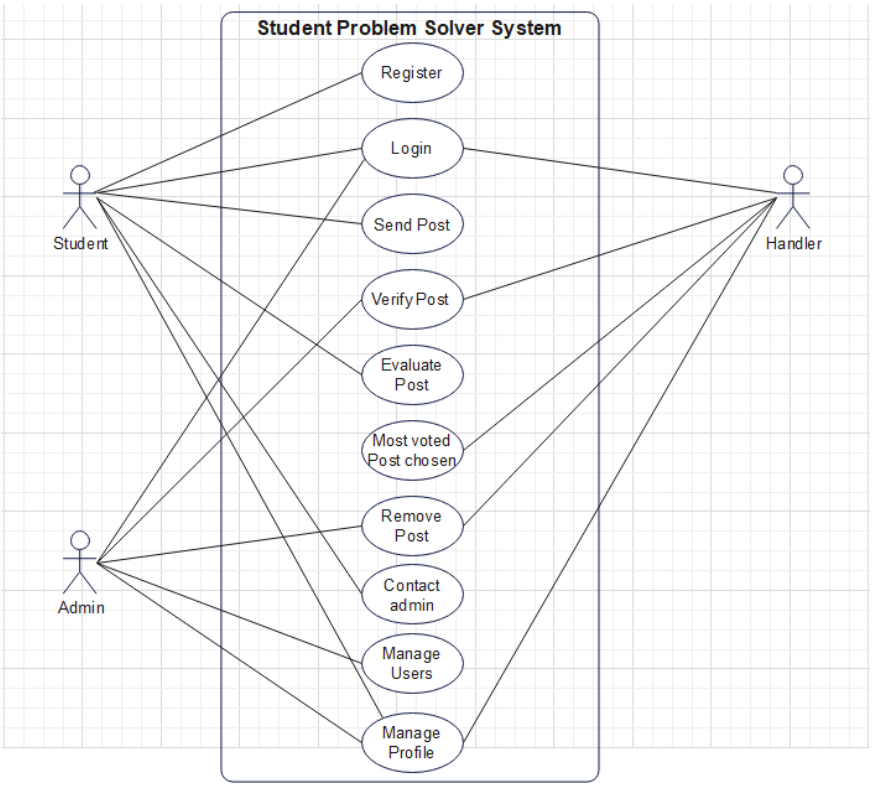


Figure 4.7.1: Use Case Diagram

4.7.2 Description

1.Use Case: Registration

- **Description**: The user(student) needs to register their account. To register, users will be given a form webpage in which they will have to fill in their personal details like control id, roll no, email, username, and password.
- **Actors**: Student
- **Pre-Condition**: Users(students) need to fill all the details asked in the form in the correct format.
- **Exception**: If the format of any detail is incorrect or any required field is kept empty, registration will not be successful. The control id, roll no, email should be matching to what was given to college during and after the admission process. If any mistake occurs it will lead to improper filling of details.
- **Post-Condition**: Registered successfully. Can proceed to login.

2.Use Case: Login

- **Description**: This use case describes the login process where the user needs to login to the system after creating an account, with username and password.
- **Actors**: Student, Handler, Admin
- **Pre-Condition**: The student/admin/handler needs to provide the correct login and password to login to the system.
- **Exception**: If any detail is not correct, the student/admin/handler cannot login to the system and must try again.
- **Post-Condition**: Provided all the details are correct, the student/admin/handler will get logged in to the system, to their respective account.

3.Use Case: Send Post

- **Description**: This use case describes the Post sending process by a student
- **Actors**: Student
- **Pre-Condition**: The student must be logged into the system with correct details.
- **Exception**: If login details are incorrect, the student cannot login to the system and send a post. And more than the 3 posts cannot be sent per week unless removed or deleted.
- **Post-Condition**: The post is sent to the handler for verification before being posted or removed.

4.Use Case: Verify Post

- **Description**: This use case describes the Post being verified by a Handler and in case of some issues admin too.
- **Actors**: Handler, Admin
- **Pre-Condition**: The Handler/ Admin must be logged into the system with correct details and posts must be sent by students to be verified.
- **Exception**: If login details are incorrect, the Handler/ Admin cannot login to the system and verify the post.
- **Post-Condition**: The post is verified and is posted for voting for students or is removed.

5.Use Case: Evaluate Post

- **Description**: This use case describes the Post which was verified, are being evaluated by students such that they view and vote the posts.
- **Actors**: Student

- **Pre-Condition**: The student must be logged into the system with correct details and post must be verified to be available for voting.
- **Exception**: If login details are incorrect, the student cannot login to the system and view and vote post.
- **Post-Condition**: The posts are voted by the students and most voted posts are sorted at the top which would be selected by the handler accordingly.

6.Use Case: Most voted post chosen

- **Description**: This use case describes the process in which the Posts with most votes are prioritised and chosen by the handler to be solved.
- **Actors**: Handler
- **Pre-Condition**: The Handler must be logged into the system with correct details and post must be present to be voted by students.
- **Exception**: If login details are incorrect, the Handler cannot login to the system and select the most voted posts.
- **Post-Condition**: The most voted post is selected and the post is removed as its purpose is completed.

7.Use Case: Remove post

- **Description**: This use case describes the Post being removed by Handler and in case of issue admin can do this functionality.
- **Actors**: Handler, Admin
- **Pre-Condition**: The Handler/ Admin must be logged into the system with correct details and post must be present to be removed.
- **Exception**: If login details are incorrect, the Handler/ Admin cannot login to the system and remove the post.
- **Post-Condition**: The post is removed and can be deleted later by admin.

8.Use Case: Contact admin

- **Description**: This use case describes the process of a student or handler contacting the admin through the contact us page in case of any issue.
- **Actors**: Student, Handler

- **Pre-Condition**: The Student/ Handler must be logged into the system with correct details.
- **Exception**: If login details are incorrect, the Student/ Handler cannot login to the system and contact Admin.
- **Post-Condition**: The admin would respond to the request made by Student/ Handler

9.Use Case: Manage users

- **Description**: This use case describes the process of managing users(create,view,edit,delete) by the Admin
- **Actors**: Admin
- **Pre-Condition**: The Admin must be logged into the system with correct details.
- **Exception**: If login details are incorrect, the Admin cannot login to the system and manage users.
- **Post-Condition**: The changes of create, edit delete by admin would be reflected on the system.

10.Use Case: Manage profile

- **Description**: This use case describes the process of managing profile(view and edit) of Student/Handler/Admin
- **Actors**: Student, Handler, Admin
- **Pre-Condition**: The Student/Handler/Admin must be logged into the system with correct details.
- **Exception**: If login details are incorrect, the Student/Handler/Admin cannot login to the system and manage profile.
- **Post-Condition**: The changes of edit by Student/Handler/Admin would be reflected on the system.

4.8 Scenarios

1. Registration

1. The user inputs the required details (control id, roll no, email)

2. The system requests for details of the user from database

3.The system verifies these details

4.The system then sends an otp to the respective mail given and tells the user to enter otp through a modal.

5.The system authenticate the otp and proceed to take the input of username and password.

6.The system check the username availability and check for validation.

7. On successfully passing the check constraints, the details are stored in the database and if successfully registered redirected to the home page

2. Login

1. The user inputs the required details (username and password)

2. The system requests for details of the user from database

3.The system verifies these details

4.If verification is successful, and the user is redirected to the home page

3. Evaluate Post-View and Vote Post

1.All posts who have its attribute post_status set to "verified " are loaded to the user in a sorted manner of highest to lowest votes from the database, after login. They can view the post posted.

2.The appropriate posts are voted.

3.When a post is voted on. It is sorted again according to the current order of the post's vote.

4.The post vote count is incremented by one and updated in the database.

5.On refreshing it is loaded with the new order with the updated post vote.

4. Send Post

1.After selecting the post tab, the user is redirected to the posting page.

2.The user can fill the details of the post and send the post.

3.The details are sent and stored into the database and an attribute of the post, post_status is set to "to be verified"

4. A message "Post will be posted after verification " is shown to the user. A maximum of three votes are allowed to be posted by a user per week unless removed.

5. Manage Profile

1.After selecting the profile tab, the user is redirected to the profile page.

2.Data is loaded from the database and profile details are displayed.

3.If it is edited, only the password could be changed by clicking the edit button.

4.On clicking edit, a modal appears where details could be changed.

5.After changing details by clicking on the edit of the modal, the new data is sent to the database and updated.

6.The page gets refreshed automatically and new data is visible in the profile page.

6. Contact us

1.After selecting the contact us tab, the user is redirected to the contact us page.

2.Details are to be filled by the user.

3.On submitting the form is sent to the email of admin where he can view the form details.

7. Verify Post

1.All posts are loaded to the Handler from the database, after logging in and selecting the post tab. Handler can view and verify the posts.

2.The appropriate posts are verified.

3.After verify button is clicked to a post, the attribute of the post, post_status is set to "verified" and the button text is changed to verified.

4.Admin can verify a removed post in post tab section, which has a similar functioning.

8. Remove Post

1.A remove button is present beside the verify button, which can be clicked to remove the post.

2.After clicking, a confirmation message is displayed “Are you sure you want to remove the post?” in which on confirming, the post status of the post is set to "removed" in database.

3.After 3 days of a post its post status is automatically set to "removed"

4.Admin can remove a verified post in home tab section, which has a similar functioning.

5.Further the removed post can be deleted by the admin

6.By clicking the delete button it is deleted from the database.

9. Manage Users

1.After login the admin can select the manage user tab, to redirect to the manage user tab.

2.Here four tabs are available to manage which are student, student_acc, handler and admin.

3.On clicking any of the tabs the data is loaded from the database accordingly i.e.if student_acc is selected then the student account data is loaded to admin.

4. Options(buttons) are available to manage which are- create, edit, and delete options. On clicking create or edit a modal appears where data is to be entered and then on confirming the data is updated to the database. On clicking delete the respective data is deleted from the database.

4.9 Sequence diagram

A sequence diagram in a Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams typically are associated with use case realizations in the Logical View of the system under development.

Symbols:

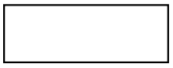
Name	Symbol	Description
Activation Box		The period when an object if performing an action
Object		An object that is created and performs action
Synchronous Message		Does the communication between different objects

Table 4.9.1: Sequence Diagram Symbols

Sequence Diagram for User Registration:

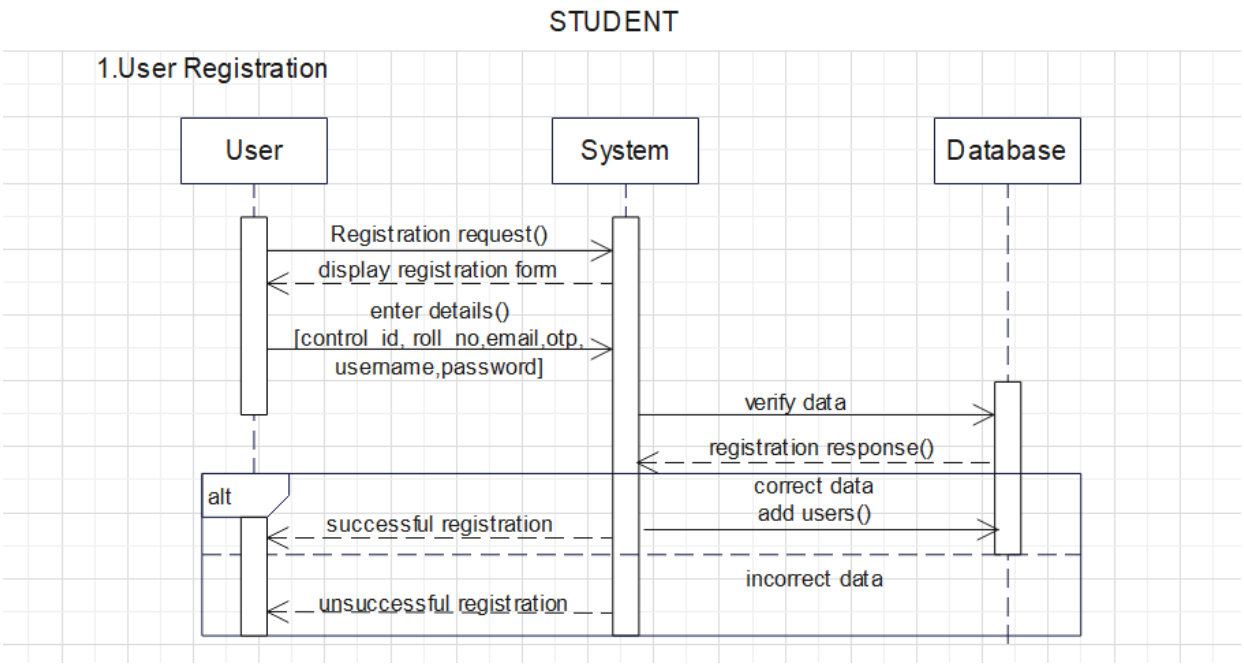


Figure 4.9.1: Sequence Diagram for User Registration

Sequence Diagram for User Login:

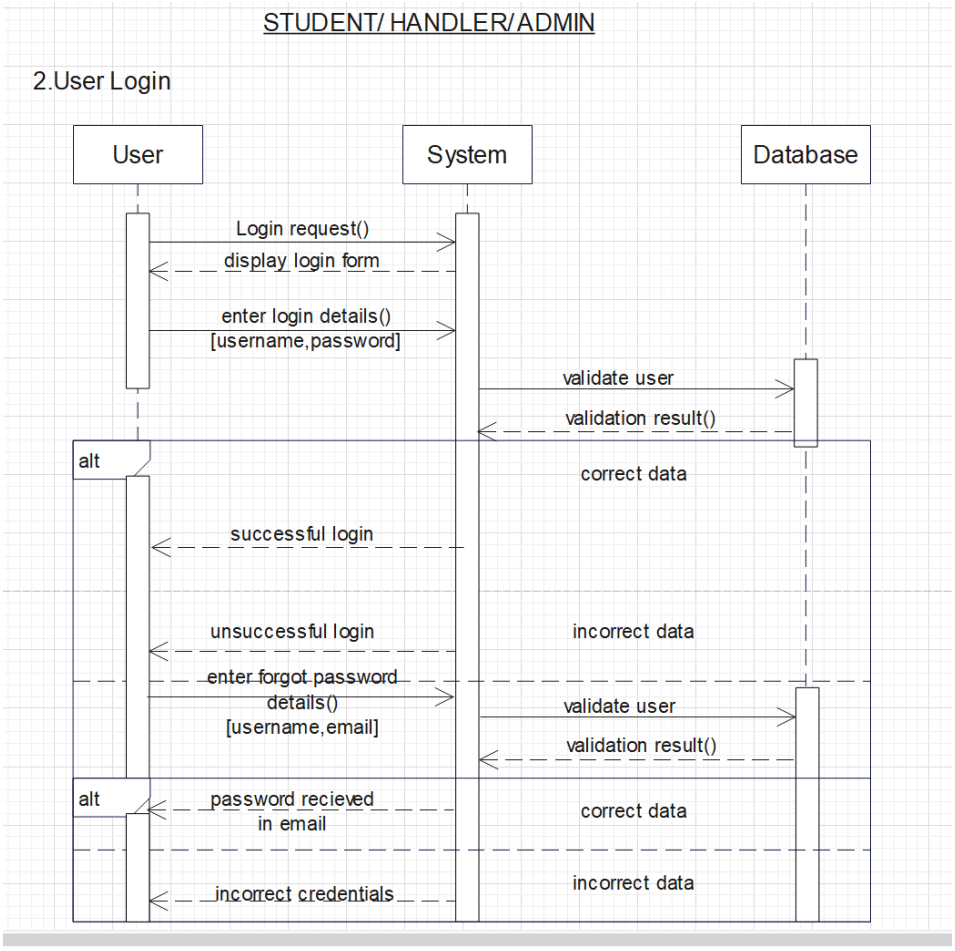


Figure 4.9.2: Sequence Diagram for User Login

Sequence Diagram for Send Post:

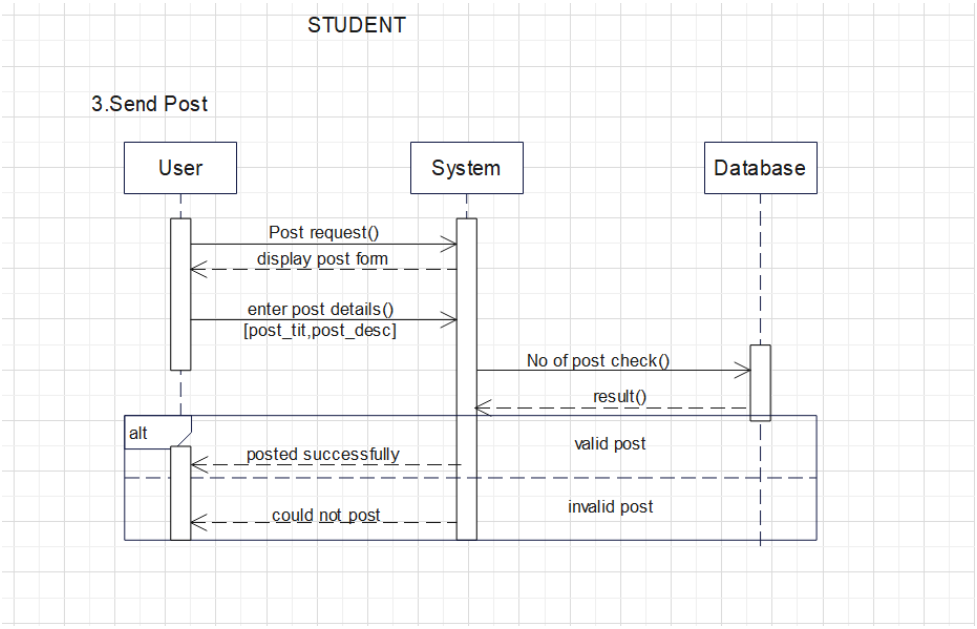


Figure 4.9.3: Sequence Diagram for Send Post

Sequence Diagram for Evaluate Post:

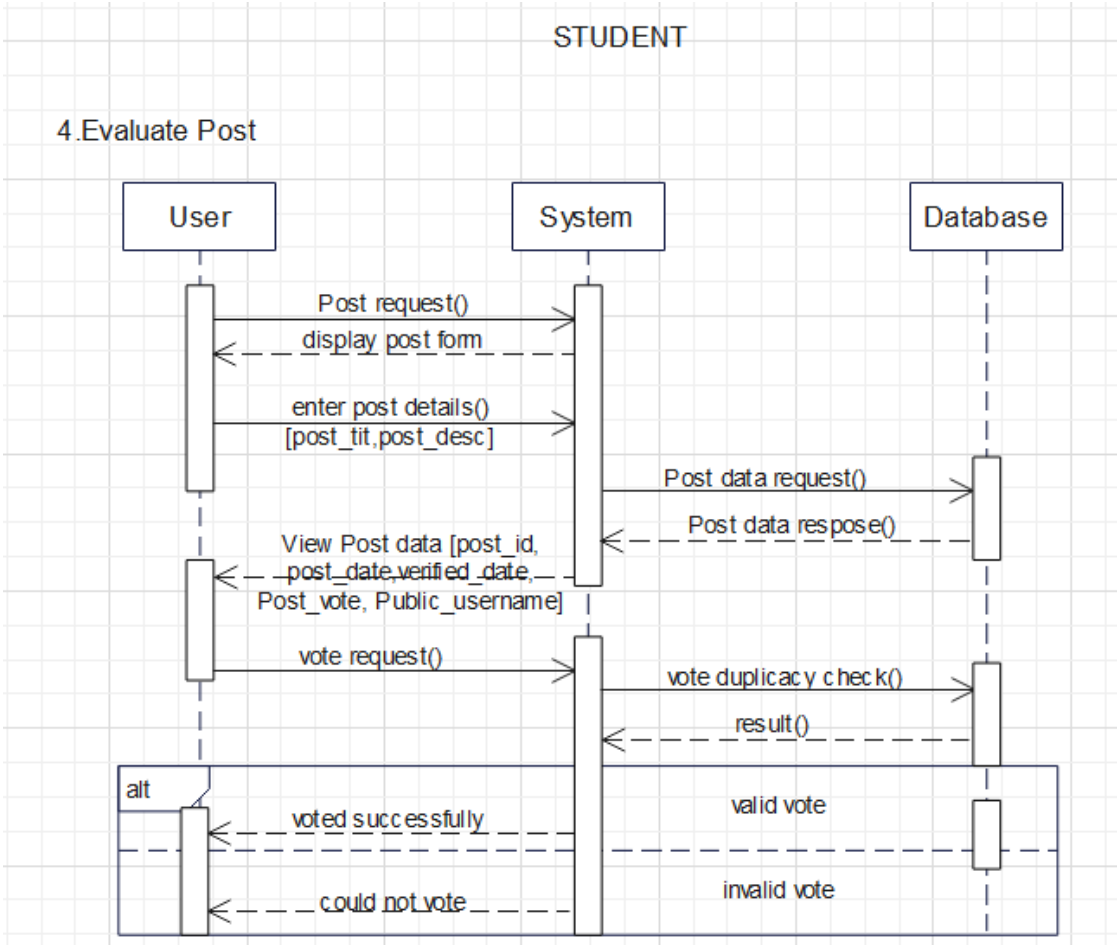


Figure 4.9.4: Sequence Diagram for Evaluate Post

Sequence Diagram for Contact Admin:

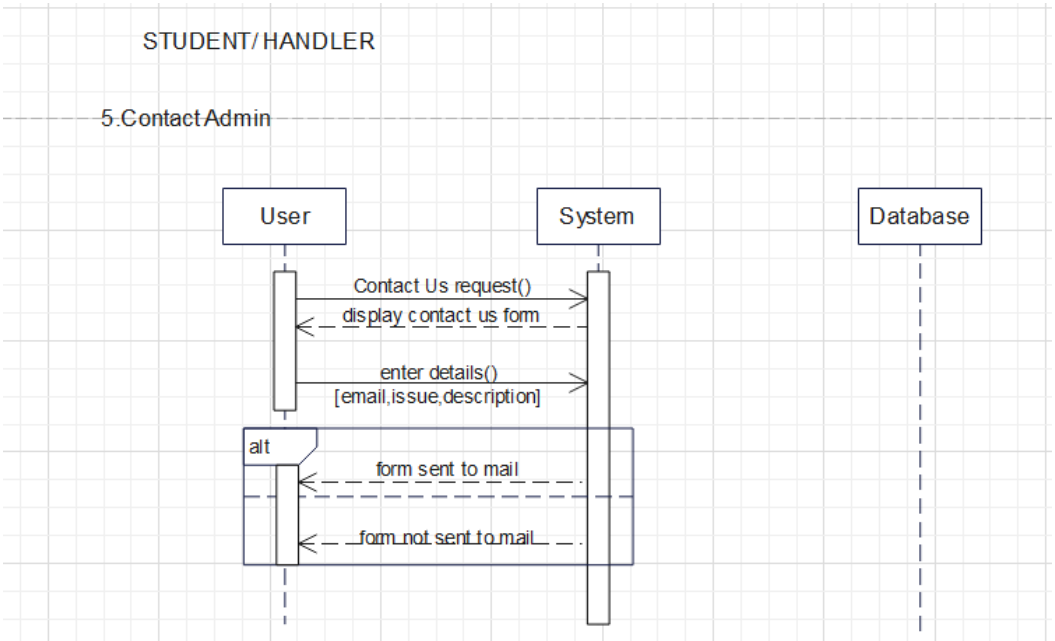


Figure 4.9.5: Sequence Diagram for Contact Admin

Sequence Diagram for Manage Profile:

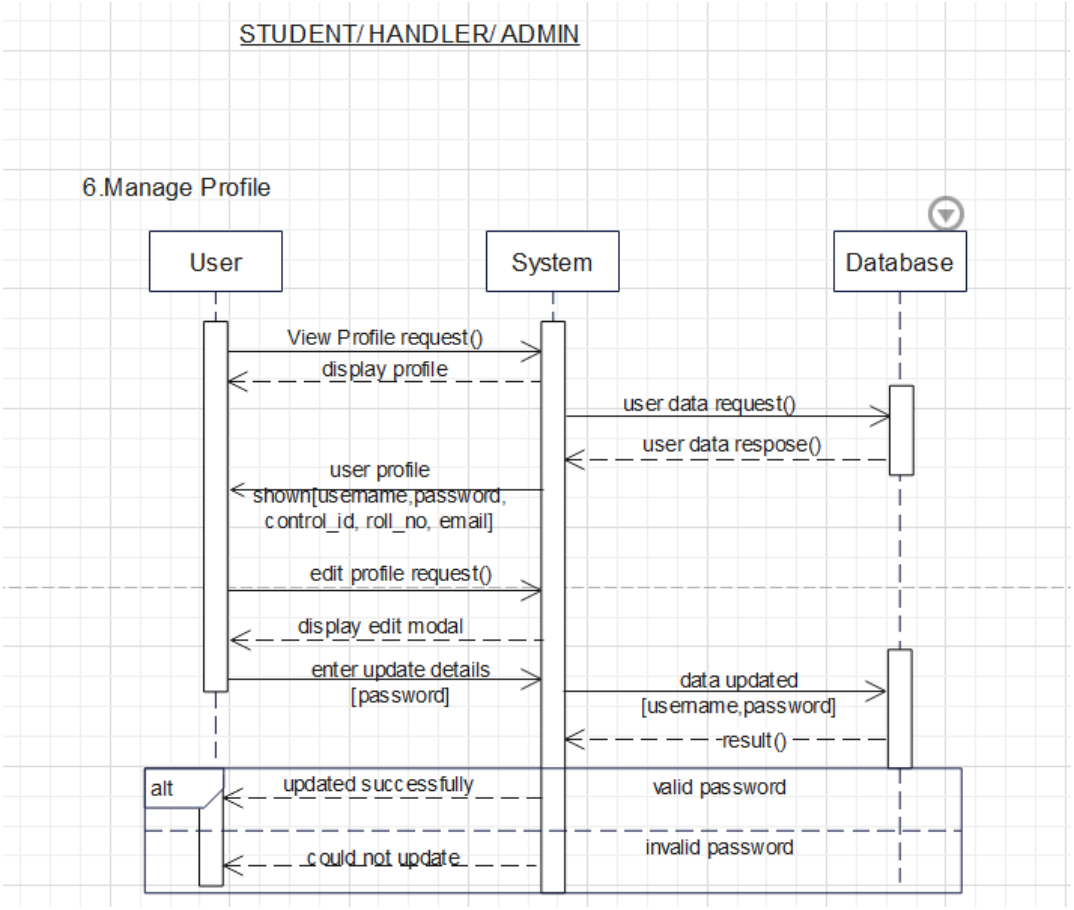


Figure 4.9.6: Sequence Diagram for Manage Profile

Sequence Diagram for Verify Post:

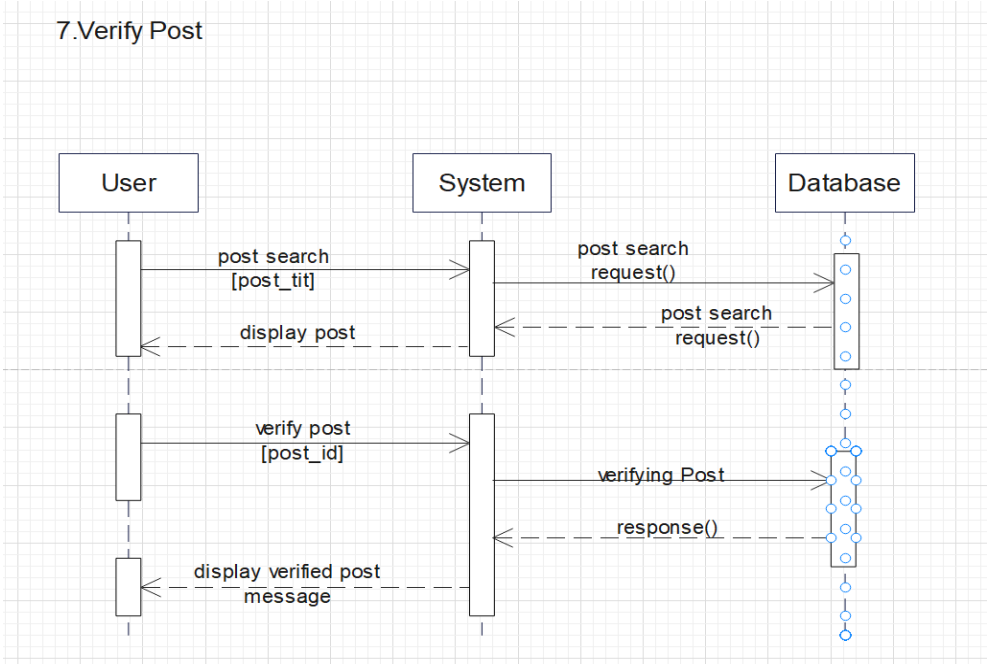


Figure 4.9.7: Sequence Diagram for Verify Post

Sequence Diagram for View Post [Most voted post chosen]:

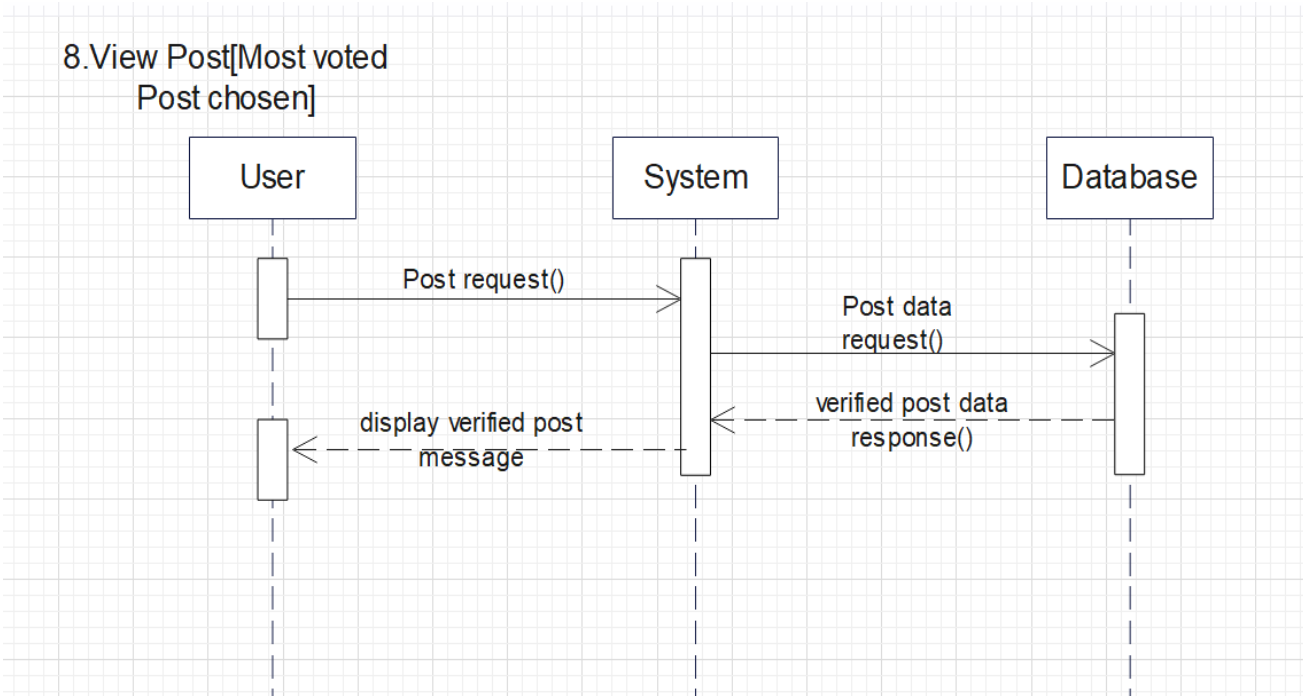


Figure 4.9.8: Sequence Diagram for View Post [Most voted post chosen]

Sequence Diagram for Remove Post:

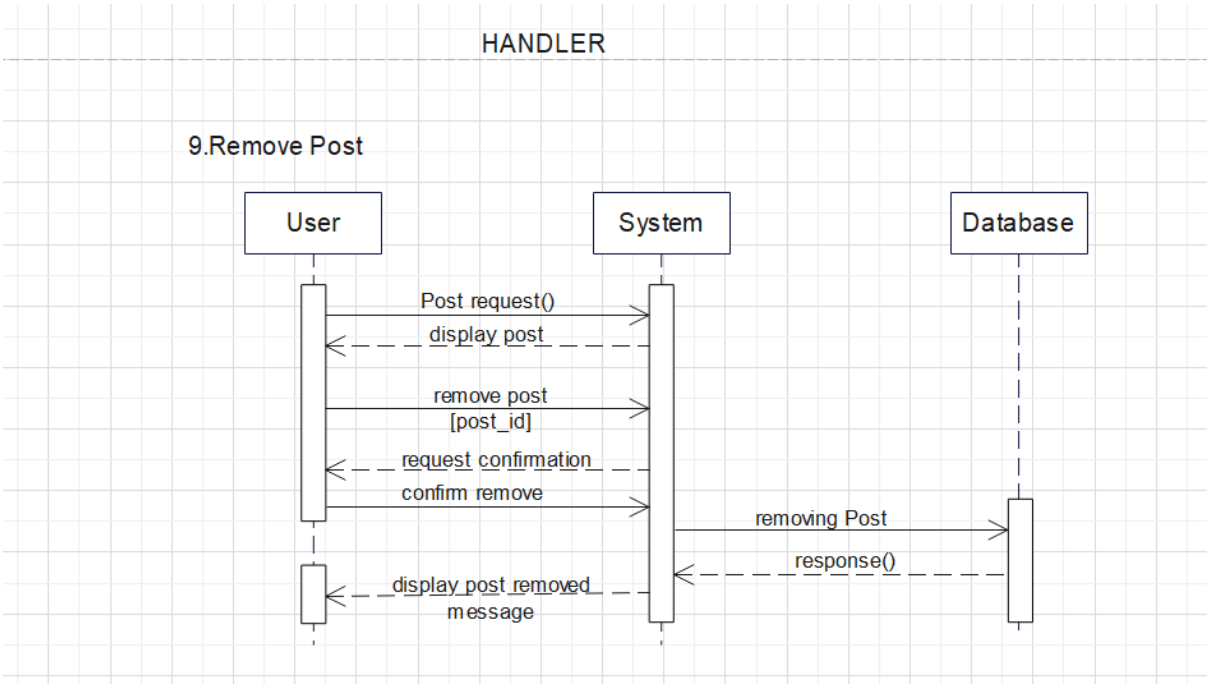


Figure 4.9.9: Sequence Diagram for Remove Post

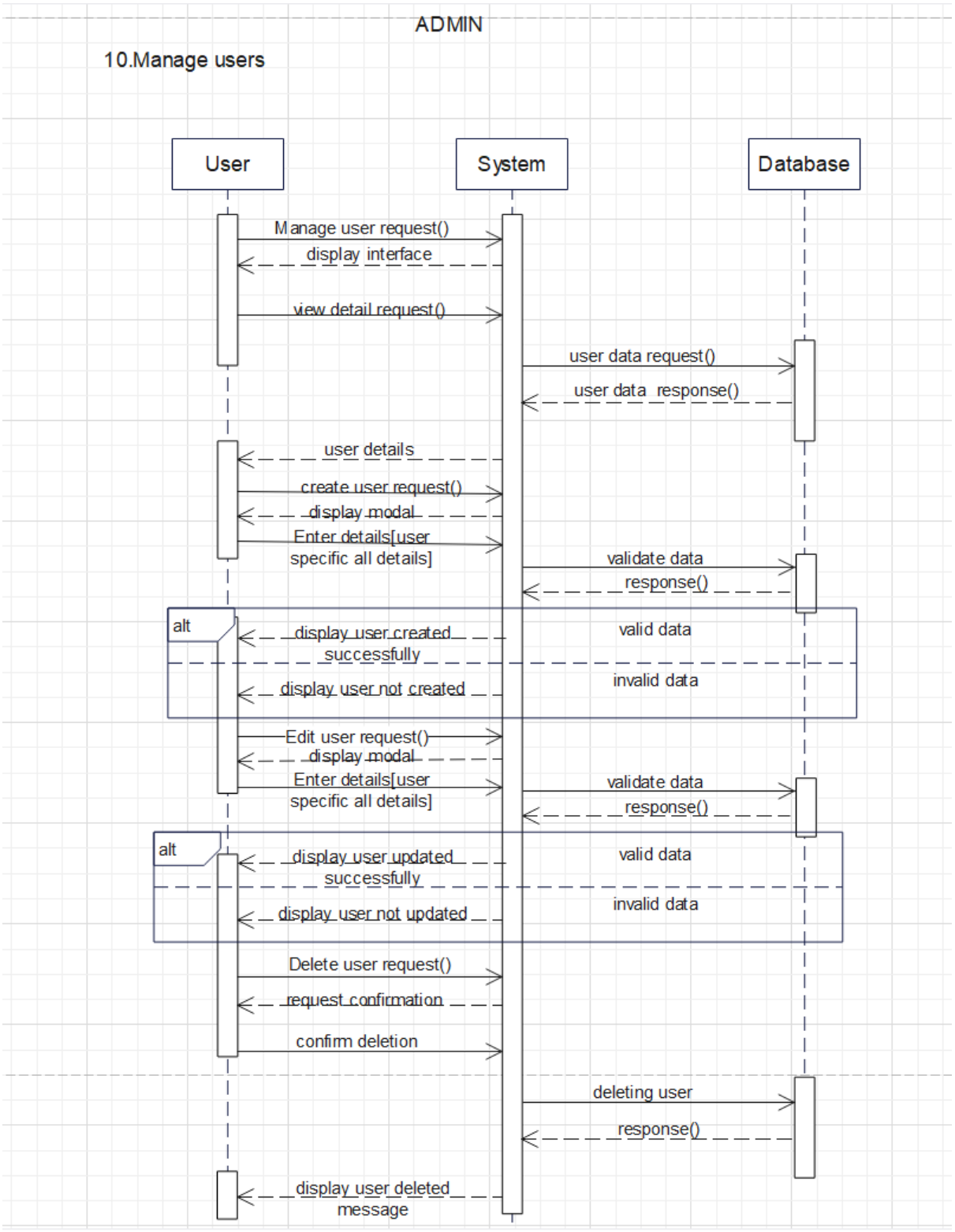


Figure 4.9.10: Sequence Diagram for Manage users

4.10 Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

Symbols:

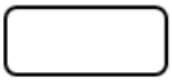
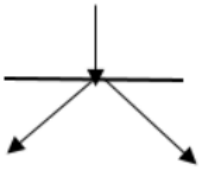
Name	Symbol	Description
Initial state		This shows the starting point or first activity of the flow.
Final state		The end of the Activity diagram, also called as a final activity.
Action		It represents the activity to be performed.
Decision		A logic where a decision is to be made is depicted by a diamond.
Synchronization		It combines or splits two activity flows.
Transition		A transition link represents control flow between nodes.

Table 4.10.1: Activity Diagram Symbols

Activity diagram for student:

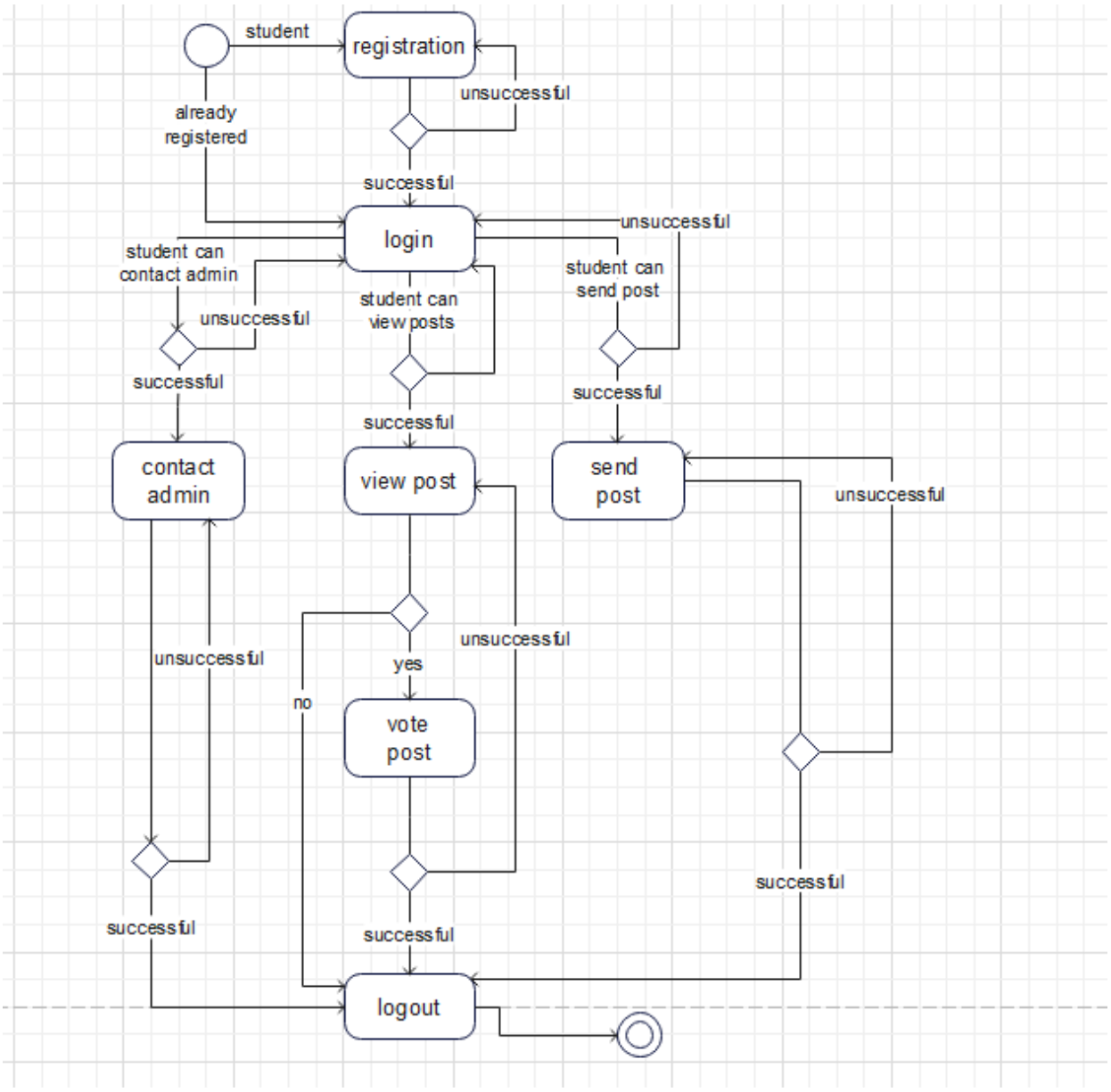


Figure 4.10.1: Activity diagram for student

Activity diagram for handler:

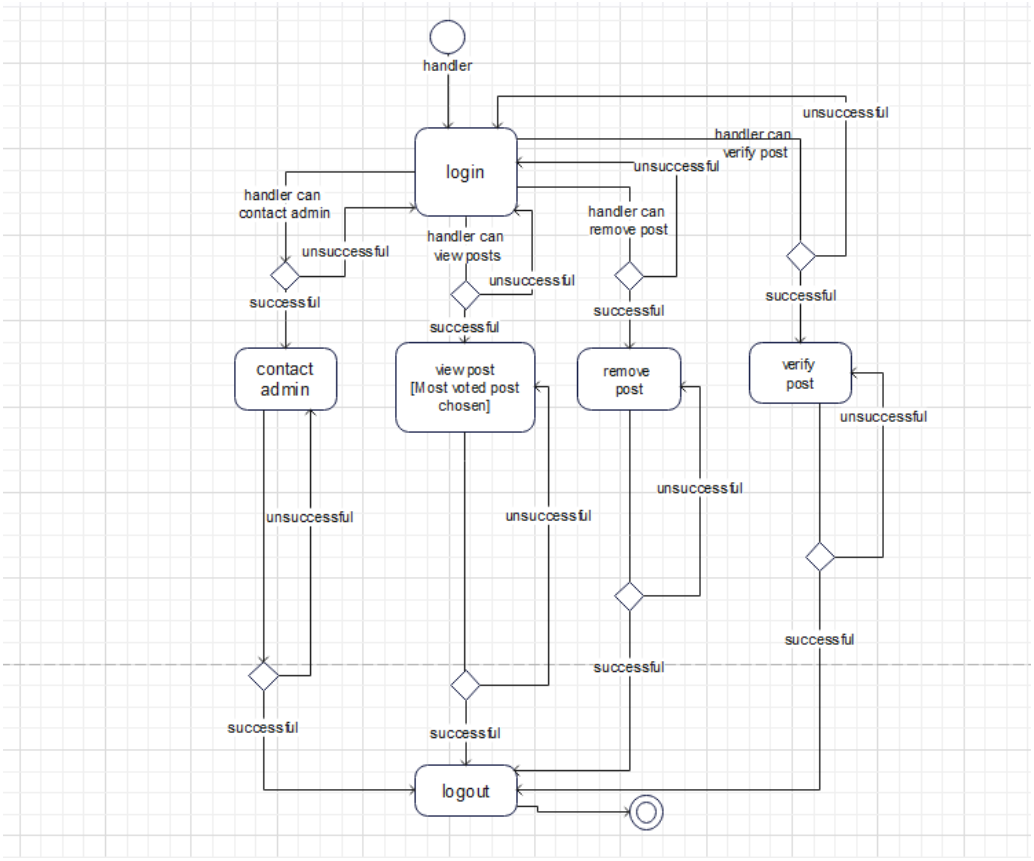


Figure 4.10.2: Activity diagram for handler

Activity diagram for admin:

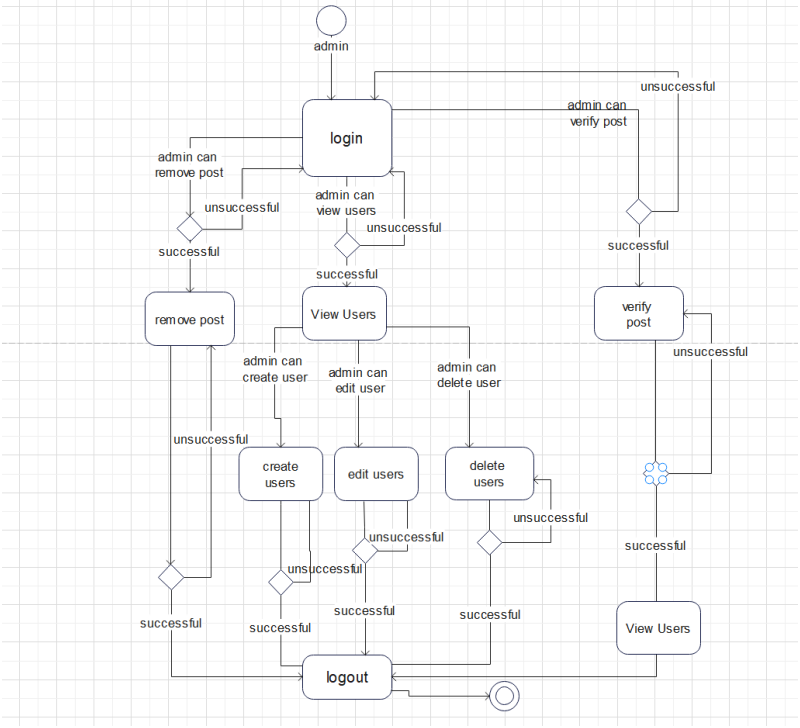


Figure 4.10.3: Activity diagram for admin

4.11 State Diagram

A state diagram is a type of diagram used in computer science and related fields to describe the behaviour of systems. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction.

Symbols:

Name	Symbol	Description
Initial State		It is the starting state
Transition		It shows the flow and what action is been performed
State		We use a rounded rectangle to represent a state. A state represents the conditions or circumstances of an object of a class at an instant of time.

Table 4.11.1: State chart Diagram Symbols

State diagram for student:

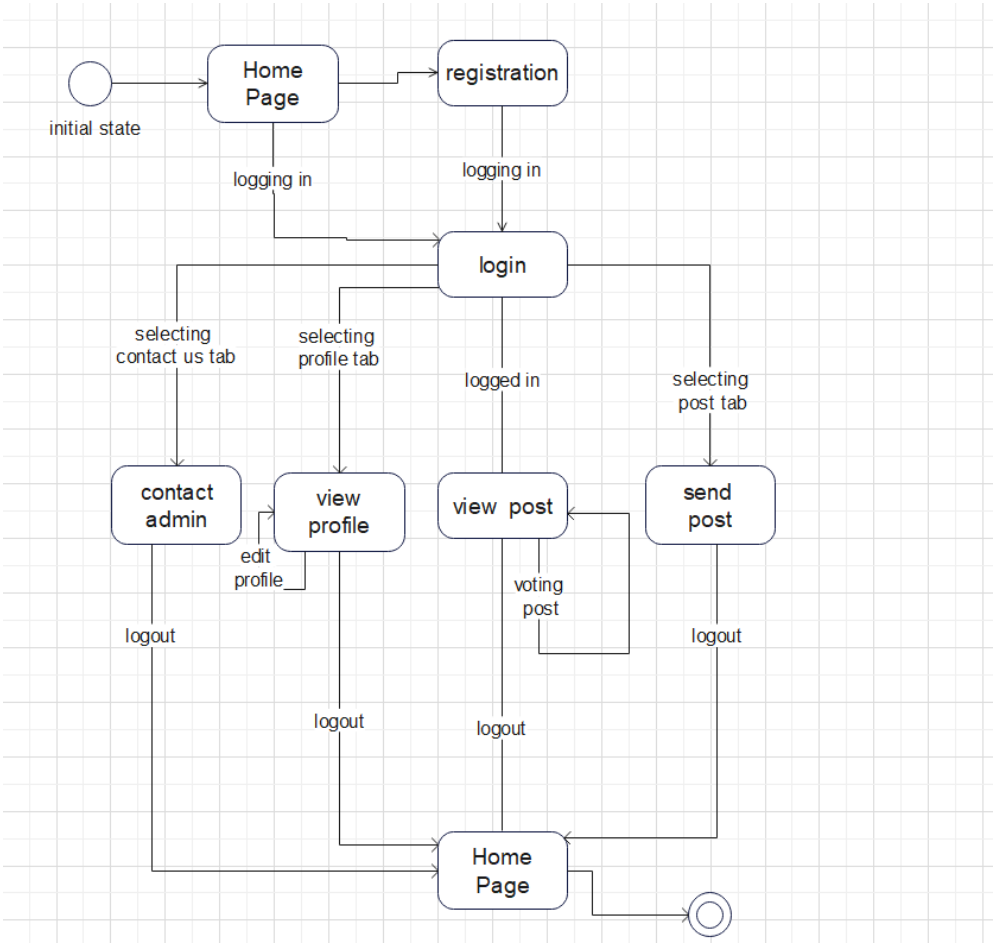


Figure 4.11.1: State diagram for student

State diagram for student:

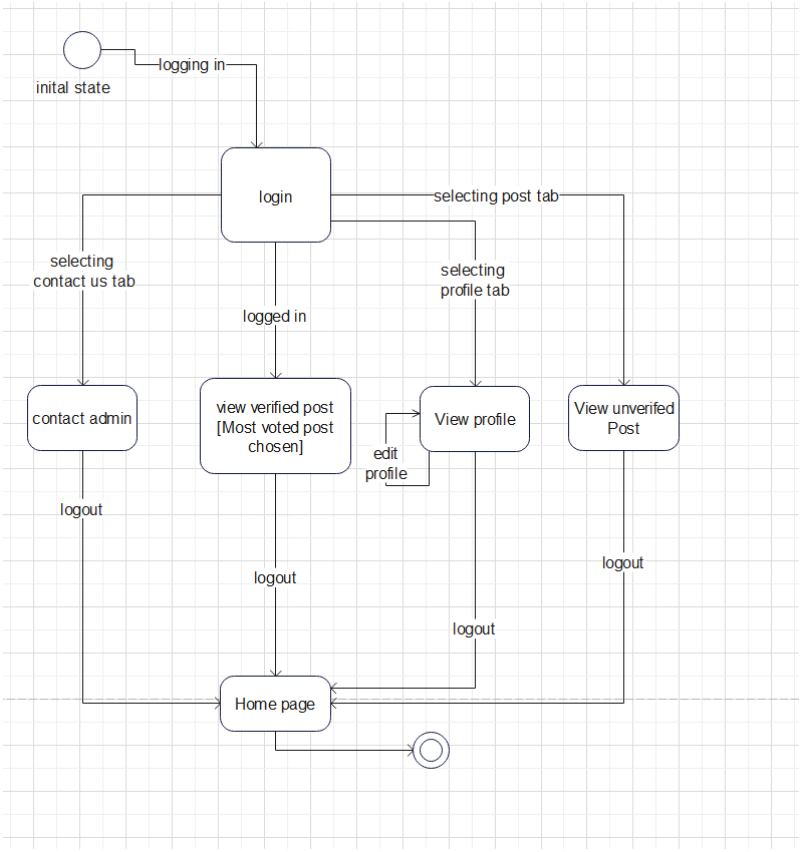


Figure 4.11.2: State diagram for handler

State diagram for student:

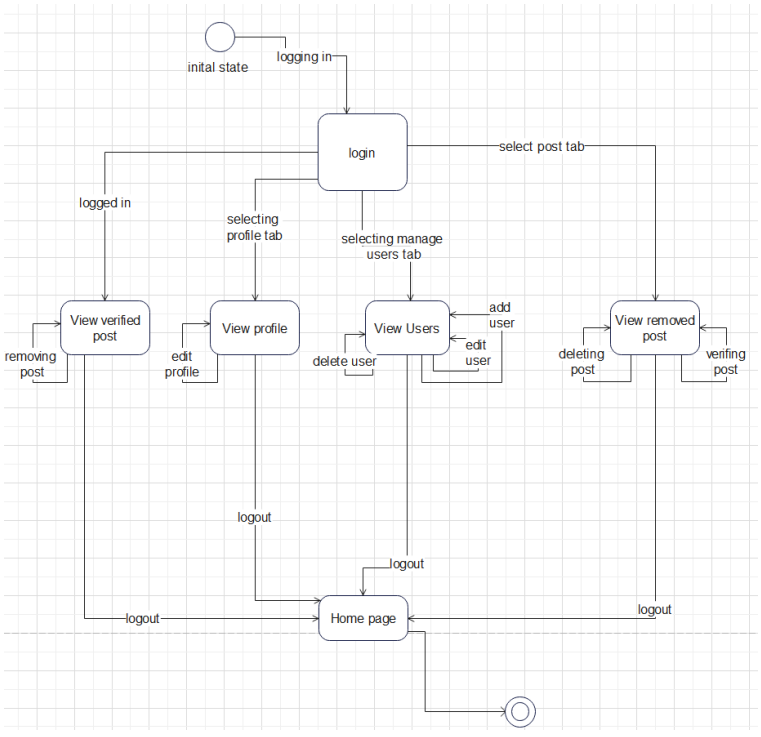


Figure 4.11.3: State diagram for admin

4.12 User Interface Design

Home Page:

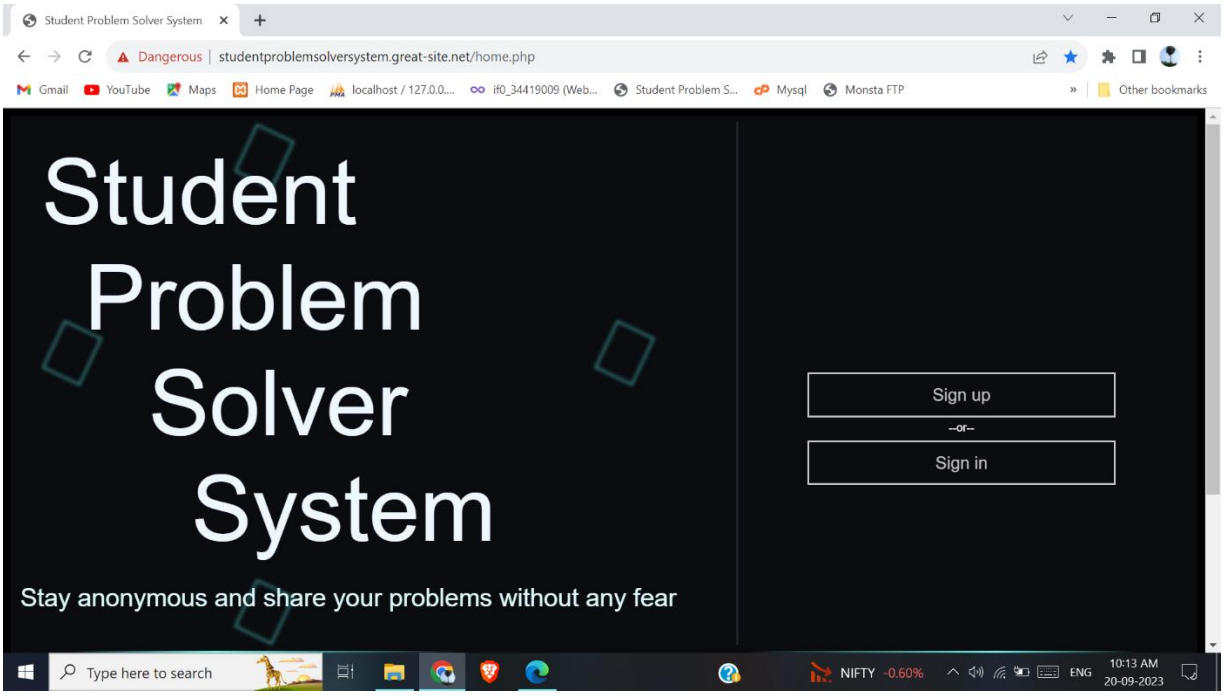


Figure 4.12.1: Home Page UI

Registration:

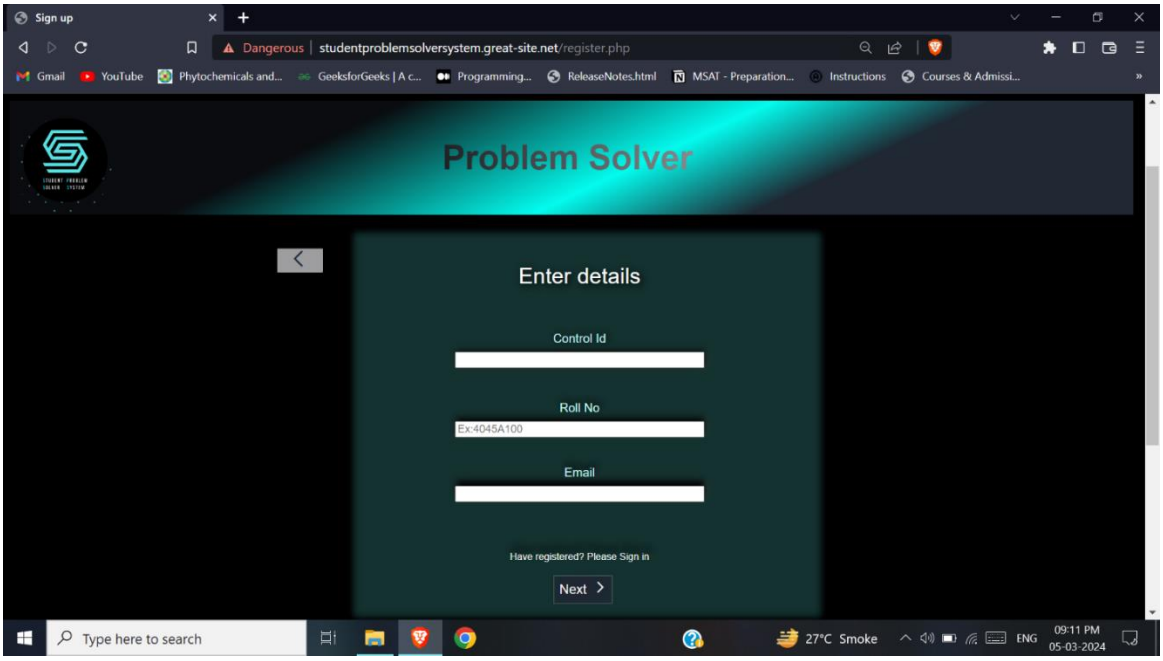


Figure 4.12.2.1: Register Page 1 UI

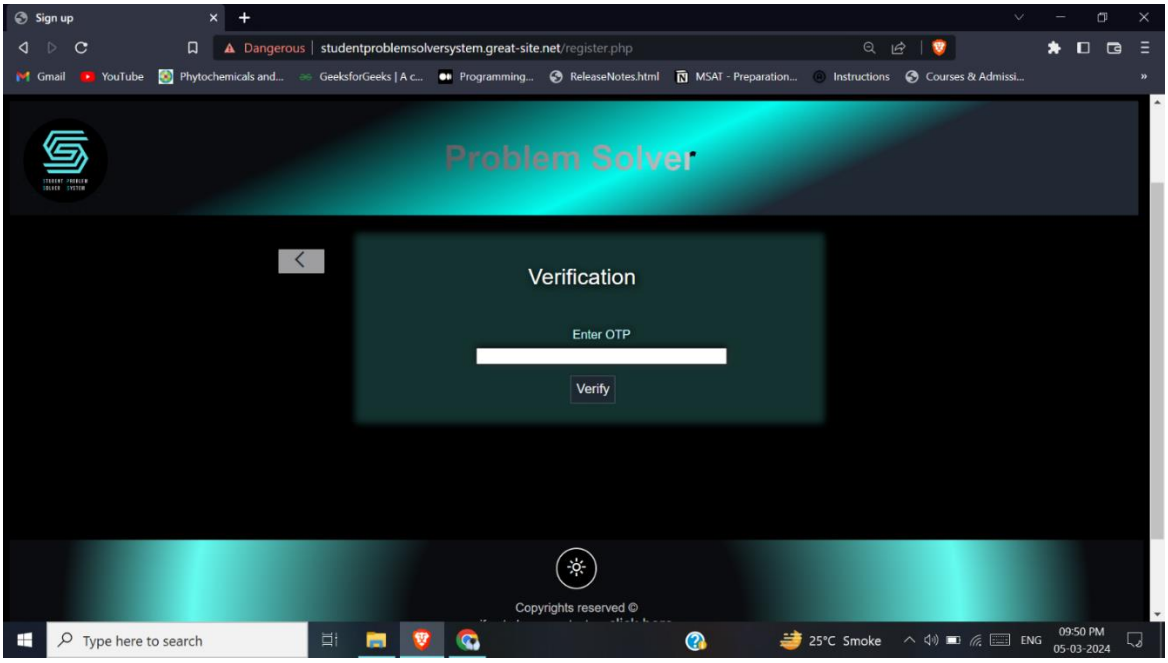


Figure 4.12.2.2: Register Page 2 UI

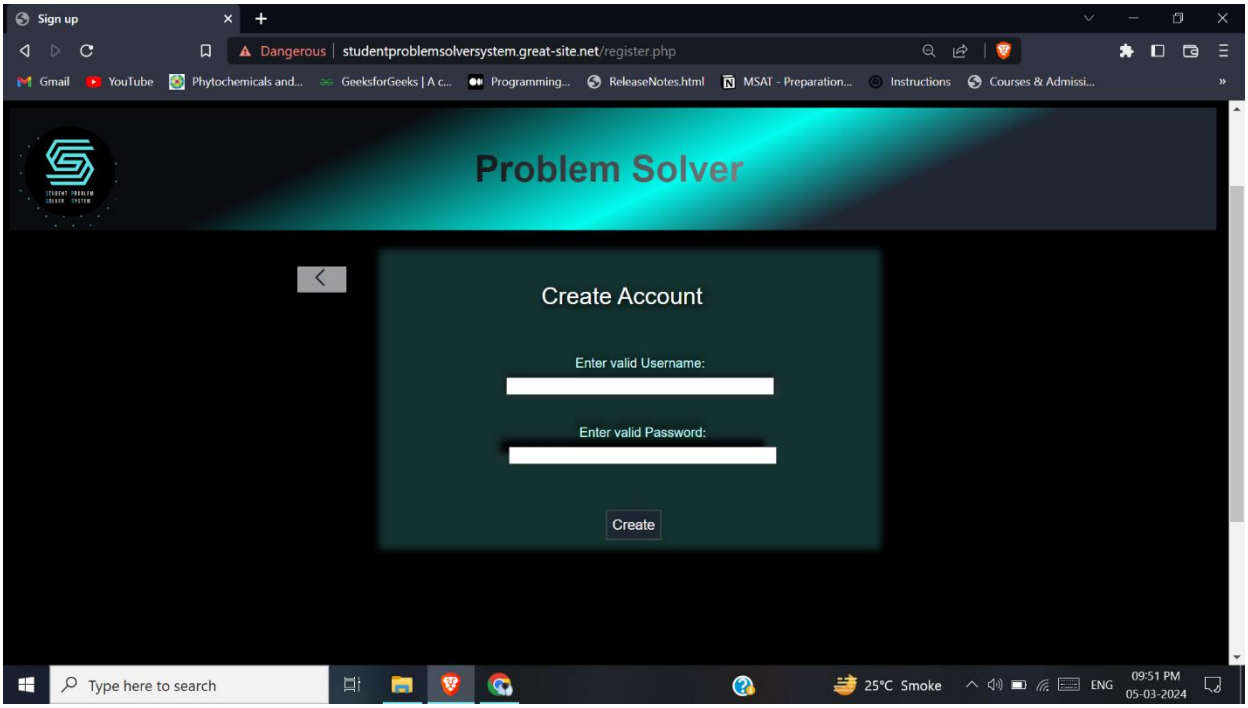


Figure 4.12.2.3: Register Page 3 UI

Login:

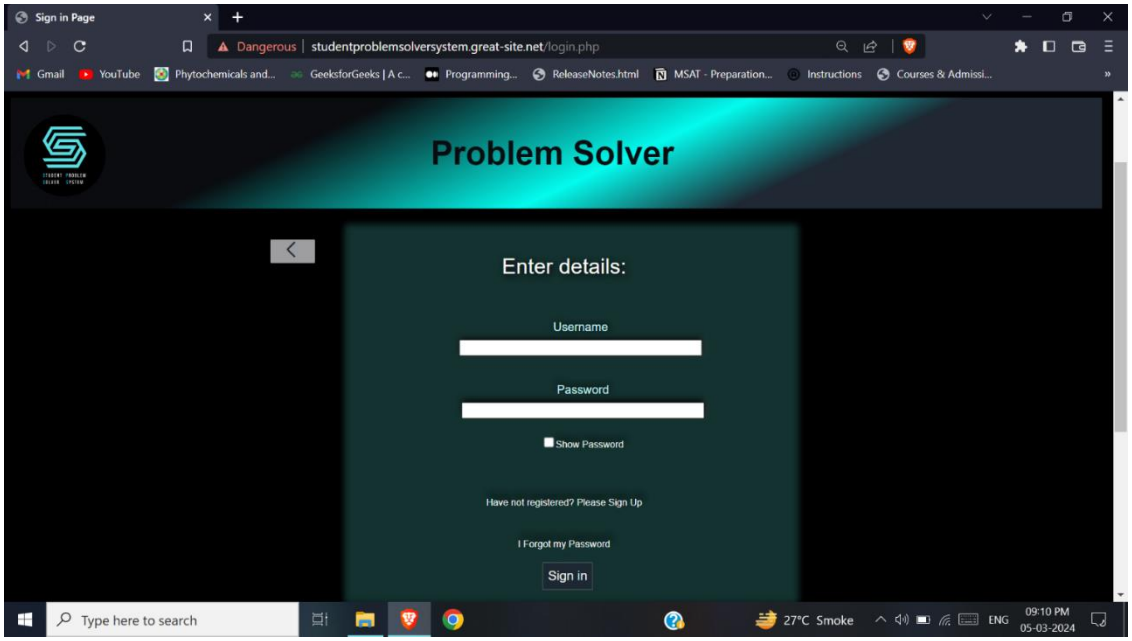


Figure 4.12.3: Login Page UI

View and Vote Post:

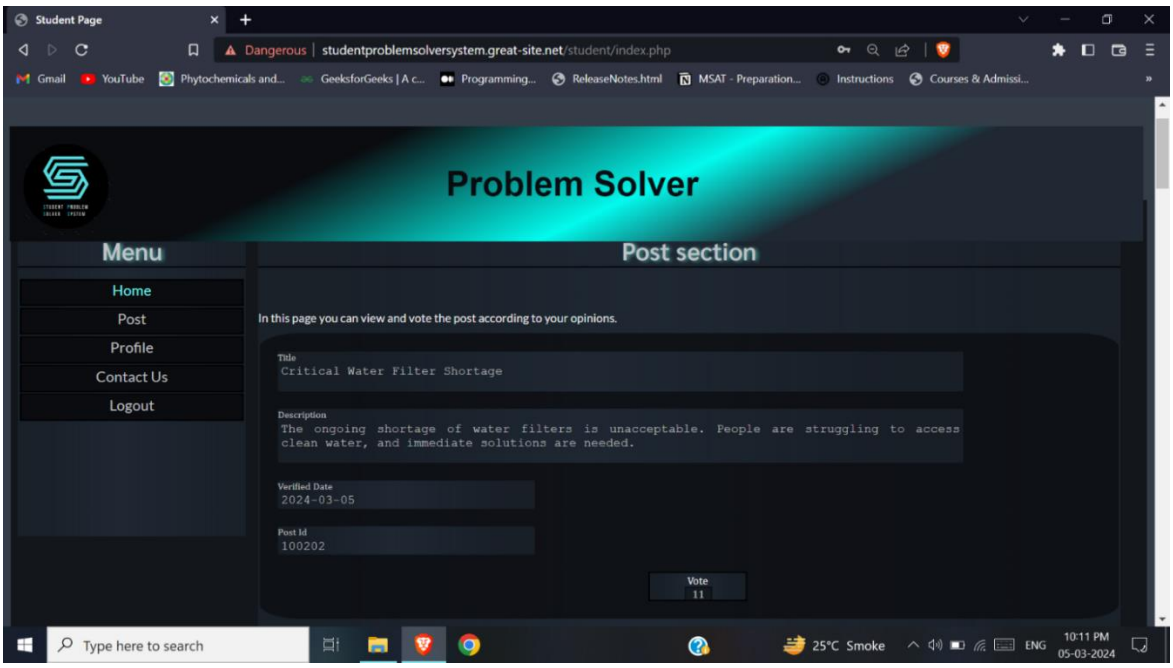


Figure 4.12.4: Student Home Page UI

Send Post:

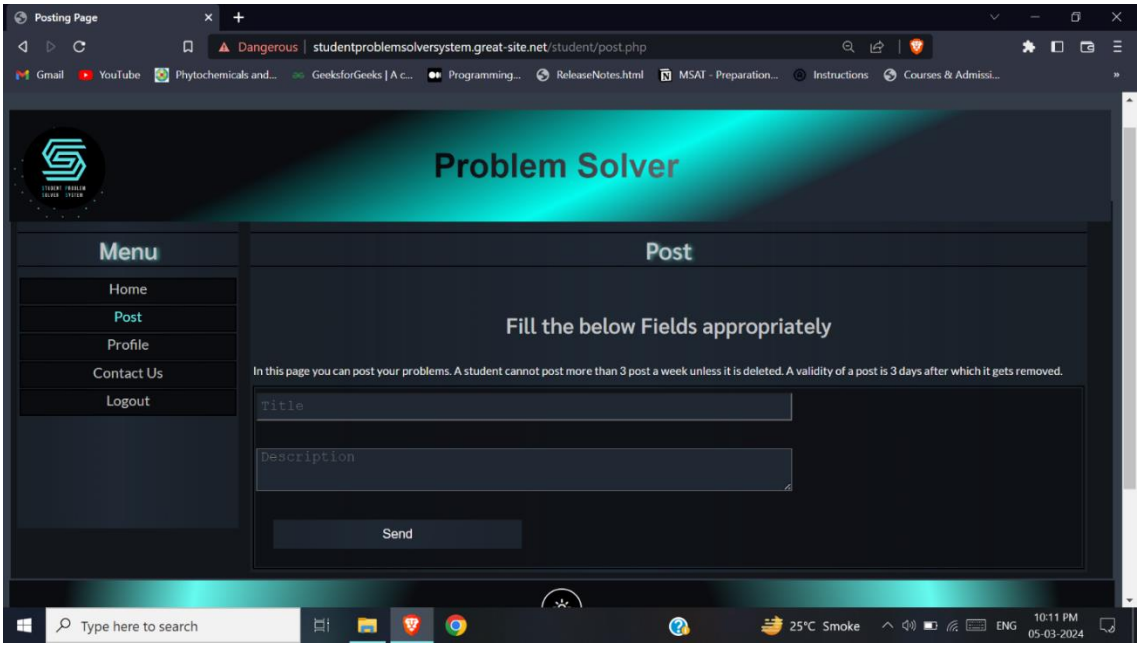


Figure 4.12.5: Student Send Post Page UI

View and Edit Profile:

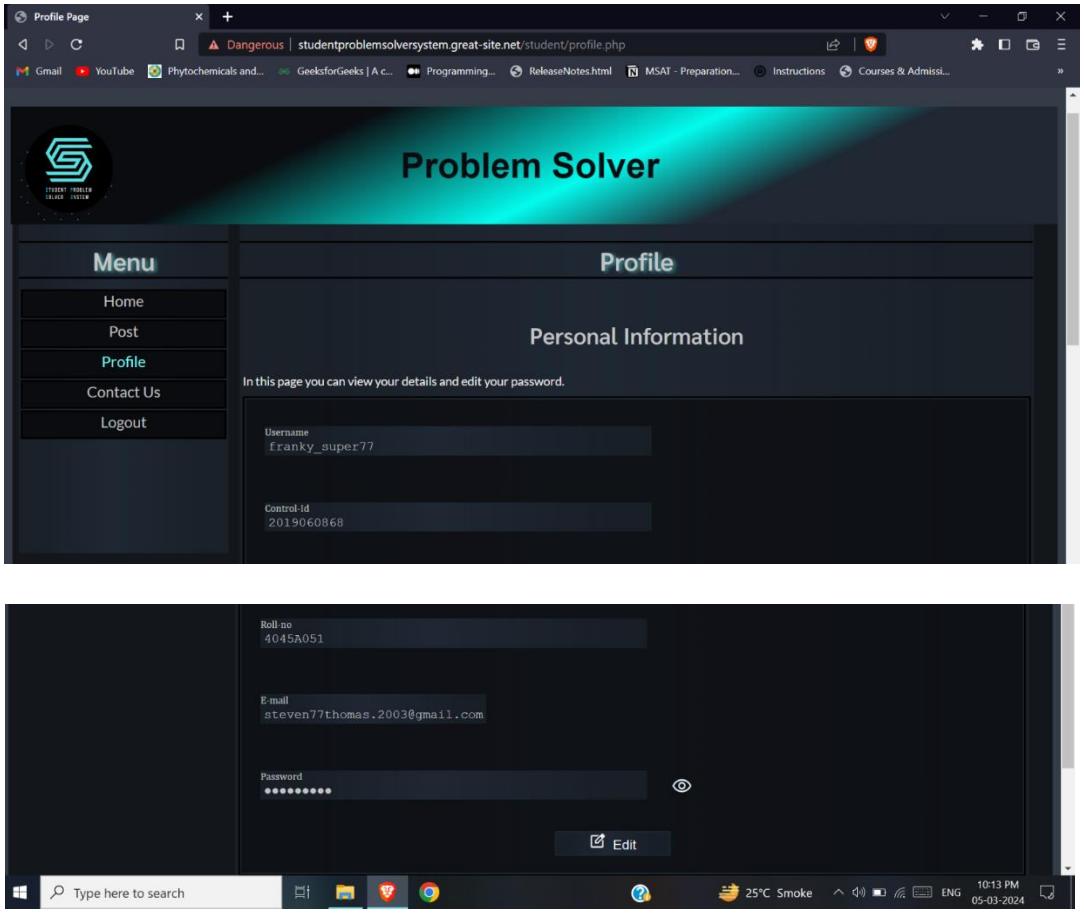


Figure 4.12.6: Student Profile Page UI

Contact us page:

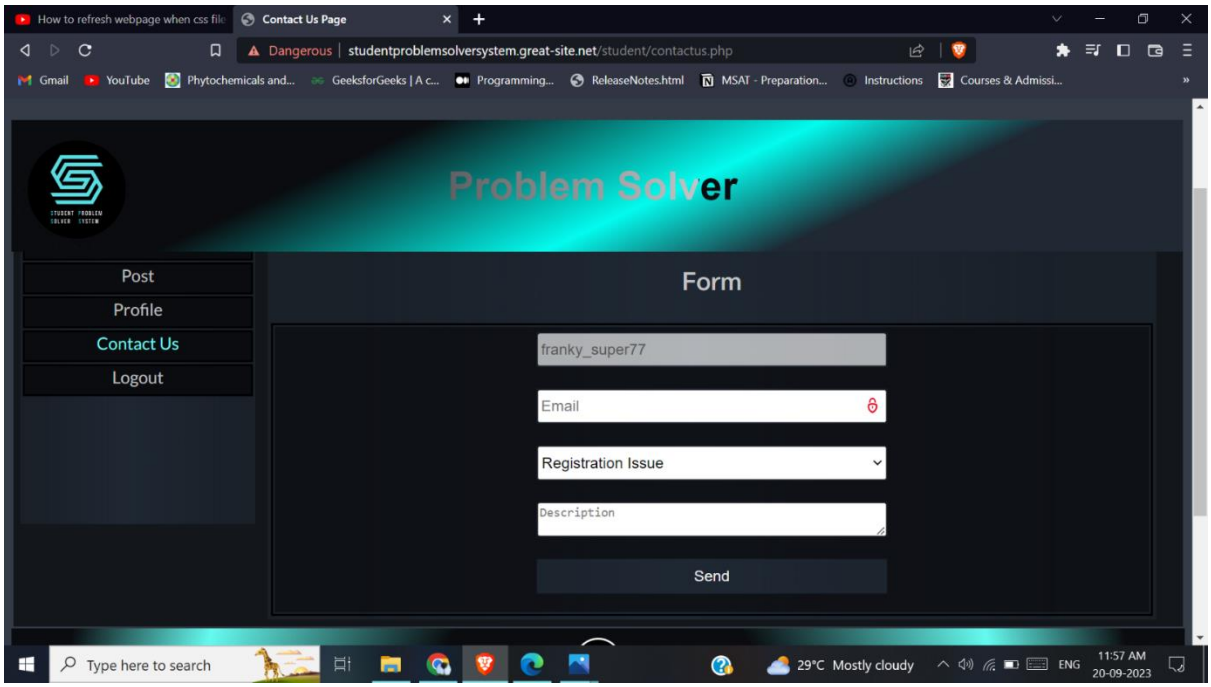


Figure 4.12.7: Student Contact Us Page UI

View Post [Most voted Post]:

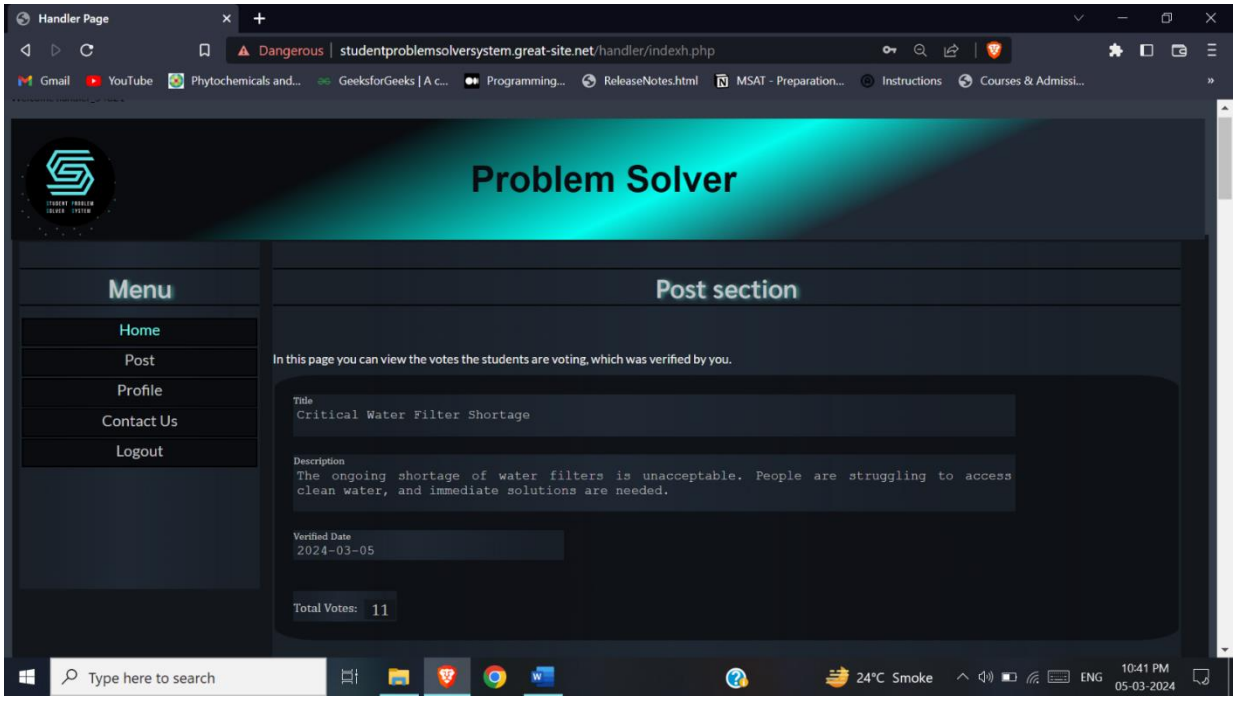


Figure 4.12.8: Handler Home Page UI

Verify and Remove Post:

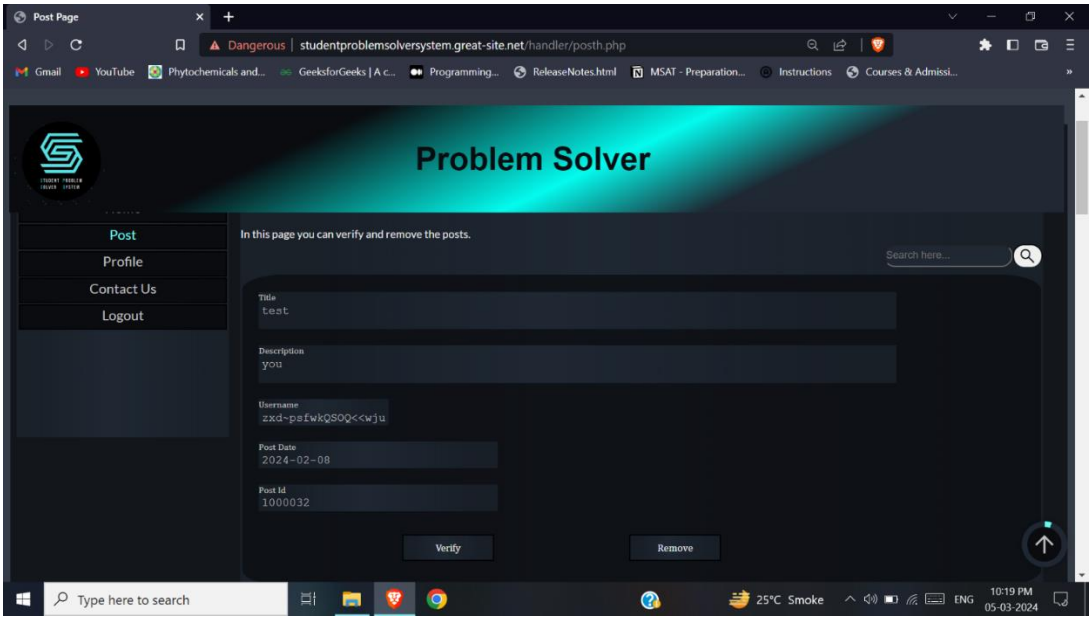


Figure 4.12.9: Handler Post Page UI

View and Edit Profile:

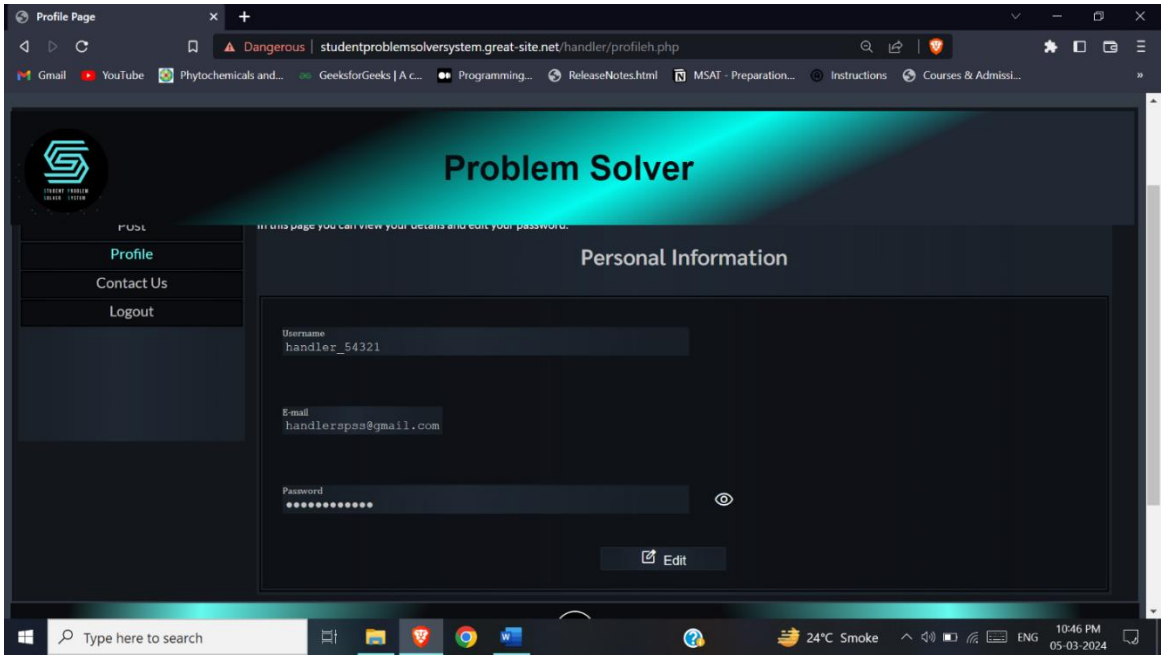


Figure 4.12.10: Handler Profile Page UI

View and Remove Post:

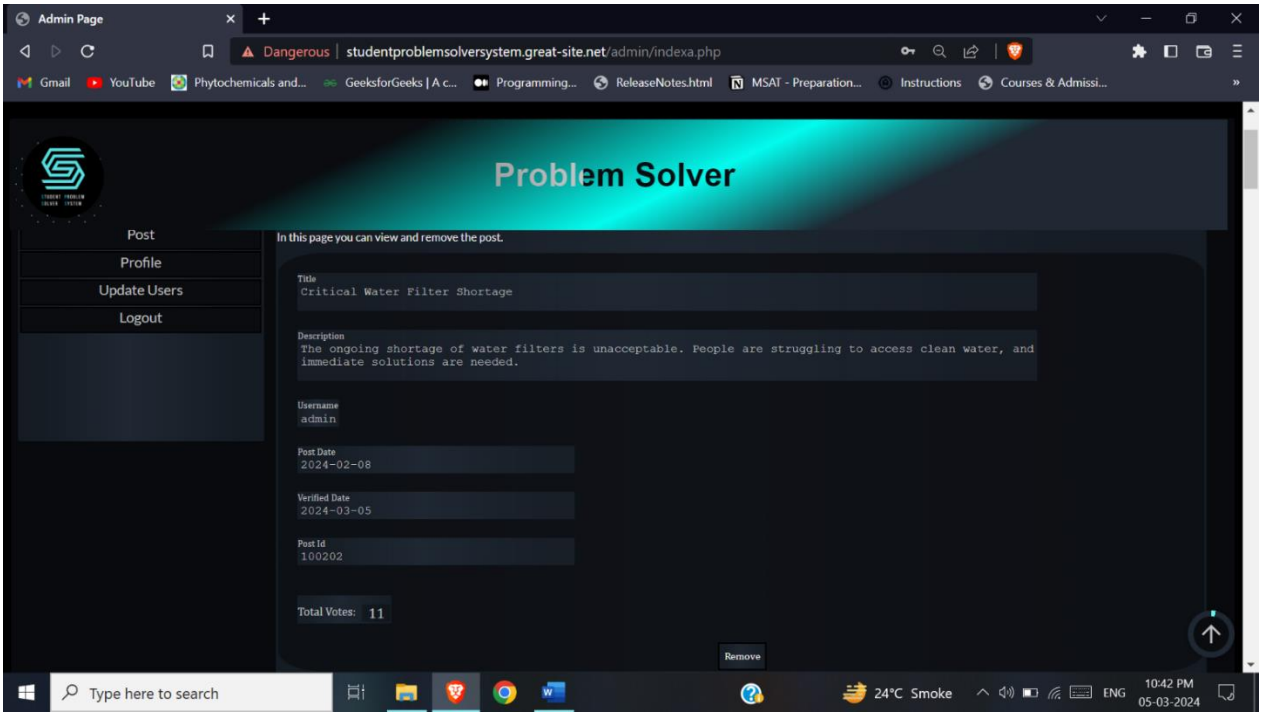


Figure 4.12.11: Admin Home Page UI

Remove Post:

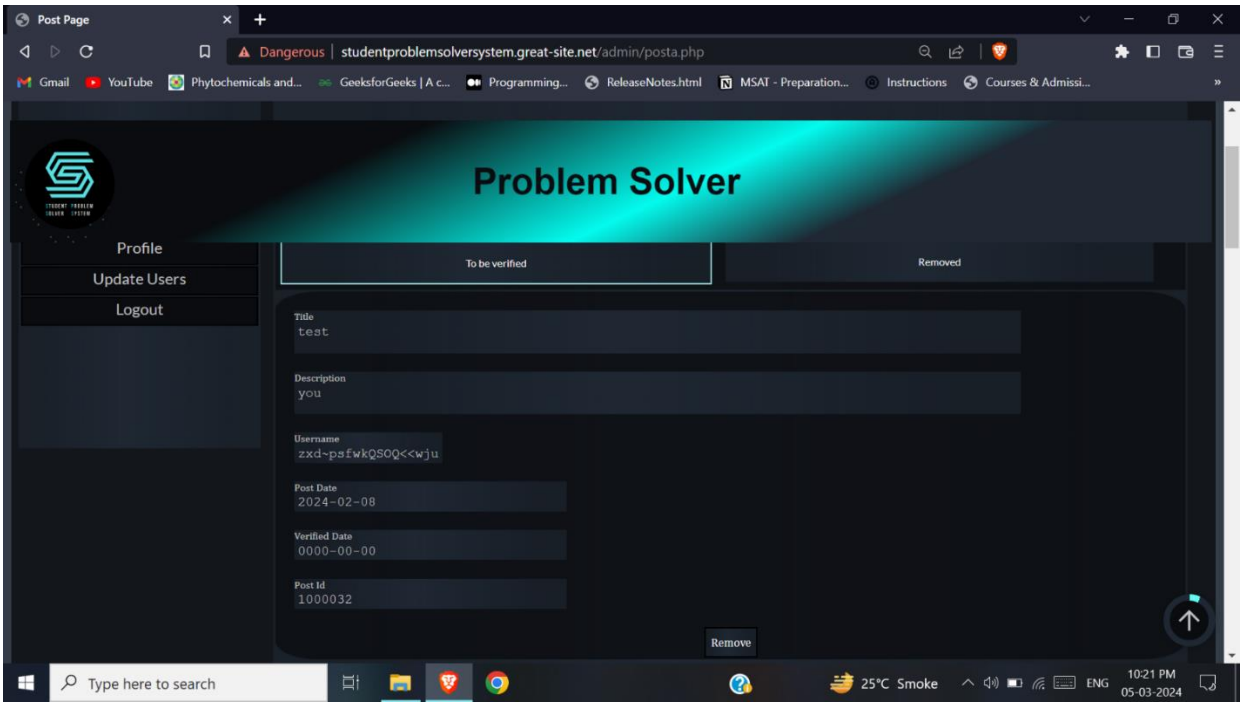


Figure 4.12.12.1: Admin Post Page 1 UI

Verify and Delete Post:

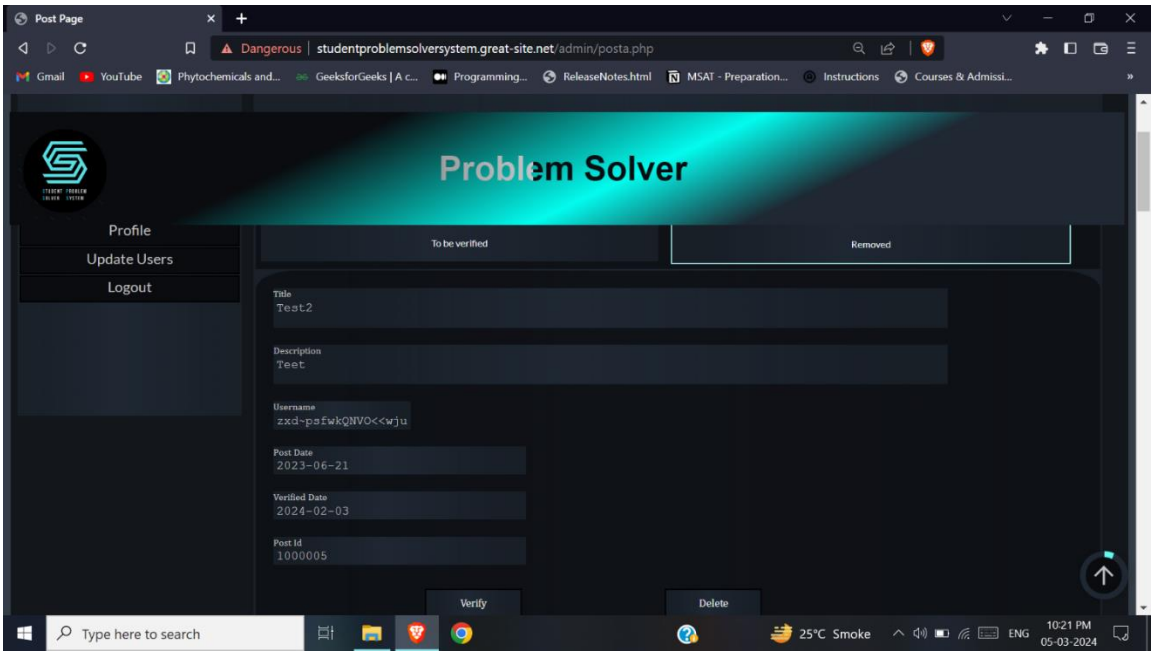


Figure 4.12.12.2: Admin Post Page 2 UI

Manage users Page:

View student accounts:

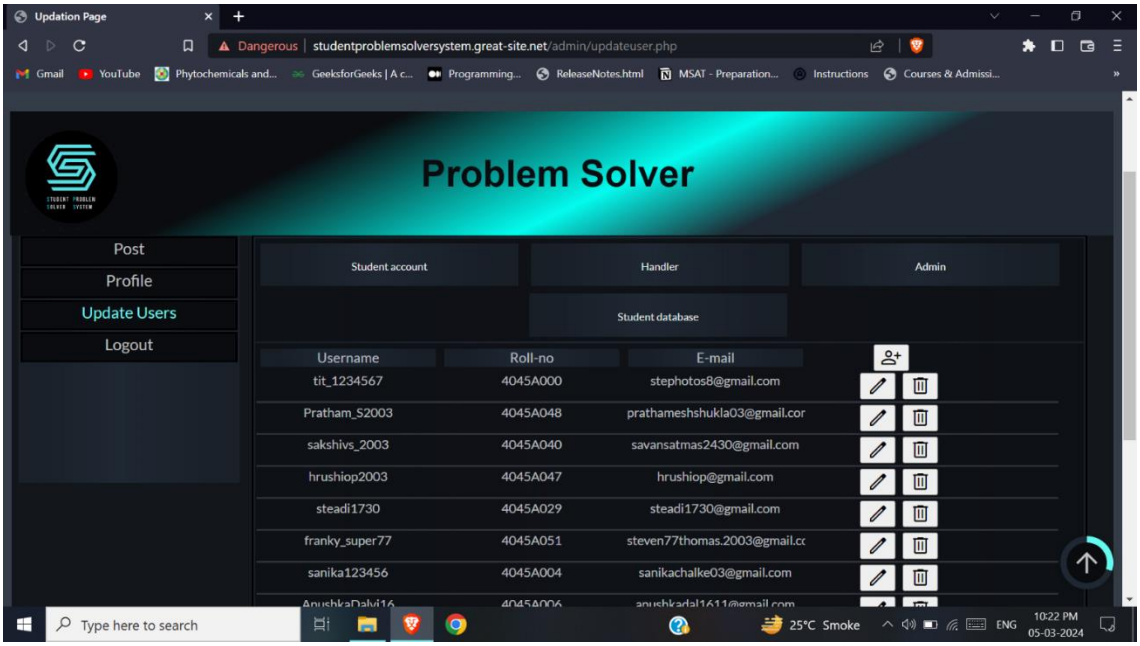


Figure 4.12.13.1: Admin Update Users Page 1 UI

View handlers:

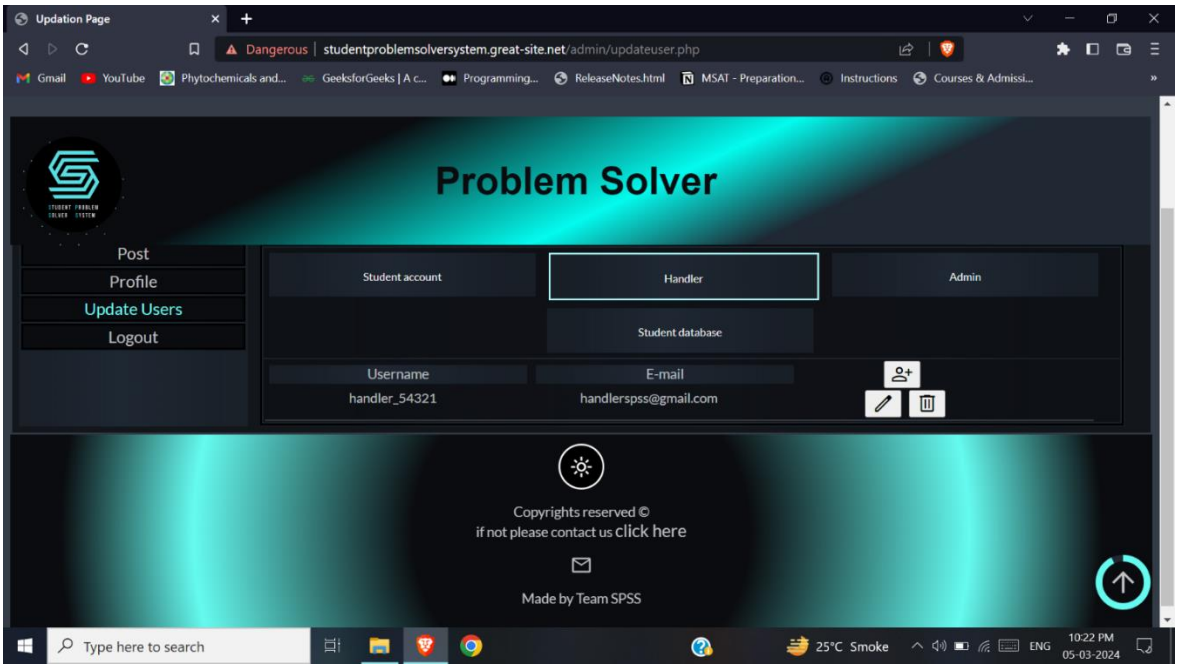


Figure 4.12.13.2: Admin Update Users Page 2 UI

View Admin:

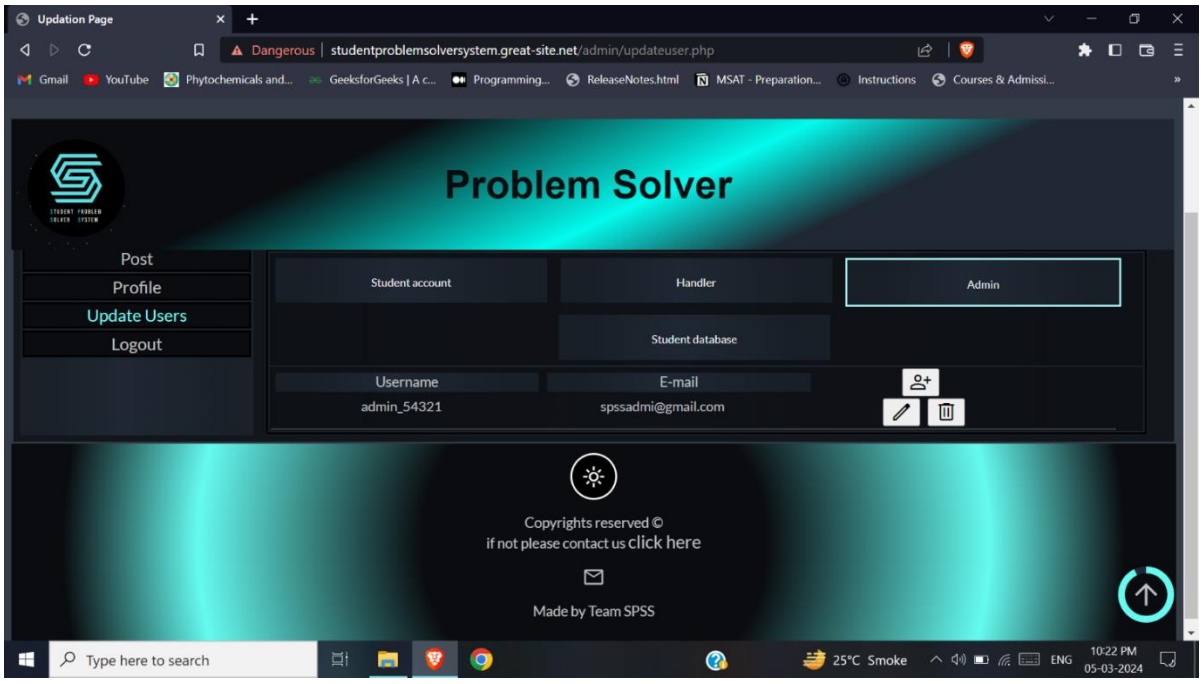


Figure 4.12.13.3: Admin Update Users Page 3 UI

View student database:

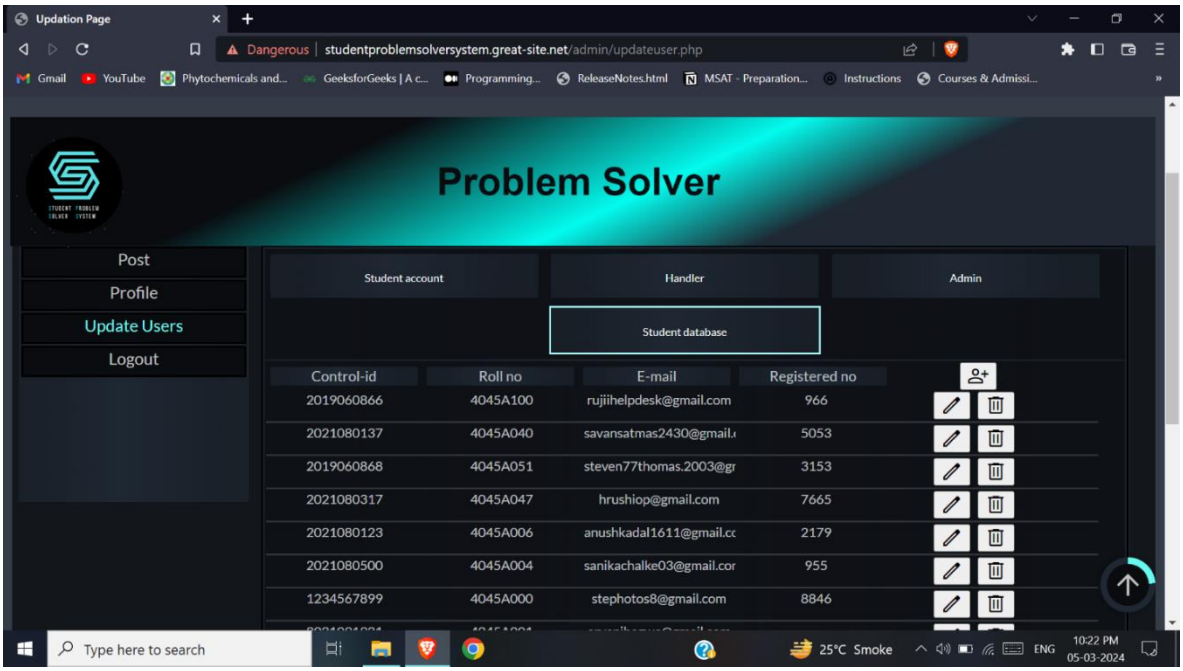


Figure 4.12.13.4: Admin Update Users Page 4 UI

Create user for student account (admin):

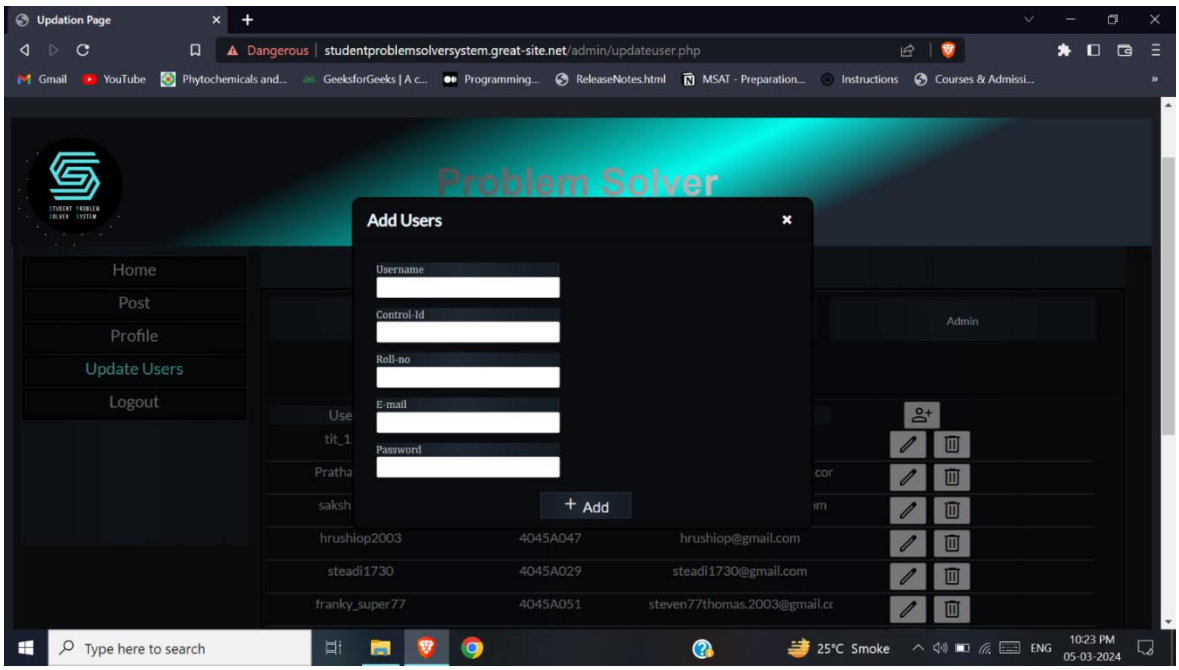


Figure 4.12.13.5: Add user model for student account UI

Edit user for handler (admin):

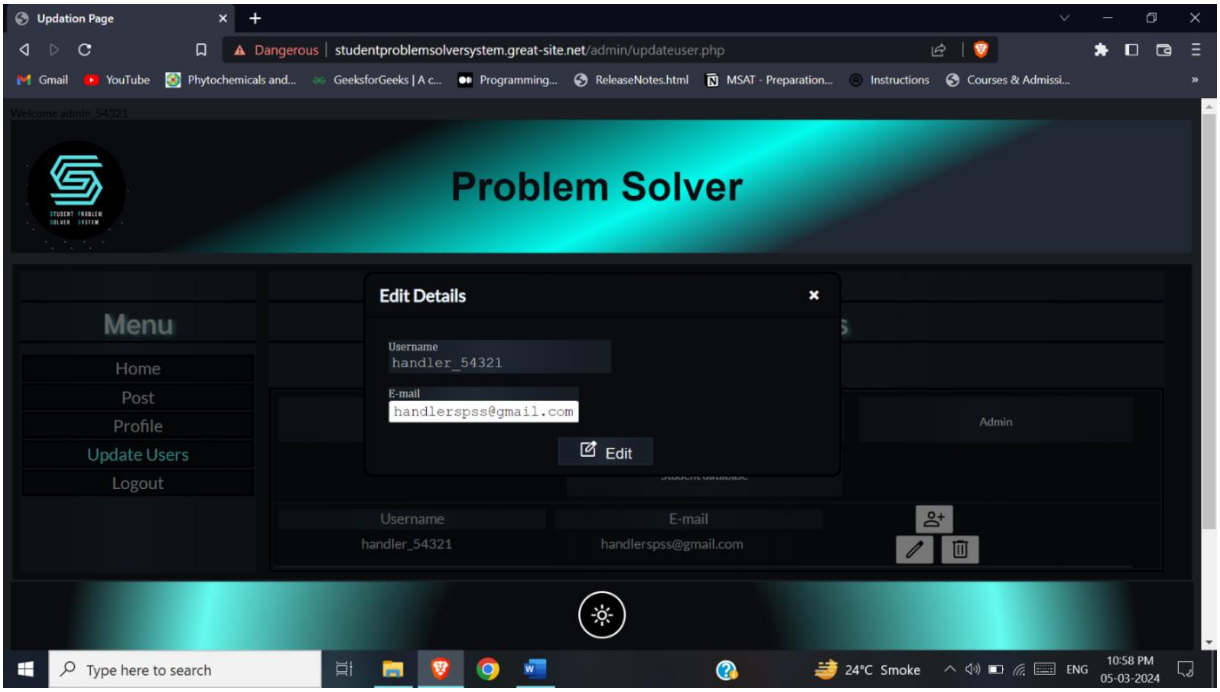


Figure 4.12.13.6: Edit user model for handler UI

Edit user for student account (admin):

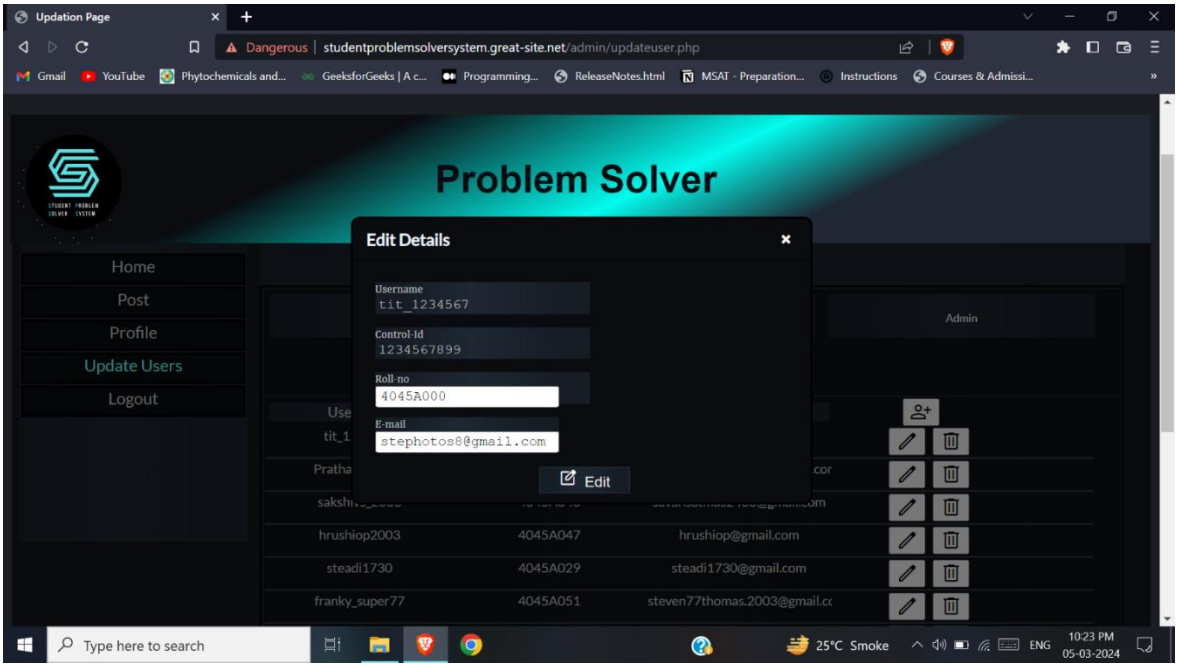


Figure 4.12.13.7: Edit user model for student database UI

4.13 Test Cases

For all users:

Login:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter username: frankysuper77 Password: student@77	Logged in successfully		
2	Enter username: frankysuper77 Password: stude@77	Invalid Password		
3	Enter username: frankysuper	Invalid username		
4	Null values	Please enter details		

Table 4.13.1: Login Test Case

Edit Profile:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	New Password: bye@12345	Details updated		
2	New Password: bye@	Enter valid password		
3	Password: null	Please enter password		

Table 4.13.2: Edit Profile Test Case

Contact Us:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Email: rujiihelpdesk@gmail.com Select Issue: Login issue Description: forgot password	Your mail has been sent		
2	Email: rujigmail.com	Invalid email		
3	Null values	Please enter details		

Table 4.13.3: Contact Us Test Case

For Student:

Registration:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter control id:2019060868 Roll no: 4045A051 Email: rujiihelpdesk@gmail.com Otp: 467856 username: frankysuper78 password: bye#04566	Registered Successfully		
2	control id:201906086	Invalid Control id		
3	control id:2019060866	Already registered control id		
4	Roll no: 4045A222	Invalid Rollno		
5	Email: new	Invalid email		
6.	Email: test@gmail.com	Already used mail		
7.	Otp:43	Invalid otp		
8	Enter username: frankysuper*77	Invalid username should not consist special characters and length should be between 10 and 20		
9	Enter username: frankysuper77	Already registered Username		
10	Password: student@	Password should consist a letter, a digit and special		

		character, and length should be between 8 and 30		
11	Null values	Please enter details		

Table 4.13.4: Registration Test Case

Send Post:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter Title: Filter non-availability Enter Description: Filter is not available in every floor of the college. It should be made available. So please vote for this post.	Post sent successfully, will be posted after verified.		
2	Enter Title: This is title of 4th post Enter Description: This is description of 4th post.	Cannot post more than 3 post per week		
3	Null values	Please enter details		

Table 4.13.5: Send Post Test Case

Vote Post:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Vote post	Voted		
2	Vote same post again	You have already voted		
3	Null values	Please enter details		

Table 4.13.6: Vote Post Test Case

For handler and admin

Verify Post:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Verify post	Verified		

Table 4.13.7: Verify Post Test Case

Remove Post:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Remove post	Post removed		

Table 4.13.8: Remove Post Test Case

Delete Post:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Delete post	Post deleted		

Table 4.13.9: Delete Post Test Case

For Admin:

Add student account:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter control id:2019060869 Roll no: 4045A055 Email: rujiihelpdesk2@gmail.com username: frankysuper79 password: bye#04569	User added		
2	control id:201906086	Invalid Control id		
3	control id:2019060866	Registered control id		
4	Roll no: 4045A222	Invalid Rollno		
5	Email: new	Invalid email		

6.	Email: test@gmail.com	Already used mail		
7	Enter username: frankysuper*77	Invalid username, should not consist special characters and length should be between 10 and 20		
8	Enter username: frankysuper77	Registered Username		
9	Password: student@	Password should consist a letter, a digit and special character, and length should be between 8 and 30		
10	Null values	Please enter details		

Table 4.13.10: Add student account Test Case

Edit student account:

Test No	**Test Case Name**	**Expected Output**	**Actual Output**	**Remark**
1	Roll no: 4045A045 Email: rujiihelpdesk3@gmail.com username: frankysuper80 password: bye#04570	User data updated		
2	Roll no: 4045A222	Invalid Rollno		
3	Email: new	Invalid email		
4	Email: test@gmail.com	Already used mail		

5	Enter username: frankysuper*77	Invalid username, should not consist special characters and length should be between 10 and 20		
6	Enter username: frankysuper77	Registered Username		
7	Password: student@	Password should consist a letter, a digit and special character, and length should be between 8 and 30		
8	Null values	Please enter details		

Table 4.13.11: Edit student account Test Case

Add handler:

Test No	**Test Case Name**	**Expected Output**	**Actual Output**	**Remark**
1	Enter username: hand_1234 Password: hand@hand12	User added		
2	Enter username: hand*1234	Invalid username, should not consist special characters and length should be between 10 and 20		
3	Password: hand12	Password should consist a letter, a digit and special		

		character, and length should be between 8 and 30		
4	Null values	Please enter details		

Table 4.13.12: Add handler Test Case

Add Admin:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter username: admi_1234 Password: admin@admi12	User added		
2	Enter username: admi*1234	Invalid username, should not consist special characters and length should be between 10 and 20		
3	Password: admi12	Password should consist a letter, a digit and special character, and length should be between 8 and 30		
4	Null values	Please enter details		

Table 4.13.13: Add Admin Test Case

Edit handler:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter username: hand_1235 Password: hand@hand13	User data updated		
2	Enter username: hand*1235	Invalid username, should not consist special characters and length should be between 10 and 20		

3	Password: hand13	Password should consist a letter, a digit and special character, and length should be between 8 and 30		
4	Null values	Please enter details		

Table 4.13.14: Edit handler Test Case

Edit Admin:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter username: admi_1235 Password: admi@admi13	User data updated		
2	Enter username: admi*1235	Invalid username, should not consist special characters and length should be between 10 and 20		
3	Password: admi13	Password should consist a letter, a digit and special character, and length should be between 8 and 30		
4	Null values	Please enter details		

Table 4.13.15: Edit admin Test Case

Delete user:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Delete user	User deleted		

Table 4.13.16: Delete user Test Case

Chapter 5

Coding and Implementation

5.1 Implementation approaches

This project was implemented using the Incremental model. In incremental methodology, the model is designed, implemented, and tested incrementally. If in case there is a change in the requirements of the user then that part can be redesigned, re-implemented, and tested again iteratively. This model allows us to update the system at each increment and we can add new functionalities as well.

The interfaces are designed and created using HTML CSS and JavaScript. After the user interfaces were created, database connectivity was performed by using PHP. I have used MySQL as my database and hosted the site at the free web hosting site "infinityfree.com ". The coding part of my project is done in JavaScript and PHP language. The project was divided into modules. These modules were created one by one and after completion of each module, unit testing was performed on that module. When the module fulfils its requirements, it was integrated into the main project. After integration, each functionality was checked which can also be said to be as integration testing.

After adding all the modules to my main project, finally system testing was performed to check whether the system is working accordingly or not.

I have used validations wherever it was required. The system is made by considering all the problems into the view and the final project should be fulfilling all the requirements.

5.2 Coding and Efficiency

5.2.1 Coding:

JavaScript code of Registration:

Student filled details are retrieved and validated.

register.php

under the script tag

```
 let rotp="";
   var validate1=true;
    var validate2=false;
//auto fill the other input elements witht the control id input on onchange()
function autofill()
```

```
{
   $c_id = $('#rrcid').val();

   if($c_id=="")
   {
      alert("Enter control id");
   }
   else{
   $.ajax({
                type: 'POST',
                url: 'student/check_cid_fetch.php',
                data: {control_id:$c_id},
                success: function(data) {
                   data=JSON.parse(data);
                   if(data=="NC")
                   {
                      console.log("No data");
                       $('#rrrno').val("");
                    $('#rrmai').val("");
                   }
                   else{
                        console.log("data present");
                    $('#rrrno').val(data.roll_no);
                    $('#rrmai').val(data.email);
                   }
                }   })}}
//function for vlaidation of the controlid, roll no and email and to go the next container of otp
 $('#next1').on('click',function(e){
   $control_id = $('#rrcid').val();//covert id to php variable
   $roll_no = $('#rrrno').val();
   $e_mail = $('#rrmai').val();
```

```
 let valid=false;
valid=true;
if(valid){
   if($control_id!=="")
   {
      if($roll_no!=="")
      {
         if($e_mail!=="")
         {
            $.ajax({
               type: 'POST',
               url: 'student/check_cid.php',
               data: {control_id:$control_id,roll_no:$roll_no,e_mail:$e_mail},
               success: function(data) {
                  console.log(data);
                  if(data=="EU")
                  {
                     $('.verify-cont').show();
                     $('.verify-fake').show();
                     $('.sign-up-cont').hide();
                     $('.sign-up-fake').hide();
                     console.log("OTP is send to the given email");
                     console.log("otp sent successfully");
               let rmsg="";let rto="";let rsub="";let n=[];
                   rotp=Math.floor((Math.random()*1000000)+1);
                  rmsg='This is your OTP '+rotp;
                  rto=$e_mail;
                  rsub="OTP Verification";
                   Email.send({
   SecureToken : "6b9d686d-28ab-44b7-9793-ba592582e2a1",
   To : rto,
```

```
    From : "studentproblemsolver05@gmail.com",
    Subject : rsub,
    Body : rmsg
}).then(
  alert('otp sent to mail. Please check spam.')
);
                }
                    else if(data=="EUEU"){
                      alert("Already registered");
                      }
                    else if(data=="EUNR")
                    {
                        alert("roll no does not match with control id");
                    }
                    else if(data=="EUNE")
                    {
                        alert("email does not match with control id");
                    }
                    else if(data=="NU")
                    {
                        alert("control id does not exist");
                    }
                    else{
                         alert("something went wrong");
                    }
                  },
                  error: function() {
                    console.log(response.status);
                  },
                })
            }
```

```
            else{
        alert("fill email");
    }
        }
        else{
        alert("fill roll no");
    }
    }
    else{
        alert("fill control id");
    }
}
    e.preventDefault();
 })

 //function to verify the otp and to go to the next container of username and password
   function verifyotp(){
    $otp=$('#rrotp').val();
      if($otp!=="")
      {
            if($otp==rotp){
        alert("Correct otp");
        $('.verify-cont').hide();
     $('.verify-fake').hide();
        $('.user-cont').show();
      $('.user-fake').show();
        }
        else{
        alert("Please enter valid otp");
        }
     }
```

```
        else
           {
           alert("Enter otp");
           }
         }

        //validation for the username and password
   function createacc(){
    $use=$('#rruse').val();
    $pas=$('#rrpas').val();
  if($use.length>20)
  {
    alert("Username length must not exceed 20 characters");
     validate1=false;
  }
  else if($use.length<10)
  {
    alert("Username length must not be less than 10 characters");
     validate1=false;
  }
  else{
let s=$use;let counta=0;let countn=0;
let countc=0;let invalid=0;validate1=true;
    for(let i=0;i<$use.length;i++)
    {
let str=s.charAt(i);
if((/[a-zA-Z]/).test(str))
{
  counta+=1;
}
else if(str=='_')
```

```
{
  countc+=1;
}
else if(str=='0' ||str=='1' ||str=='2' ||str=='3' ||str=='4' ||str=='5' ||str=='6' ||str=='7' ||str=='8'
||str=='9')
{
  countn+=1;
}
else{
  invalid=1;
  break;
}
    }
    if(invalid==1)
    {
      alert("In username, No space or special character allowed, except underscore(_)");
      validate1=false;
    }
    else if(counta==0)
    {
      alert("In username, Minimum one alphabet required");
       validate1=false;
    }
    else if(countn==0)
    {
      alert("In username, Minimum one number required");
       validate1=false;
    }
  }
  if($pas.length<8)
  {
    alert("Password should not be less than 8 characters");
```

```
  }
  else{
let s=$pas;let counta=0;let countn=0;
let countc=0;let invalid2=0;validate2=false;
    for(let i=0;i<$pas.length;i++)
    {
let str=s.charAt(i);
if((/[a-zA-Z]/).test(str))
{
  counta+=1;
}
else if(str==' ')
{
  invalid2=1;
}
else if(str=='0' ||str=='1' ||str=='2' ||str=='3' ||str=='4' ||str=='5' ||str=='6' ||str=='7' ||str=='8'
||str=='9')
{
  countn+=1;
}
else{
  countc+=1;
}
}
    if(invalid2==1){
      alert("In password, No space character allowed");
    }
    else if(counta==0)
    {
      alert("In password, Minimum one alpahbet required");
    }
    else if(countn==0)
```

```
    {
      alert("In password, Minimum one number required");
    }
    else if(countc==0)
    {
      alert("In password, Minimum one special character required");
    }
    else{
      validate2=true;
    }
  }
if(validate1==true && validate2==true)
{
    if($use!=="")
      {
       if($pas!==""){
        $.ajax({
          type:'POST',
    url:'student/check_usepass.php',
    data:{control_id:$control_id,roll_no:$roll_no,e_mail:$e_mail,use:$use,pas:$pas},
    success: function(data){
      console.log(data);
     if(data=='NU'){
alert("Correct credentials");
 window.location='../student/index.php';
        }
        else if(data=='EU')
        {
          alert("Username already exists.Please change.");
        }
        else{
```

```
        alert("Some issue");
        }
    }


})
      }
      else{
       alert("Please enter password");
      }
     }
     else
     {
     alert("Please enter username");
     }


   }  }
 let valid=false;
```

Backend code of Registration:

On entering controlid the other details are fetched and filled. The student details are inserted to the student_account table after successful creation of an account.

check_cid.php

```
<?php
$con =
mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_spss_db');
if(mysqli_connect_errno())
{
   die("Not connected");
}
if(!empty($_POST['control_id']) &&  !empty($_POST['roll_no']) &&
!empty($_POST['e_mail']))
{
```

```
   $control_id = mysqli_real_escape_string($con,$_POST['control_id']);
   $roll_no = mysqli_real_escape_string($con,$_POST['roll_no']);
   $e_mail = mysqli_real_escape_string($con,$_POST['e_mail']);
   $checking_existing_user = "SELECT roll_no FROM student WHERE control_id =
'$control_id'";
   $result_of_existing_user = mysqli_query($con,$checking_existing_user);
   $checking_mail = "SELECT email FROM student WHERE control_id = '$control_id'";
   $result_of_mail=mysqli_query($con,$checking_mail);
$check_regno="SELECT registered_no FROM student WHERE control_id = '$control_id'";
$result_of_regno=mysqli_query($con,$check_regno);
   if(mysqli_num_rows($result_of_existing_user)>0)
   {
      $result_of_existing_userstr= implode("",(mysqli_fetch_row($result_of_existing_user)));
      $result_of_mailstr= implode("",(mysqli_fetch_row($result_of_mail)));
      $result_of_regnostr= implode("",(mysqli_fetch_row($result_of_regno)));
 if($result_of_regnostr!=""){//already registered error
   echo("EUEU");
 }
      else if($result_of_existing_userstr != $roll_no)
   {
      echo("EUNR");
   }
      else if($result_of_mailstr != $e_mail){
      echo("EUNE");
   }
   else{
      echo ("EU") ;//EXISTING USER
   }
   }
   else{
      echo("NU");//not user
   }
```

```
}
?>
```

check_usepass.php

```
<?php
session_start();
$con = mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_spss_db');
$cid=$_POST['control_id'];
$rno=$_POST['roll_no'];
$email=$_POST['e_mail'];
$use=$_POST['use'];
$_SESSION['USERNAME']=$use;
$pas=$_POST['pas'];
$head = mt_rand(1, 10000);
$headstr = strval($head);

$sarr = str_split($headstr);
for ($i = 0; $i < strlen($headstr); $i++) {
   $sarr[$i] = chr(ord($headstr[$i]) + 30);
}
$str = implode('', array_reverse($sarr));
$sarr = str_split($use);
for ($i = 0; $i < strlen($use); $i++) {
   $sarr[$i] = chr(ord($use[$i]) + 5);
}
$str = $str . implode('', array_reverse($sarr));
$str1 = substr($str, 0, strlen($str) / 2);
$str2 = substr($str, strlen($str) / 2);
$str = $str2 . $str1;
$puse=$str;
$res=mysqli_query($con,"SELECT * FROM stud_account WHERE stu_username='$use'");
```

```
$count=mysqli_num_rows($res);
if($count>0){
   echo "EU";//existing username
}
else{

$simple_string = "$pas";
// Store the cipher method
$ciphering = "AES-128-CTR";
// Use OpenSSl Encryption method
$iv_length = openssl_cipher_iv_length($ciphering);
$options = 0;
// Non-NULL Initialization Vector for encryption
$encryption_iv = '1234567891011121';
// Store the encryption key
$encryption_key = "$puse";
// Use openssl_encrypt() function to encrypt the data
$encryption = openssl_encrypt($simple_string, $ciphering,
                              $encryption_key, $options, $encryption_iv);

   mysqli_query($con,"INSERT INTO `stud_account`(`stu_username`, `stu_email`,
`stu_password`, `public_username`, `stu_controlid`, `stu_rollno`) VALUES
('$use','$email','$encryption','$puse','$cid','$rno')");
   mysqli_query($con,"UPDATE `student` SET `registered_no`='$head' WHERE
control_id='$cid'");
   // mysqli_query($con,"INSERT INTO `session_tb`(`sess_username`, `sess_userstatus`)
VALUES ('$use','active')");
   $_SESSION['IS_LOGIN']=$email;
   echo "NU";
}
?>
```

JavaScript code of index page(student):

Post component is created according to the data fetched and sorted in an ascending order of votes in and displayed in the page.

index.php

inside the script tag

```
function load_toPost(){
let tempp="";let n=[];
var sess_use="<?php echo $_SESSION['USERNAME']?>";
let iu=document.querySelector('#indexusername');
iu.innerText=sess_use;
$.ajax({
type: 'POST',
url: '../handler/indexh_fetch.php',
data: {use:sess_use},
success:function(data){
if(data=="zero"){
    let y = document.getElementById("no-post");
    y.style.display = "block";
}
 else if(data=="NF"){
     alert("Could not update posts");
 }
 else{
   let y = document.getElementById("no-post");
    y.style.display = "none";
let dataa=JSON.parse(data);
dataa=dataa.sort((a,b)=>{
 return parseInt(b.post_votes) - parseInt(a.post_votes);
})
    for(let ii=0;ii<dataa.length;ii++) {
    addpost();
    }
```

```
    let ppid=0;
    for(let ii=0;ii<dataa.length;ii++) {
        ppid=ppid+1;
    let s='#posttt'+ppid.toString();
    let ppo=document.querySelector(s);
ppo.childNodes[0].lastChild.innerText=dataa[ii].post_title;
ppo.childNodes[1].lastChild.innerText=dataa[ii].post_desc;
ppo.childNodes[2].lastChild.innerText=dataa[ii].verified_date;
ppo.childNodes[3].lastChild.innerText=dataa[ii].post_id;
ppo.childNodes[4].lastChild.innerText=dataa[ii].post_votes;
    }
 }
}
})
   }


var bodyp,divp;var post_new;
function addtitle()
{
  const posttitle = document.createElement("div");
  posttitle.innerText ="Title";
   posttitle.setAttribute('id','post-title');
   posttitle.classList.add('post-con-itm');
  divp.appendChild(posttitle);
  let para = document.createElement("p");
  para.setAttribute('id','contentHoldertit');
     posttitle.appendChild(para);
}
function adddesc(){
  const postdesc = document.createElement("div");
```

```
  postdesc.innerText ="Description";
   postdesc.setAttribute('id','post-desc');
   postdesc.classList.add('post-con-itm');
  divp.appendChild(postdesc);
   para = document.createElement("p");
  para.setAttribute('id','contentHolderdes');
     postdesc.appendChild(para);
}
function adduse(){
  const postuse = document.createElement("div");
  postuse.innerText ="Username";
   postuse.setAttribute('id','post-usenamee');
   postuse.classList.add('post-con-itm');
   postuse.style.width='fit-content';
  divp.appendChild(postuse);
   para = document.createElement("p");
  para.setAttribute('id','contentHolderuse');
     postuse.appendChild(para);
}
function addpostdate()
{
  const postdate = document.createElement("div");
  postdate.innerText ="Post Date";
   postdate.setAttribute('id','post-date');
   postdate.classList.add('post-con-itm');
  divp.appendChild(postdate);
   para = document.createElement("p");
  para.setAttribute('id','contentHolderpstd');
     postdate.appendChild(para);
}
function addposteddate(){
```

```
  const posteddate = document.createElement("div");
     posteddate.innerText ="Verified Date";
      posteddate.setAttribute('id','posted-date');
      posteddate.classList.add('post-con-itm');
     divp.appendChild(posteddate);
      para = document.createElement("p");
     para.setAttribute('id','contentHolderpstdd');
        posteddate.appendChild(para);
}
function addpostid(){
  const postid = document.createElement("div");
  postid.innerText ="Post Id";
   postid.setAttribute('id','post-id');
   postid.classList.add('post-con-itm');
  divp.appendChild(postid);
   para = document.createElement("p");
  para.setAttribute('id','contentHolderpsid');
     postid.appendChild(para);
}
function addposttvote(){
  let postvote = document.createElement("button");
  postvote.innerText ="Vote";
   postvote.setAttribute('id','post-vote');
   postvote.classList.add('post-con-itm','submi');
  divp.appendChild(postvote);
  let count = document.createElement("div");
  count.setAttribute('id','post-count');
  count.classList.add('post-con-itm');
  postvote.appendChild(count);
  postvote.addEventListener("click",e =>{
    countincphp(e);
```

```
  });
}

function countincphp(e){
  let votee =e.target.parentNode.childNodes[4].lastChild.innerText;
let iuu=document.querySelector('#indexusername');
console.log(iuu.innerText);
  jQuery.ajax({

    type: 'POST',
    url: 'index_voteupdate.php',
    data: {use:iuu.innerText,pid:e.target.parentNode.childNodes[3].lastChild.innerText},
    success:function(data){
      console.log(data);
      if(data=='NV'){
        alert("vote not updated");
      }
      else if(data=='AV'){
        alert("already voted");

      }
     else if(Number.isInteger(parseInt(data))){
       alert("vote added");
       e.target.parentNode.childNodes[4].lastChild.innerText=data;
       window.location.reload();
       e.target.parentNode.childNodes[4].disabled=true;
  }
  else{
    alert("some issue");
  }
    }
```

```
  });
}
```

Backend code of index page(student):

Post table data is fetched. Vote data is updated in post table and post_vote table.

indexh_fetch.php

```
<?php
session_start();
$con =
mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_sp
ss_db');
$use=$_POST['use'];
$sql=mysqli_query($con,"UPDATE post SET post_status='removed' WHERE
DATEDIFF(CURDATE(), verified_date)>3 and verified_date IS NOT null");
$res=mysqli_query($con,"SELECT * FROM post WHERE post_status='verified'");
$count=mysqli_num_rows($res);
if($count>0){
   $data=array();
  while($mypost= mysqli_fetch_assoc($res)){
   array_push($data,$mypost);
  }
  echo json_encode($data);
}
else if($count==0){
    echo "zero";
}
else{
    echo "NF";
}

?>
```

index_voteupdate.php

```
<?php
session_start();
$con = mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_spss_db');
 $use=$_POST['use'];
$pid=$_POST['pid'];
$res=mysqli_query($con,"SELECT * FROM post WHERE post_status='verified'");
$count=mysqli_num_rows($res);
if($count>0)
   {
//if already voted the info will be in the table post_vote_tb    username and pv_id
$ress=mysqli_query($con,"SELECT * FROM `post_vote_tb` WHERE `username`='$use' AND `pv_id`='$pid';");
$countt=mysqli_num_rows($ress);
if($countt>0)
{
echo "AV";
}
else{
      $sql=mysqli_query($con,"SELECT post_votes FROM post WHERE post_id='$pid'");
      $pvote=implode(',mysqli_fetch_row($sql));
      $pvoten=(int)$pvote+1;
      $pvote=strval($pvoten);
      mysqli_query($con,"UPDATE `post` SET `post_votes`='$pvote' WHERE post_id='$pid'");
     // mysqli_query($con,"UPDATE `post` SET `verified_date`=CURRENT_TIMESTAMP WHERE post_pusername='$puse' and post_id='$pid'");
     mysqli_query($con,"INSERT INTO `post_vote_tb`(`username`, `pv_id`) VALUES ('$use','$pid')");
     echo $pvote;
}
   }
```

```
    else{
    echo "NV";
    }
    ?>
```

JavaScript code of Post page(student):

Post filled details are retrieved and validated.

post.php

Inside script

```
  var sess_use = "<?php echo $_SESSION['USERNAME']?>";
    function PostValidate(){
        console.log("Validation reached");
  let ptitle= document.getElementById("titp").value;
  let pdes= document.getElementById("desp").value;
    if(ptitle==""){
      alert("Please enter the title");
    }
   if(pdes==""){
      alert("Please enter the description");
    }
    else{
        let intent1 = processNLP(ptitle);
 let intent2 = processNLP(pdes);
if(intent1 && intent2)
{
  if(confirm("Are you sure you want to post"))
  {
    $.ajax({
                type: 'POST',
                url: 'stu_postingg.php',
                data: {tit:ptitle,des:pdes,use:sess_use},
```

```
                success: function(data) {
                  // console.log(data);
                   if(data=="A")
                   {
                      alert("This will be posted after it gets verified");
                      document.getElementById("titp").value="";
                      document.getElementById("desp").value="";
                   }
                   else if(data=="Alr"){
                      alert("Similar post available already.")
                   }
                   else if(data=="NAP")
                   {
                      alert("Too many posts. Only three most allowed per week to a user.");
                   }
                   else if(data=="explicit"){
                      alert("You are posting post against our guidelines. Please use right language and valid content");
                   }
                   else{
                      alert("Some issue");
                   }
                }
             });
  }
    }
     else{
                      alert("You are posting post against our guidelines. Please use right language and valid content");
                   }
    }
  }
```

```
function processNLP(input) {
  // Basic keyword-based intent detection
 let keywords = ['abuse', 'attack', 'hate', 'harassment', 'threat', 'violence',
  'spam', 'scam', 'fraud', 'phishing', 'malware', 'virus','shit','*',
  'illegal', 'contraband', 'explicit', 'porn', 'nudity', 'sexual',
  'racist', 'sexist', 'discrimination', 'bullying', 'intimidation',
  'drugs', 'weapons', 'terrorism', 'extremism', 'radicalization',
  'antisemitism', 'Islamophobia', 'discriminate', 'intolerance', 'hostility',
  'violence', 'violent', 'assault', 'attack', 'aggression', 'combat',
  'warning', 'alert', 'caution', 'advisory', 'danger', 'attention', 'note', 'hazard', 'forewarn',
'forealert', 'alarm', 'threat', 'imminent'];

  for (let keyword of keywords) {
  if (input.toLowerCase().includes(keyword.toLowerCase())) { // Convert keyword to
lowercase for case-insensitive comparison
      // Handle the presence of an abusive word (e.g., reject the input)
      console.log('Content contains abusive language. Please revise your input.');
      return false;
  }
}
   return true;
      }
```

Backend code of Post page(student):

Post data is sent to the Post table.

stu_posting.php

```
<?php
session_start();
$con =
mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_sp
ss_db');
$tit=strtolower($_POST['tit']);
$des=strtolower($_POST['des']);
$use=$_POST['use'];
```

```
$ptit='%'.$tit.'%';

$votes=0;

// Process input through basic NLP

$intent1 = processNLP($tit);

$intent2 = processNLP($des);


$status="to be verified";

//$pd=CURRENT_TIMESTAMP;

$pdd=null;

$vd=null;

$res=mysqli_query($con,"SELECT * FROM post WHERE post_username='$use' and
post_status!='removed'");

$count=mysqli_num_rows($res);

$res2=mysqli_query($con,"SELECT * FROM post WHERE post_status='verified' and
post_title LIKE '$ptit'");

$count2=mysqli_num_rows($res2);

if($intent1 && $intent2)

{

if($count2>0)

{

 echo("Alr");

}

else if($count>=0){

    if($count<3)

 {

   $sql=mysqli_query($con,"SELECT public_username FROM stud_account WHERE
stu_username='$use'");

$puse= implode("",(mysqli_fetch_row($sql)));

   // mysqli_query($con,"INSERT INTO `post`(`post_votes`, `post_status`, `post_title`,
`post_desc`, `post_date`, `posted_date`, `verified_date`, `post_username`,`post_pusername`)
VALUES
('$votes','$status','$tit','$des',CURRENT_TIMESTAMP,'$pdd','$vd','$use','$puse')");
```

```
   mysqli_query($con,"INSERT INTO `post`(`post_votes`, `post_status`, `post_title`,
`post_desc`, `post_date`, `verified_date`, `post_username`,`post_pusername`) VALUES
('$votes','$status','$tit','$des',CURRENT_TIMESTAMP,'$vd','$use','$puse')");
    echo("A");
 }
 else{
    echo("NAP");
 }
}
else{
    echo "NA";
}
}
else{
    echo "explicit";
}

function processNLP($input) {
    // Basic keyword-based intent detection
   $keywords = array(
    'abuse', 'attack', 'hate', 'harassment', 'threat', 'violence',
    'spam', 'scam', 'fraud', 'phishing', 'malware', 'virus','shit','*',
    'idiot', 'moron', 'stupid', 'fool', 'dummy', 'dumb', 'ignorant',
    'hate', 'hateful', 'bigotry', 'prejudice', 'xenophobia', 'homophobia',
    'antisemitism', 'Islamophobia', 'discriminate', 'intolerance', 'hostility',
    'violence', 'violent', 'assault', 'attack', 'aggression', 'combat',
    'warfare', 'harm', 'injury', 'kill', 'murder', 'bloodshed', 'gun', 'knife',
    'warning', 'alert', 'caution', 'advisory', 'danger', 'attention', 'note', 'hazard', 'forewarn',
'forealert', 'alarm', 'threat', 'imminent'
);
```

```
    foreach ($keywords as $keyword) {
        if (stripos($input, $keyword) !== false) {
            return false;
        }
    }

    // Default to the entire input if no keyword is found
    return $input;
}
?>
```

JavaScript code of Post page(handler):

Post component is created according to the data fetched and displayed in the page. The post can be verified or removed. The required post can be searched in a search bar.

posth.php

inside the script tag

```
let pppid2=0;
function remove_post()
{
 let pts = document.querySelectorAll(".post");
for (let i = 0; i < pts.length; i++) {
    pts[i].remove();
    console.log("removed",i);
}
}
 let pts = document.querySelectorAll(".post");
for (let i = 0; i < pts.length; i++) {
    pts[i].remove();
    console.log("removed",i);
}
function search_post(){
    let text=document.querySelector("#searcht").value;
    console.log(text);
```

```
    if(text!=""){
//alert("searching..");
let tempp="";let n=[];   // var sess_use =
var sess_use = "<?php echo $_SESSION['USERNAME']?>";
$.ajax({
type: 'POST',
url: 'posth_searchfetch.php',
data: {use:sess_use,st:text},
success:function(data){
if(data=="zero"){
    remove_post();
    let y = document.getElementById("no-post");
    y.innerText="no post's available on the current search";
    y.style.display = "block";
}
 else if(data=="NF"){
     alert("Could not update posts");
 }
 else{
     console.log("post alert");
   let y = document.getElementById("no-post");
    y.style.display = "none";
//can try by get element by classs name
remove_post();
let dataa=JSON.parse(data);
// console.log(dataa[0].post_id);
//console.log(dataa);
    // let pts=document.querySelectorAll(".post");
    // pts.remove();
```

```
    for(let ii=0;ii<dataa.length;ii++) {
    addposthsearch();
    }
    //use one more loop

    for(let ii=0;ii<dataa.length;ii++) {
        pppid2=pppid2+1;
    let s='#posttt'+pppid2.toString();
    console.log(s);
    let ppo=document.querySelector(s);
ppo.childNodes[0].lastChild.innerText=dataa[ii].post_title;
ppo.childNodes[1].lastChild.innerText=dataa[ii].post_desc;
ppo.childNodes[2].lastChild.innerText=dataa[ii].post_pusername;
ppo.childNodes[3].lastChild.innerText=dataa[ii].post_date;
ppo.childNodes[4].lastChild.innerText=dataa[ii].verified_date;
ppo.childNodes[5].lastChild.innerText=dataa[ii].post_id;
    }
 }
}
})
    }


}



   function load_toVerify(){
let tempp="";let n=[]; // var sess_use =
var sess_use = "<?php echo $_SESSION['USERNAME']?>";
$.ajax({
type: 'POST',
```

```
url: 'posth_fetch.php',
data: {use:sess_use},
success:function(data){
if(data=="zero"){
    let y = document.getElementById("no-post");
    y.innerText="Currently no post's available";
    y.style.display = "block";
}
 else if(data=="NF"){
     alert("Could not update posts");
 }
 else{
     console.log("post alert");
   let y = document.getElementById("no-post");
    y.style.display = "none";
//can try by get element by classs name
let dataa=JSON.parse(data);
// console.log(dataa[0].post_id);
//console.log(dataa);
    for(let ii=0;ii<dataa.length;ii++) {
    addposth();
    }

    let ppid=0;
    for(let ii=0;ii<dataa.length;ii++) {
        ppid=ppid+1;
    let s='#posttt'+ppid.toString();
    let ppo=document.querySelector(s);
   // console.log(ppo.childNodes[4]);
ppo.childNodes[0].lastChild.innerText=dataa[ii].post_title;
ppo.childNodes[1].lastChild.innerText=dataa[ii].post_desc;
```

```
ppo.childNodes[2].lastChild.innerText=dataa[ii].post_pusername;
ppo.childNodes[3].lastChild.innerText=dataa[ii].post_date;
ppo.childNodes[4].lastChild.innerText=dataa[ii].verified_date;
ppo.childNodes[4].style.display="none";
ppo.childNodes[5].lastChild.innerText=dataa[ii].post_id;
    }
 }
}
})
   }
```

main.js

```
function addposttvoteh(){
  let postvote = document.createElement("p");
  postvote.innerText ="Total Votes: ";
   postvote.setAttribute('id','post-vote-disp');
   postvote.classList.add('post-con-itm');
  divp.appendChild(postvote);
  let count = document.createElement("p");
  count.setAttribute('id','post-count-disp');
  count.classList.add('post-con-itm');
   postvote.appendChild(count);
}


var box ;
function addremove()
{
  divp.appendChild(box);

  postremove = document.createElement("button");
  postremove.innerText ="Remove";
   postremove.setAttribute('id','post-rem');
```

```
   postremove.classList.add('post-con-itm','submi');
    box.appendChild(postremove);
    postremove.addEventListener("click",e =>{
      removepostphp(e);
    });
}
function addpost(){
  ppid=ppid+1;
  let s='posttt'+ppid.toString();
   bodyp= document.getElementById('post-section');
  //add post
   divp = document.createElement("div");
 console.log("reached add post");
   divp.classList.add('post');
   divp.setAttribute('id',s);//indivitdual id created for the posts
  bodyp.appendChild(divp);
        //add title
        addtitle();
        //add desc
        adddesc();
        //add posteddate
        addposteddate();
        //add postid
        addpostid();
        //add postvote
        addposttvote();

}

function addposthindexh(){
  ppid=ppid+1;
```

```
 let s='posttt'+ppid.toString();
  bodyp= document.getElementById('post-section');
 //add post
  divp = document.createElement("div");
console.log("reached add post");
  divp.classList.add('post');
  divp.setAttribute('id',s);//indivitdual id created for the posts
 bodyp.appendChild(divp);
      //add title
      addtitle();
      //add desc
      adddesc();
      //add posteddate
      addposteddate();
      //add postvote
      addposttvoteh();

}

//posth code
var postverify;var ppid=0;var ppid2=0;
function verifypost(){
 box = document.createElement("div");
 box.classList.add('box');
 divp.appendChild(box);
  postverify = document.createElement("button");
  postverify.innerText ="Verify";
   postverify.setAttribute('id','post-verify');
   postverify.classList.add('post-con-itm','submi');
    box.appendChild(postverify);
    postverify.addEventListener("click",
```

```
    e => {addverify(e);},true);
    postverify.addEventListener("click",
    e => {addverifyphp(e);},true);
}

function addverifyphp(e){
jQuery.ajax({
  type: 'POST',
  url: '../handler/posth_verify.php',
  data:
{puse:e.target.parentNode.parentNode.childNodes[2].lastChild.innerText,pid:e.target.parentN
ode.parentNode.childNodes[5].lastChild.innerText},
  success:function(data){
  if(data=='V'){
    console.log("verify added");
  }
  else if(data=='NV'){
    alert("not verified");
  }
  else{
    alert("some issue");
  }
  }
});
}
function removepostphp(e){
  jQuery.ajax({

    type: 'POST',
    url: '../handler/posth_remove.php',
    data:
{puse:e.target.parentNode.parentNode.childNodes[2].lastChild.innerText,pid:e.target.parentN
ode.parentNode.childNodes[5].lastChild.innerText},
```

```
    success:function(data){
    if(data=='R'){
       alert("Removed post");
       e.target.parentNode.parentNode.remove();
    }
    else if(data=='NR'){
      alert("not removed");
    }
    else{
      alert("some issue");
    }
    }
  });
  }

function addverify(e){
    e.target.innerText ="Verified";
    e.target.style.pointerEvents="none";
}

function addposth(){
  ppid=ppid+1;
  let s='posttt'+ppid.toString();
   bodyp= document.getElementById('post-section');
  //add post
   divp = document.createElement("div");
 console.log("reached add post");
   divp.classList.add('post');
   divp.setAttribute('id',s);//indivitdual id created for the posts
  bodyp.appendChild(divp);
       //add title
```

```
        addtitle();
        //add desc
        adddesc();
        //add username
        adduse();
        //add postdate
        addpostdate();
        //add posteddate
        addposteddate();
        //add postid
        addpostid();
       //verify post
       verifypost();
       //add remove
       addremove();
}
function addposthsearch(){
  ppid2=ppid2+1;
  let s='posttt'+ppid2.toString();
   bodyp= document.getElementById('post-section');
  //add post
   divp = document.createElement("div");
 console.log("reached add post");
   divp.classList.add('post');
   divp.setAttribute('id',s);//indivitdual id created for the posts
  bodyp.appendChild(divp);
        //add title
        addtitle();
        //add desc
        adddesc();
        //add username
```

```
        adduse();
        //add postdate
        addpostdate();
        //add posteddate
        addposteddate();
        //add postid
        addpostid();
       //verify post
       verifypost();
       //add remove
       addremove();
}
```

Backend code of Post page(handler):

Post data is updated to the Post table. The required post in the search is retrieved.

posth_fetch.php

```
<?php
session_start();
$con =
mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_spss_db');
$use=$_POST['use'];
$res=mysqli_query($con,"SELECT * FROM post WHERE post_status='to be verified'");
$count=mysqli_num_rows($res);
if($count>0){
   $data=array();
  while($mypost= mysqli_fetch_assoc($res)){
   array_push($data,$mypost);
  }
  echo json_encode($data);
```

```
}
else if($count==0){
   echo "zero";
}
else{
   echo "NF";
}
?>
```

posth_remove.php

```
<?php
session_start();
$con = mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_spss_db');
$puse=$_POST['puse'];
$pid=$_POST['pid'];
$res=mysqli_query($con,"SELECT * FROM post WHERE post_status='to be verified' OR post_status='verified'");
$count=mysqli_num_rows($res);
if($count>0)
   {
      mysqli_query($con,"UPDATE `post` SET `post_status`='removed' WHERE post_pusername='$puse' and post_id='$pid'");
      echo "R";
   }
   else{
   echo "NR";
   }
   ?>
```

posth_searchfetch.php

```
<?php
```

```
session_start();

$con = mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_spss_db');

$use=$_POST['use'];

$st11="%".$_POST['st']." %";

$st22="% ".$_POST['st']." %";

$st33="% ".$_POST['st']."%";

$st44="%".$_POST['st']."s%";

 // Get user input

$st1 = mysqli_real_escape_string($con, $st11);

$st2 = mysqli_real_escape_string($con, $st22);

$st3 = mysqli_real_escape_string($con, $st33);

$st4 = mysqli_real_escape_string($con, $st44);

$res=mysqli_query($con,"SELECT * FROM post WHERE post_status = 'to be verified' AND (post_title LIKE '$st1' OR post_title LIKE '$st2' OR post_title LIKE '$st3' OR post_title LIKE '$st4')");

$count=mysqli_num_rows($res);

if($count>0){

   $data=array();

  while($mypost= mysqli_fetch_assoc($res)){

   array_push($data,$mypost);

  }

  echo json_encode($data);

}

else if($count==0){

    echo "zero";

}

else{

    echo "NF";

}?>
```

posth_verify.php

```
<?php
```

```
session_start();

$con =
mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_sp
ss_db');

$puse=$_POST['puse'];

$pid=$_POST['pid'];

$res=mysqli_query($con,"SELECT * FROM post WHERE post_status='to be verified' OR
post_status='removed'");

$count=mysqli_num_rows($res);

if($count>0)

   {

      mysqli_query($con,"UPDATE `post` SET `post_status`='verified' WHERE
post_pusername='$puse' and post_id='$pid'");

      mysqli_query($con,"UPDATE `post` SET `verified_date`=CURRENT_TIMESTAMP
WHERE post_pusername='$puse' and post_id='$pid'");

      echo "V";

   }

   else{

   echo "NV";

   }

   ?>
```

JavaScript code of Post page(admin):

Post component is created according to the data fetched and displayed in the page. The post can be removed.

The removed post can be deleted.

posta.php

```
var t=0;

let btn1=document.querySelector('#tab1');

let btn2=document.querySelector('#tab2');

btn1.addEventListener('click', function(){

btn1.style.border="2px solid rgb(176, 254, 254)";

btn2.style.border="none";
```

```
let pconta1=document.querySelector('#pconta1');
pconta1.style.display="block";
let pconta2=document.querySelector('#pconta2');
pconta2.style.display="none";
});
btn2.addEventListener('click', function(){
btn2.style.border="2px solid rgb(176, 254, 254)";
btn1.style.border="none";
let pconta1=document.querySelector('#pconta1');
pconta1.style.display="none";
let pconta2=document.querySelector('#pconta2');
pconta2.style.display="block";
});
  function load_toVerify(){
let tempp="";let n=[];  var sess_use = "<?php echo $_SESSION['USERNAME']?>";
$.ajax({
type: 'POST',
url: '../handler/posth_fetch.php',
data: {use:sess_use},
success:function(data){
if(data=="zero"){
   let y = document.getElementById("no-post2");
   y.style.display = "block";
}
 else if(data=="NF"){
    alert("Could not update posts");
 }
 else{
    console.log("post alert");
  let y = document.getElementById("no-post");
   y.style.display = "none";
```

```
//can try by get element by classs name
let dataa=JSON.parse(data);
// console.log(dataa[0].post_id);
//console.log(dataa);
    for(let ii=0;ii<dataa.length;ii++) {
    addposta();
    }
    //use one more loop
    let ppid=0;
    for(let ii=0;ii<dataa.length;ii++) {
        ppid=ppid+1;
    let s='#posttt'+ppid.toString();
    let ppo=document.querySelector(s);
   //  console.log(ppo.childNodes[0].lastChild);
ppo.childNodes[0].lastChild.innerText=dataa[ii].post_title;
ppo.childNodes[1].lastChild.innerText=dataa[ii].post_desc;
ppo.childNodes[2].lastChild.innerText=dataa[ii].post_pusername;
ppo.childNodes[3].lastChild.innerText=dataa[ii].post_date;
ppo.childNodes[4].lastChild.innerText=dataa[ii].verified_date;
ppo.childNodes[5].lastChild.innerText=dataa[ii].post_id;
    }
    t=ppid;
 }
}
})
   }
   function load_toRemove(){
let tempp="";let n=[];  var sess_use = "<?php echo $_SESSION['USERNAME']?>";
console.log("recahed LR");
$.ajax({
type: 'POST',
```

```
url: 'posta_fetchr.php',
data: {use:sess_use},
success:function(data){
//console.log(data);
if(data=="zero"){
    let y = document.getElementById("no-post2");
    y.style.display = "block";
}
 else if(data=="NF"){
     alert("Could not update posts");
 }
 else{
     console.log("post alert");
   let y = document.getElementById("no-post2");
    y.style.display = "none";
//can try by get element by classs name
let dataa=JSON.parse(data);
// console.log(dataa[0].post_id);
//console.log(dataa);
    for(let ii=0;ii<dataa.length;ii++) {
    addpostar();
    }
    //use one more loop
   let ppid=t;//let j=0;
    for(let ii=0;ii<dataa.length;ii++) {
        ppid=ppid+1;
    let ss='#postttt'+ppid.toString();
    let ppo=document.querySelector(ss);
    console.log("check");
    //console.log(ppo);
ppo.childNodes[0].lastChild.innerText=dataa[ii].post_title;
```

```
ppo.childNodes[1].lastChild.innerText=dataa[ii].post_desc;
ppo.childNodes[2].lastChild.innerText=dataa[ii].post_pusername;
ppo.childNodes[3].lastChild.innerText=dataa[ii].post_date;
ppo.childNodes[4].lastChild.innerText=dataa[ii].verified_date;
ppo.childNodes[5].lastChild.innerText=dataa[ii].post_id;
    }
 }
}
})
   }
```

main.js

```
function addposta(){//posta
  ppid=ppid+1;
  let s='posttt'+ppid.toString();
   //body= document.getElementById('post-section');
   bodyp= document.getElementById('pconta1');
  //add post
   divp = document.createElement("div");
 console.log("reached add post");
   divp.classList.add('post');
   divp.setAttribute('id',s);//indivitdual id created for the posts
  bodyp.appendChild(divp);
        //add title
        addtitle();
        //add desc
        adddesc();
        //add username
        adduse();
        //add postdate
        addpostdate();
        //add posteddate
```

```
        addposteddate();
        //add postid
        addpostid();
       //add remove
       addremovea();
}

function addpostaa(){//indexa
  ppid=ppid+1;
  let s='posttt'+ppid.toString();
   bodyp= document.getElementById('post-section');
  //add post
   divp = document.createElement("div");
 console.log("reached add post");
   divp.classList.add('post');
   divp.setAttribute('id',s);//indivitdual id created for the posts
  bodyp.appendChild(divp);
        //add title
        addtitle();
        //add desc
        adddesc();
        //add username
        adduse();
        //add postdate
        addpostdate();
        //add posteddate
        addposteddate();
        //add postid
        addpostid();
        //add postvote
        addposttvoteh();
```

```
      //add remove
      addremovea();
}
function addremoveR()
{
  divp.appendChild(box);
  postremove = document.createElement("button");
  postremove.innerText ="Delete";
   postremove.setAttribute('id','post-rem');
   postremove.classList.add('post-con-itm','submi');
    box.appendChild(postremove);
    postremove.addEventListener("click",e =>{
      deletepostphp(e);
    });
}
function addremovea()
{
   box = document.createElement("div");
   box.classList.add('box');
  divp.appendChild(box);
  postremove = document.createElement("button");
  postremove.innerText ="Remove";
   postremove.setAttribute('id','post-rem');
   postremove.classList.add('post-con-itm','submi');
  box.style.alignSelf ="center";
    box.appendChild(postremove);
    postremove.addEventListener("click",e =>{
      removepostphp(e);
    });
}
  function deletepostphp(e){
```

```
   jQuery.ajax({


     type: 'POST',
     url: 'posta_delete.php',
     data:
{puse:e.target.parentNode.parentNode.childNodes[2].lastChild.innerText,pid:e.target.parentN
ode.parentNode.childNodes[5].lastChild.innerText},
     success:function(data){
     if(data=='R'){
       alert("post deleted");
       e.target.parentNode.parentNode.remove();
     }
     else if(data=='NR'){
      alert("not deleted");
     }
     else{
      alert("some issue");
     }
     }
   });
   }
function addpostar(){
 ppid=ppid+1;
 let s='postttt'+ppid.toString();
  bodyp= document.getElementById('pconta2');
 //add post
  divp = document.createElement("div");
console.log("reached add post");
  divp.classList.add('post');
  divp.setAttribute('id',s);//indivitdual id created for the posts
 bodyp.appendChild(divp);
      //add title
```

```
        addtitle();
        //add desc
        adddesc();
        //add username
        adduse();
        //add postdate
        addpostdate();
        //add posteddate
        addposteddate();
        //add postid
        addpostid();
       //verify post
       verifypost();
       //add remove
       addremoveR();
}



function addposttvotea(){
  box = document.createElement("div");
  box.classList.add('box');
 divp.appendChild(box);
  let postvote = document.createElement("button");
  postvote.innerText ="Vote";
  postvote.setAttribute('id','post-verify');
  postvote.classList.add('post-con-itm','submi');
  box.appendChild(postvote);
  let count = document.createElement("div");
  count.setAttribute('id','post-count');
```

```
  count.classList.add('post-con-itm');
  postvote.appendChild(count);
  postvote.addEventListener("click",e =>{
    countincphp(e);
  });
}
```

Backend code of Post page(admin):

Post data is updated to the Post table. Post data is deleted from the Post table.

posta_fetchr.php

```
<?php
session_start();
$con = mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_spss_db');
$use=$_POST['use'];
$res=mysqli_query($con,"SELECT * FROM post WHERE post_status='removed'");
$count=mysqli_num_rows($res);
if($count>0){
   $data=array();
  while($mypost= mysqli_fetch_assoc($res)){
   array_push($data,$mypost);
  }
  echo json_encode($data);
}
else if($count==0){
    echo "zero";
}
else{
    echo "NF";
}
?>
```

posta_delete.php

```
<?php
session_start();
$con =
mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_sp
ss_db');
$puse=$_POST['puse'];
$pid=$_POST['pid'];
$res=mysqli_query($con,"SELECT * FROM post WHERE post_status='removed'");
$count=mysqli_num_rows($res);
if($count>0)
   {
      mysqli_query($con,"DELETE FROM `post` WHERE  post_id='$pid'");
      echo "R";
   }
   else{
   echo "NR";
   }
   ?>
```

JavaScript code of Manage users page(admin):

The user data can be added, edited in a module, and deleted. Validation is performed on adding and editing.

updateusers.php

```
inside script
var dataa;
function addDetailsphpa(e){
   console.log("deatils added a");
  //  console.log(e.target.parentNode.childNodes[1].childNodes[3].value);
let use=e.target.parentNode.childNodes[1].childNodes[3].value;
let mai=e.target.parentNode.childNodes[3].childNodes[3].value;
let pas=e.target.parentNode.childNodes[5].childNodes[3].value;
// if(pas!=""){
```

```
// if(use!=""){
 var regexu=/^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d_]{10,20}$/;
var regexp=/^(?=.*[0-9])(?=.*[!@#$%^&*])[a-zA-Z0-9!@#$%^&*]{6,16}$/;
if(use.length!=0 && regexu.test(use))
{
if(pas.length!=0 && regexp.test(pas))
{
        if(mai!="")
        {
           $.ajax({
              type: 'POST',
              url: 'adad_add.php',
              data: {use:use,e_mail:mai,pas:pas},
              success: function(data) {
                // console.log(data);
                 if(data=="NU")
                 {
                    console.log("Existing user");
                    alert("User created successfully");
                    window.location.reload();
                 }
                 else if(data=="EUEU"){
                   alert("Already registered");
                   }
                 else if(data=="MU")
                 {
                    alert("email already exist");
                 }
                 else{
                    alert("something went wrong");
                 }
```

```
                },
                error: function() {
                    console.log(response.status);
                },
            })
        }
        else{
      alert("fill email");
   }
        }

else{
   alert("invalid password");
}
}
else{
   alert("invalid username");
}

}

function addDetailsphph(e){
   console.log("deatils added h");
let use=e.target.parentNode.childNodes[1].childNodes[3].value;
let mai=e.target.parentNode.childNodes[3].childNodes[3].value;
let pas=e.target.parentNode.childNodes[5].childNodes[3].value;
// if(pas!=""){
// if(use!=""){
   var regexu=/^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d_]{10,20}$/;
var regexp=/^(?=.*[0-9])(?=.*[!@#$%^&*])[a-zA-Z0-9!@#$%^&*]{6,16}$/;
if(use.length!=0 && regexu.test(use))
```

```
{
if(pas.length!=0 && regexp.test(pas))
{
        if(mai!="")
        {
           $.ajax({
              type: 'POST',
              url: 'adha_add.php',
              data: {use:use,e_mail:mai,pas:pas},
              success: function(data) {
                 //console.log(data);
                 if(data=="NU")
                 {
                    console.log("Existing user");
                    alert("User created successfully");
                    window.location.reload();
                 }
                 else if(data=="EUEU"){
                   alert("Already registered");
                   }
                    else if(data=="MU")
                 {
                    alert("email already exist");
                 }
                 else{
                    alert("something went wrong");
                    //alert(data);
                 }
              },
              error: function() {
                 console.log(response.status);
```

```
                },
            })
        }
        else{
      alert("fill email");
   }
   }
else{
   alert("invalid password");
}
}
else{
   alert("invalid username");
}
}
function addDetailsphp(e){
//console.log(e.target.parentNode.childNodes);
let use=e.target.parentNode.childNodes[1].childNodes[3].value;
let cid=e.target.parentNode.childNodes[3].childNodes[3].value;
let rno=e.target.parentNode.childNodes[5].childNodes[3].value;
let mai=e.target.parentNode.childNodes[7].childNodes[3].value;
let pas=e.target.parentNode.childNodes[9].childNodes[3].value;
//console.log(use,cid, rno,mai,pas);
var regexu=/^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d_]{10,20}$/;
var regexp=/^(?=.*[0-9])(?=.*[!@#$%^&*])[a-zA-Z0-9!@#$%^&*]{6,16}$/;
if(use.length!=0 && regexu.test(use))
{
if(pas.length!=0 && regexp.test(pas))
{

if(cid!="")
```

```
{
   if(rno!="")
   {
      if(mai!="")
      {
         $.ajax({
            type: 'POST',
            url: 'adst_add.php',
            data: {use:use,control_id:cid,roll_no:rno,e_mail:mai,pas:pas},
            success: function(data) {
              // console.log(data);
               if(data=="NU")
               {
                 // console.log("Existing user");
                  alert("User created successfully");
                  window.location.reload();
               }
               else if(data=="EUEU"){
                 alert("Already registered");
                 }

               else if(data=="CU")
               {
                  alert("control id already exist");
               }
               else if(data=="RU")
               {
                  alert("roll no already exist");
               }
               else if(data=="MU")
               {
```

```
                alert("email already exist");
            }
            else if(data=="invalidc")
            {
                alert("invalid Control Id");
            }
            else if(data=="invalidr")
            {
                alert("invalid roll no");
            }
            else if(data=="invalide")
            {
                alert("invalid email");
            }
            else{
                alert("something went wrong");
                //alert(data);
            }
        },
        error: function() {
            console.log(response.status);
        },
      })
    }
    else{
  alert("fill email");
}
  }
  else{
  alert("fill roll no");
}
```

```
    }
    else{
        alert("fill control id");
    }

 }
else{
    alert("invalid password");
}
}
else{
    alert("invalid username");
}

}


///chnages to be done
function addDetailssphp(e){

let cid=e.target.parentNode.childNodes[1].childNodes[3].value;
let mai=e.target.parentNode.childNodes[3].childNodes[3].value;
let rno=e.target.parentNode.childNodes[5].childNodes[3].value;//here 5th is password but at client side in roll no

    if(cid!=="")
    {

          if(mai!=="")
           {
              if(rno!=="")
       {
```

```
$.ajax({
    type: 'POST',
    url: 'adsts_add.php',
    data: {control_id:cid,roll_no:rno,e_mail:mai},
    success: function(data) {
      // console.log(data);
       if(data=="NU")
       {
         // console.log("Existing user");
          alert("User created successfully");
          window.location.reload();
       }
       else if(data=="EUEU"){
         alert("Already registered");
         }

       else if(data=="CU")
       {
          alert("control id already exist");
       }
       else if(data=="RU")
       {
          alert("roll no already exist");
       }
       else if(data=="MU")
       {
          alert("email already exist");
       }
       else{
          alert("something went wrong");
          alert(data);
```

```
                }
            },
            error: function() {
                console.log(response.status);
            },
        })
    }
    else{
    alert("fill roll no");
 }


  }
    else{
    alert("fill email");
    }
}
else{
    alert("fill control id");
}



}

function addDetailsa(){
console.log("modal for admin created");
const openModalButtons = document.querySelector('.data-open-modal4');
const closeModalButtons = document.querySelector('#data-close-modal4');
const overlay= document.getElementById('overlay');
 openModalButtons.addEventListener('click',()=>{
    const modal = document.querySelector('.modal4');
    openModall4(modal);
```

```
  });

overlay.addEventListener('click',()=>{
  const modal = document.querySelector('.modal4.active4');
    closeModall4(modal);
});
closeModalButtons.addEventListener('click',()=>{
  const modal = document.querySelector('.modal4');
  closeModall4(modal);
  });

let addf=document.querySelector('#addfa');
if(addf!=null)
addf.addEventListener('click',e=>{
    addDetailsphpa(e);
});
function openModall4(modal){
  if(modal==null) return
  modal.classList.add('active4');
  overlay.classList.add('active4');
}
function closeModall4(modal){
  if(modal==null) return
  modal.classList.remove('active4');
  overlay.classList.remove('active4');
}
}

function addDetailsh(){
console.log("modal for handler created");
const openModalButtons = document.querySelector('.data-open-modal3');
```

```
const closeModalButtons = document.querySelector('#data-close-modal3');
const overlay= document.getElementById('overlay');
 openModalButtons.addEventListener('click',()=>{
    const modal = document.querySelector('.modal3');
    openModall3(modal);
   });

overlay.addEventListener('click',()=>{
  const modal = document.querySelector('.modal3.active3');
    closeModall3(modal);
});
closeModalButtons.addEventListener('click',()=>{
  const modal = document.querySelector('.modal3');
  closeModall3(modal);
  });

let addf=document.querySelector('#addfh');
if(addf!=null)
addf.addEventListener('click',e=>{
    addDetailsphph(e);
});
function openModall3(modal){
  if(modal==null) return
  modal.classList.add('active3');
  overlay.classList.add('active3');
}
function closeModall3(modal){
  if(modal==null) return
  modal.classList.remove('active3');
  overlay.classList.remove('active3');
}
```

```
}

 function addDetails(){
const openModalButtons = document.querySelector('.data-open-modal2');
const closeModalButtons = document.querySelector('#data-close-modal2');
const overlay= document.getElementById('overlay');
 openModalButtons.addEventListener('click',()=>{
    const modal = document.querySelector('.modal2');
    openModall(modal);
   });
overlay.addEventListener('click',()=>{
  const modal = document.querySelector('.modal2.active2');
    closeModall(modal);
});
closeModalButtons.addEventListener('click',()=>{
  const modal = document.querySelector('.modal2');
  closeModall(modal);
  });
let addf=document.querySelector('#addf');
if(addf!=null){
addf.addEventListener('click',e=>{
    addDetailsphp(e);
});
}

function openModall(modal){
  if(modal==null) return
  modal.classList.add('active2');
  overlay.classList.add('active2');
}
function closeModall(modal){
```

```
  if(modal==null) return
  modal.classList.remove('active2');
  overlay.classList.remove('active2');
}
 }
 function addDetailss(){
const openModalButtons = document.querySelector('.data-open-modal5');
const closeModalButtons = document.querySelector('#data-close-modal5');
const overlay= document.getElementById('overlay');
 openModalButtons.addEventListener('click',()=>{
    const modal = document.querySelector('.modal5');

    openModalls(modal);
   });
// openModalButtons.addEventListener('click',()=>cosole.log("clcikedd"));
overlay.addEventListener('click',()=>{
  const modal = document.querySelector('.modal5.active5');
    closeModalls(modal);
});
closeModalButtons.addEventListener('click',()=>{
  const modal = document.querySelector('.modal5');
  closeModalls(modal);
  });
let addf=document.querySelector('#addfs');
if(addf!=null){
addf.addEventListener('click',e=>{
    addDetailssphp(e);
});
}

function openModalls(modal){
```

```
  if(modal==null) return
  modal.classList.add('active5');
  overlay.classList.add('active5');

}
function closeModalls(modal){
  if(modal==null) return
  modal.classList.remove('active5');
  overlay.classList.remove('active5');
}
 }
 function changedDetailsa(){
    console.log("changeda");
    let  mail = document.querySelector('#aemcontentHoldermai');
   let  user = document.querySelector('#aemcontentHolderusee');
 //  let  password = document.querySelector('#aemcontentHolderpas');
    var regexu=/^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d_]{10,20}$/;
var regexp=/^(?=.*[0-9])(?=.*[!@#$%^&*])[a-zA-Z0-9!@#$%^&*]{6,16}$/;
if(user.innerText.length!=0 && regexu.test(user.innerText))
{
// if(password.value.length!=0 && regexp.test(password.value))
// {
     $.ajax({

                        type: 'POST',
                        url: 'adad_change.php',
                        data: {use:user.innerText,mai:mail.value},
                        success:function(data){
                        // console.log(data);
                         if(data=="P"){
                            alert("Details have been updated. Please close and refresh once.");
```

```
                        window.location.reload();
                    }
                    else if(data=="NF"){
                        alert("Could not update details");
                    }
                    else{
                        alert("Some issue");
                    }
                    }
                    })
//                      }
// else{
//     alert("invalid password");
// }
}
else{
    alert("invalid username");
}
}

function changedDetailsh(){
    console.log("changedh");
    let  mail = document.querySelector('#hemcontentHoldermai');
   let  user = document.querySelector('#hemcontentHolderusee');
 //  let  password = document.querySelector('#hemcontentHolderpas');
    var regexu=/^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d_]{10,20}$/;
var regexp=/^(?=.*[0-9])(?=.*[!@#$%^&*])[a-zA-Z0-9!@#$%^&*]{6,16}$/;
if(user.innerText.length!=0 && regexu.test(user.innerText))
{
// if(password.value.length!=0 && regexp.test(password.value))
// {
```

```
    $.ajax({

                        type: 'POST',
                        url: 'adha_change.php',
                        data: {use:user.innerText,mai:mail.value},
                        success:function(data){
                        // console.log(data);
                         if(data=="P"){
                             alert("Details have been updated. Please close and refresh once.");
                            window.location.reload();
                         }

                         else if(data=="NF"){
                             alert("Could not update details");
                         }
                         else{
                             alert("Some issue");
                         }
                        }
                        })
//                          }
// else{
//     alert("invalid password");
// }
}
else{
    alert("invalid username");
}

}
function changedDetails(){
```

```
   let  cid = document.querySelector('#semcontentHoldercid');
   let  maill = document.querySelector('#semcontentHoldermai');
   let  rno = document.querySelector('#semcontentHolderrno');
   let  user = document.querySelector('#semcontentHolderusee');
  // let  password = document.querySelector('#semcontentHolderpas');

   var regexu=/^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d_]{10,20}$/;
//var regexp=/^(?=.*[0-9])(?=.*[!@#$%^&*])[a-zA-Z0-9!@#$%^&*]{6,16}$/;
if(user.innerText.length!=0 && regexu.test(user.innerText))
{
// if(password.value.length!=0 && regexp.test(password.value))
// {
// if(pas!=""){
// if(use!=""){

      if(rno.value!="")
      {
         if(maill.value!="")
         {
           $.ajax({
                    type: 'POST',
                    url: 'adst_change.php',
                    data:
{cid:cid.innerText,use:user.innerText,pas:password.value,mai:maill.value,rno:rno.value},
                    success:function(data){
                     console.log(data);
                     if(data=="P"){
                        alert("Details have been updated. Please close and refresh once.");
                       window.location.reload();
                     }
                //        else if(data=="EUEU"){
                //    alert("Already registered");
```

```
            //    }



            // else if(data=="RU")
            // {
            //     alert("roll no already exist");
            // }
            // else if(data=="MU")
            // {
            //     alert("email already exist");
            // }

            else if(data=="invalidr")
            {
                alert("invalid roll no");
            }
            else if(data=="invalide")
            {
                alert("invalid email");
            }
            else{
                alert("something went wrong");
                //alert(data);
            }


        })
        }
        else{
    alert("fill email");
}
    }
```

```
        else{
        alert("fill roll no");
    }
    }
else{
    alert("invalid username");
}
}

  function changedDetailss(){
  let  mail = document.querySelector('#ssemcontentHoldermai');
  let  rno = document.querySelector('#ssemcontentHolderrno');
  let  cid = document.querySelector('#ssemcontentHoldercid');
  let regno = document.querySelector('#ssemcontentHolderreg');
    $.ajax({
                        type: 'POST',
                        url: 'adsts_change.php',
                        data: {cid:cid.innerText,reg:regno.value,mai:mail.value,rno:rno.value},
                        success:function(data){
                         if(data=="P"){
                             alert("Details have been updated. Please close and refresh once.");
                            window.location.reload();
                         }
                         else if(data=="NF"){
                             alert("Could not update details");
                         }
                         else{
                             alert("Some issue");
                         }
                        }
                        })
```

```
}

function deleteDetailsa(e){
 console.log('deleteDetailsa');
 let user=e.target.parentNode.parentNode.parentNode.childNodes[0].innerText;
if(confirm('Are you sure you want to delete')){
 $.ajax({
type: 'POST',
url: 'adad_delete.php',
data: {use:user},
success:function(data){
// console.log(data);
 if(data=="P"){
     alert("Details have been deleted. Please close and refresh once if changes not visisble.");
    window.location.reload();

 }
 else if(data=="NF"){
     alert("Could not delete details");
 }
 else{
     alert("Some issue");
 }
}
})
}
}

function deleteDetailsh(e){
 console.log('deleteDetailsh');
 let user=e.target.parentNode.parentNode.parentNode.childNodes[0].innerText;
```

```
if(confirm('Are you sure you want to delete')){
    $.ajax({
   type: 'POST',
   url: 'adha_delete.php',
   data: {use:user},
   success:function(data){
    //console.log(data);
    if(data=="P"){
        alert("Details have been deleted. Please close and refresh once if changes not visisble.");
       window.location.reload();
    }
    else if(data=="NF"){
        alert("Could not delete details");
    }
    else{
        alert("Some issue");
    }
   }
   })
   }
   }
   function deleteDetails(e){
    console.log('deleteDetails');
let user=e.target.parentNode.parentNode.parentNode.childNodes[0].innerText;
if(confirm('Are you sure you want to delete')){
    $.ajax({
   type: 'POST',
   url: 'adst_delete.php',
   data: {use:user},
   success:function(data){
    //console.log(data);
```

```
    if(data=="P"){
        alert("Details have been deleted. Please close and refresh once if changes not visisble.");
       window.location.reload();
    }
    else if(data=="NF"){
        alert("Could not delete details");
    }
    else{
        alert("Some issue");
    }
   }
   })
   }
}
function deleteDetailss(e){

let cid=e.target.parentNode.parentNode.parentNode.childNodes[0].innerText;
//console.log('deleteDetails',cid);
if(confirm('Are you sure you want to delete')){
    $.ajax({
   type: 'POST',
   url: 'adsts_delete.php',
   data: {cid:cid},
   success:function(data){
    //console.log(data);
    if(data=="P"){
        alert("Details have been deleted. Please close and refresh once if changes not visisble.");
       window.location.reload();
    }
    else if(data=="NF"){
        alert("Could not delete details");
```

```
    }
    else{
        alert("Some issue");
    }
   }
   })
   }
}
function studentinfo()
{
   // console.log("studentinfo");
let st=document.querySelector("#studcont");
let ha=document.querySelector("#handcont");
let ad=document.querySelector("#admicont");
let st2=document.querySelector("#studcont2");
st2.style.display="none";
st.style.display="flex";
ha.style.display="none";
ad.style.display="none";
let trr=document.querySelectorAll('.tr');
if(trr.length!=0){
    let tdcon=document.querySelector('.tdcont');
    tdcon.remove();
}
 let tempp="";let n=[];  var sess_use = "<?php echo $_SESSION['USERNAME']?>";
$.ajax({
type: 'POST',
url: 'adst_fetch.php',
data: {use:sess_use},
success:function(data){
if(data=="zero"){
```

```
    alert('No student data available');
    let y = document.getElementById("no-post");
    y.style.display = "block";

}
 else if(data=="NF"){
     alert("some issue");
 }
 else{
     console.log("data alert");
dataa=JSON.parse(data);
//console.log(dataa);
let ppid=0;
let tdcontt=document.createElement("div");
tdcontt.classList.add('tdcont');
    for(let ii=0;ii<dataa.length;ii++) {
        ppid=ppid+1;
        let s='#posttt'+ppid.toString();
        let tr=document.createElement("div");
    tr.setAttribute("class","tr");
    tr.setAttribute('id',s);
    let th=document.createElement("div");
    th.setAttribute("id","tds");
   st.appendChild(tdcontt);
    tr.appendChild(th);
    th.innerText=dataa[ii].stu_username;
     th=document.createElement("div");
    th.setAttribute("id","tds");
    tr.appendChild(th);
    th.innerText=dataa[ii].stu_controlid;
    th.style.display='none';
```

```
 th=document.createElement("div");
th.setAttribute("id","tds");
tr.appendChild(th);
th.innerText=dataa[ii].stu_rollno;
th=document.createElement("div");
th.setAttribute("id","tds");
tr.appendChild(th);
th.innerText=dataa[ii].stu_email;
th=document.createElement("div");
th.setAttribute("id","tds");
tdcontt.appendChild(tr);
tr.appendChild(th);
let ebtn=document.createElement("button");
let edit=document.createElement("span");
edit.setAttribute("class","material-symbols-outlined");
ebtn.setAttribute("class","data-modal-targets");
ebtn.addEventListener("click",e=>{
   editProfile(e);}
);
ebtn.setAttribute("id","iconss");
edit.setAttribute("id","iconsss");
edit.innerHTML="edit";
ebtn.appendChild(edit);
th.appendChild(ebtn);
ebtn.style.marginLeft="1px";
ebtn.style.marginRight="8px";
let dbtn=document.createElement("button");
let deletee=document.createElement("span");
deletee.setAttribute("class","material-symbols-outlined");
dbtn.addEventListener("click",e=>{
   deleteDetails(e);}
```

```
    );
    deletee.setAttribute("id","iconsss");
    dbtn.setAttribute("id","iconss");
    deletee.innerHTML="delete";
    dbtn.appendChild(deletee);
    th.appendChild(dbtn);
    }
 }
}
})
 }

 //changes to be made here
 function studentinfo2()
{
   // console.log("studentinfo");
let st=document.querySelector("#studcont");
let ha=document.querySelector("#handcont");
let ad=document.querySelector("#admicont");
let st2=document.querySelector("#studcont2");
st2.style.display="flex";
st.style.display="none";
ha.style.display="none";
ad.style.display="none";
let trr=document.querySelectorAll('.tr');
if(trr.length!=0){
    let tdcon=document.querySelector('.tdcont');
    tdcon.remove();
}
 let tempp="";let n=[];  var sess_use = "<?php echo $_SESSION['USERNAME']?>";
$.ajax({
```

```
type: 'POST',
url: 'adsts_fetch.php',
data: {use:sess_use},
success:function(data){
if(data=="zero"){
    alert('No student data available');
    let y = document.getElementById("no-post");
    y.style.display = "block";

}
 else if(data=="NF"){
     alert("some issue");
 }
 else{
     console.log("data alert");
dataa=JSON.parse(data);
//console.log(dataa);
let ppid=0;
let tdcontt=document.createElement("div");
tdcontt.classList.add('tdcont');
    for(let ii=0;ii<dataa.length;ii++) {
        ppid=ppid+1;
        let s='#posttt'+ppid.toString();
        let tr=document.createElement("div");
    tr.setAttribute("class","tr");
    tr.setAttribute('id',s);
    let th=document.createElement("div");
    th.setAttribute("id","tdss");
   st2.appendChild(tdcontt);
    tr.appendChild(th);
    th.innerText=dataa[ii].control_id;
```

```
 th=document.createElement("div");
th.setAttribute("id","tdss");
tr.appendChild(th);
th.innerText=dataa[ii].roll_no;
// th.style.display='none';
 th=document.createElement("div");
th.setAttribute("id","tdss");
tr.appendChild(th);
th.innerText=dataa[ii].email;
th=document.createElement("div");
th.setAttribute("id","tdss");
tr.appendChild(th);
th.innerText=dataa[ii].registered_no;
th=document.createElement("div");
th.setAttribute("id","tdss");
tdcontt.appendChild(tr);
tr.appendChild(th);
let ebtn=document.createElement("button");
let edit=document.createElement("span");
edit.setAttribute("class","material-symbols-outlined");
ebtn.setAttribute("class","data-modal-targetss");
ebtn.addEventListener("click",e=>{
    editProfiles(e);}
);
ebtn.setAttribute("id","iconss");
edit.setAttribute("id","iconsss");
edit.innerHTML="edit";
ebtn.appendChild(edit);
th.appendChild(ebtn);
ebtn.style.marginLeft="1px";
ebtn.style.marginRight="8px";
```

```
    let dbtn=document.createElement("button");
    let deletee=document.createElement("span");
    deletee.setAttribute("class","material-symbols-outlined");
    dbtn.addEventListener("click",e=>{
        deleteDetailss(e);}
    );
    deletee.setAttribute("id","iconsss");
    dbtn.setAttribute("id","iconss");
    deletee.innerHTML="delete";
    dbtn.appendChild(deletee);
    th.appendChild(dbtn);
    }
 }
}
})
 }

function handlerinfo()
{
    console.log("handlerinfo");
    let st=document.querySelector("#studcont");
let ha=document.querySelector("#handcont");
let ad=document.querySelector("#admicont");
let st2=document.querySelector("#studcont2");
st2.style.display="none";
st.style.display="none";
ha.style.display="flex";
ad.style.display="none";
let trr=document.querySelectorAll('.trh');
if(trr.length!=0){
    let tdcon=document.querySelector('.tdconth');
```

```
    tdcon.remove();
}
 let tempp="";let n=[];  var sess_use = "<?php echo $_SESSION['USERNAME']?>";
$.ajax({
type: 'POST',
url: 'adha_fetch.php',
data: {use:sess_use},
success:function(data){
if(data=="zero"){
    alert('No handler data available');
    let y = document.getElementById("no-post");
    y.style.display = "block";
}
 else if(data=="NF"){
     alert("some issue");
 }
 else{
dataa=JSON.parse(data);
//console.log(dataa);
let ppid=0;
let tdcontt=document.createElement("div");
tdcontt.classList.add('tdconth');
    for(let ii=0;ii<dataa.length;ii++) {
        ppid=ppid+1;
        let s='#posttt'+ppid.toString();
        let tr=document.createElement("div");
    tr.setAttribute("class","trh");
    tr.setAttribute('id',s);
    let th=document.createElement("div");
    th.setAttribute("id","tdh");
   ha.appendChild(tdcontt);
```

```
 tr.appendChild(th);
 th.innerText=dataa[ii].hand_username;
 th=document.createElement("div");
 th.setAttribute("id","tdh");
 tr.appendChild(th);
 th.innerText=dataa[ii].hand_email;
 th=document.createElement("div");
 th.setAttribute("id","tdh");
 tdcontt.appendChild(tr);
 tr.appendChild(th);
let ebtn=document.createElement("button");
let edit=document.createElement("span");
 edit.setAttribute("class","material-symbols-outlined");
 ebtn.setAttribute("class","data-open-targeth");
 ebtn.addEventListener("click",e=>{
    editProfileh(e);}
 );
 ebtn.setAttribute("id","iconss");
 edit.setAttribute("id","iconsss");
 edit.innerHTML="edit";
 ebtn.appendChild(edit);
 th.appendChild(ebtn);
 ebtn.style.marginLeft="1px";
 ebtn.style.marginRight="8px";
 let dbtn=document.createElement("button");
 let deletee=document.createElement("span");
 deletee.setAttribute("class","material-symbols-outlined");
 dbtn.addEventListener("click",e=>{
    deleteDetailsh(e);}
 );
 deletee.setAttribute("id","iconsss");
```

```
    dbtn.setAttribute("id","iconss");
    deletee.innerHTML="delete";
    dbtn.appendChild(deletee);
    th.appendChild(dbtn);
    }
 }
}
})
}
function admininfo()
{
    //console.log("admininfo");
    let st=document.querySelector("#studcont");
let ha=document.querySelector("#handcont");
let ad=document.querySelector("#admicont");
let st2=document.querySelector("#studcont2");
st2.style.display="none";
st.style.display="none";
ha.style.display="none";
ad.style.display="flex";
let trr=document.querySelectorAll('.tra');
if(trr.length!=0){
    let tdcon=document.querySelector('.tdconta');
    tdcon.remove();
}
 let tempp="";let n=[];  var sess_use = "<?php echo $_SESSION['USERNAME']?>";
$.ajax({
type: 'POST',
url: 'adad_fetch.php',
data: {use:sess_use},
success:function(data){
```

```
if(data=="zero"){
    alert('No admin data available');
    let y = document.getElementById("no-post");
    y.style.display = "block";
}
 else if(data=="NF"){
     alert("some issue");
 }
 else{
     console.log("data alert");
dataa=JSON.parse(data);
//console.log(dataa);
let ppid=0;
let tdcontt=document.createElement("div");
tdcontt.classList.add('tdconta');
    for(let ii=0;ii<dataa.length;ii++) {
        ppid=ppid+1;
        let s='#posttt'+ppid.toString();
        let tr=document.createElement("div");
    tr.setAttribute("class","tra");
    tr.setAttribute('id',s);
    let th=document.createElement("div");
    th.setAttribute("id","tda");
   ad.appendChild(tdcontt);
    tr.appendChild(th);
    th.innerText=dataa[ii].admin_username;
    th=document.createElement("div");
    th.setAttribute("id","tda");
    tr.appendChild(th);
    th.innerText=dataa[ii].admin_email;
    th=document.createElement("div");
```

```
    th.setAttribute("id","tda");
    tdcontt.appendChild(tr);
    tr.appendChild(th);
   let ebtn=document.createElement("button");
   let edit=document.createElement("span");
    edit.setAttribute("class","material-symbols-outlined");
    ebtn.setAttribute("class","data-open-targeta");
    ebtn.addEventListener("click",e=>{
        editProfilea(e);}
    );
    ebtn.setAttribute("id","iconss");
    edit.setAttribute("id","iconsss");
    edit.innerHTML="edit";
    ebtn.appendChild(edit);
    th.appendChild(ebtn);
    ebtn.style.marginLeft="1px";
    ebtn.style.marginRight="8px";
    let dbtn=document.createElement("button");
    let deletee=document.createElement("span");
    deletee.setAttribute("class","material-symbols-outlined");
    dbtn.addEventListener("click",e=>{
        deleteDetailsa(e);}
    );
    deletee.setAttribute("id","iconsss");
    dbtn.setAttribute("id","iconss");
    deletee.innerHTML="delete";
    dbtn.appendChild(deletee);
    th.appendChild(dbtn);
    }
 }
}
```

```
})
}

function editProfilea(e){
    console.log("editprofilea");
   // console.log(e.target.parentNode.parentNode.parentNode.childNodes[0].innerText);
    const openModalButtons = document.querySelectorAll('.data-open-targeta');
const closeModalButtons = document.querySelectorAll('#data-close-buttona');
const overlay= document.getElementById('overlay');
openModalButtons.forEach(button =>
  button.addEventListener('click',()=>{
    const modala = document.querySelectorAll('.modala');
    openModala(modala);
}))
overlay.addEventListener('click',()=>{
  const modalsa = document.querySelectorAll('.modala.activea');
  modals.forEach(modala => {
    closeModala(modalsa)
})
})
closeModalButtons.forEach(button =>
  button.addEventListener('click',()=>{
  const modala = button.closest('.modala');
  closeModala(modala)
  }))
function openModala(){
  if(modala==null) return

  modala.classList.add('activea');
  overlay.classList.add('activea');
}
```

```
function closeModala(){
  if(modala==null) return
  modala.classList.remove('activea');
  overlay.classList.remove('activea');
}
//putting details in modal
let  username = document.querySelector('#aemcontentHolderusee');
username.innerText = e.target.parentNode.parentNode.parentNode.childNodes[0].innerText ;
let email = document.querySelector('#aemcontentHoldermai');
email.value = e.target.parentNode.parentNode.parentNode.childNodes[1].innerText;
//  let password = document.querySelector('#aemcontentHolderpas');
//  password.value =e.target.parentNode.parentNode.parentNode.childNodes[2].innerText;
}

function editProfileh(e){
    const openModalButtons = document.querySelectorAll('.data-open-targeth');
const closeModalButtons = document.querySelectorAll('#data-close-buttonh');
const overlay= document.getElementById('overlay');
openModalButtons.forEach(button =>
  button.addEventListener('click',()=>{
    const modalh = document.querySelectorAll('.modalh');
    openModalh(modalh);
}))
overlay.addEventListener('click',()=>{
  const modalsh = document.querySelectorAll('.modalh.activeh');
  modals.forEach(modalh => {
    closeModalh(modalsh)
})
})
closeModalButtons.forEach(button =>
  button.addEventListener('click',()=>{
```

```
  const modalh = button.closest('.modalh');
  closeModalh(modalh)
  }))
function openModalh(){
  if(modalh==null) return
  modalh.classList.add('activeh');
  overlay.classList.add('activeh');
}
function closeModalh(){
  if(modalh==null) return
  modalh.classList.remove('activeh');
  overlay.classList.remove('activeh');
}
//putting details in modal
let  username = document.querySelector('#hemcontentHolderusee');
username.innerText = e.target.parentNode.parentNode.parentNode.childNodes[0].innerText ;
let email = document.querySelector('#hemcontentHoldermai');
email.value = e.target.parentNode.parentNode.parentNode.childNodes[1].innerText;
//  let password = document.querySelector('#hemcontentHolderpas');
//  password.value =e.target.parentNode.parentNode.parentNode.childNodes[2].innerText;
}
function editProfile(e){
  //console.log(e.target.parentNode.parentNode.parentNode.childNodes[0].innerText);
const openModalButtons = document.querySelectorAll('.data-modal-targets');
const closeModalButtons = document.querySelectorAll('#data-close-buttons');
const overlay= document.getElementById('overlay');
openModalButtons.forEach(button =>
  button.addEventListener('click',()=>{
    const modal = document.querySelectorAll('.modal');
    openModal(modal);
  }))
```

```
overlay.addEventListener('click',()=>{
  const modals = document.querySelectorAll('.modal.active');
  modals.forEach(modal => {
    closeModal(modal)
})
})
closeModalButtons.forEach(button =>
  button.addEventListener('click',()=>{
  const modal = button.closest('.modal');
  closeModal(modal)
  }))
function openModal(){
  if(modal==null) return
  modal.classList.add('active');
  overlay.classList.add('active');
}
function closeModal(){
  if(modal==null) return
  modal.classList.remove('active');
  overlay.classList.remove('active');
}

//putting details in modal
let  username = document.querySelector('#semcontentHolderusee');
username.innerText = e.target.parentNode.parentNode.parentNode.childNodes[0].innerText;
let  controlid = document.querySelector('#semcontentHoldercid');
controlid.innerText = e.target.parentNode.parentNode.parentNode.childNodes[1].innerText;
let  rno = document.querySelector('#semcontentHolderrno');
rno.value = e.target.parentNode.parentNode.parentNode.childNodes[2].innerText;
let email = document.querySelector('#semcontentHoldermai');
email.value = e.target.parentNode.parentNode.parentNode.childNodes[3].innerText;
```

```
}
 function editProfiles(e){
    //let modal;

//console.log(e.target.parentNode.parentNode.parentNode.childNodes[0].innerText,"reached"
);
const openModalButtonss = document.querySelectorAll('.data-modal-targetss');
//console.log(openModalButtonss);
const closeModalButtonss = document.querySelectorAll('#data-close-buttonss');
const overlay= document.getElementById('overlay');
openModalButtonss.forEach(button =>
  button.addEventListener('click',()=>{

    const modals = document.querySelectorAll('.modals');

    openModalss(modals);
  }))
//openModalButtonss.addEventListener('click',()=>console.log("beforeclose"));
overlay.addEventListener('click',()=>{
  const modalss = document.querySelectorAll('.modals.actives');
  modalss.forEach(modals => {
    closeModalss(modals)
})
})
closeModalButtonss.forEach(button =>
  button.addEventListener('click',()=>{
  const modal = button.closest('.modals');
  closeModalss(modals)
  }))
function openModalss(){
  if(modals==null) return
  modals.classList.add('actives');
```

```
  //console.log("open");
  overlay.classList.add('actives');
}
function closeModalss(){
  if(modals==null) return
  modals.classList.remove('actives');
  //console.log("close");
  overlay.classList.remove('actives');
}

//putting details in modal

let  controlid = document.querySelector('#ssemcontentHoldercid');
controlid.innerText = e.target.parentNode.parentNode.parentNode.childNodes[0].innerText;
let  rno = document.querySelector('#ssemcontentHolderrno');
rno.value = e.target.parentNode.parentNode.parentNode.childNodes[1].innerText;

let email = document.querySelector('#ssemcontentHoldermai');
email.value = e.target.parentNode.parentNode.parentNode.childNodes[2].innerText;
 let password = document.querySelector('#ssemcontentHolderreg');
 password.value =e.target.parentNode.parentNode.parentNode.childNodes[3].innerText;
 }
```

Backend code of Manage users page(admin):

The added, edited, and deleted data of user(student) is updated to the respective table.

adst_add.php

```
<?php
session_start();
$con =
mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_sp
ss_db');
$cid=$_POST['control_id'];
$rno=$_POST['roll_no'];
```

```
$email=$_POST['e_mail'];
$use=$_POST['use'];
$pas=$_POST['pas'];
$head = mt_rand(1, 10000);
$headstr = strval($head);

$sarr = str_split($headstr);
for ($i = 0; $i < strlen($headstr); $i++) {
   $sarr[$i] = chr(ord($headstr[$i]) + 30);
}

$str = implode('', array_reverse($sarr));
$sarr = str_split($use);
for ($i = 0; $i < strlen($use); $i++) {
   $sarr[$i] = chr(ord($use[$i]) + 5);
}

$str = $str . implode('', array_reverse($sarr));
$str1 = substr($str, 0, strlen($str) / 2);
$str2 = substr($str, strlen($str) / 2);
$str = $str2 . $str1;
$puse=$str;
$res=mysqli_query($con,"SELECT * FROM stud_account WHERE stu_username='$use'");
$count=mysqli_num_rows($res);
if($count>0){
   echo "EUEU";//existing username
}
else{
   $ress1=mysqli_query($con,"SELECT * FROM stud_account WHERE
stu_controlid='$cid'");
   $countt1=mysqli_num_rows($ress1);
   $ress2=mysqli_query($con,"SELECT * FROM stud_account WHERE stu_rollno='$rno'");
```

```
    $countt2=mysqli_num_rows($ress2);
    $ress3=mysqli_query($con,"SELECT * FROM stud_account WHERE
stu_email='$email'");
    $countt3=mysqli_num_rows($ress3);
    $ress4=mysqli_query($con,"SELECT * FROM student WHERE control_id='$cid'");
    $countt4=mysqli_num_rows($ress4);
    $ress5=mysqli_query($con,"SELECT * FROM student WHERE roll_no='$rno'");
    $countt5=mysqli_num_rows($ress5);
    $ress6=mysqli_query($con,"SELECT * FROM student WHERE email='$email'");
    $countt6=mysqli_num_rows($ress6);
   if($countt1>0){
     echo "CU";
   }
   else if($countt2>0){
    echo "RU";
 }
  else if($countt3>0){
    echo "MU";
  }
   else if($countt4==0){
    echo "invalidc";
  }
   else if($countt5==0){
    echo "invalidr";
  }
   else if($countt6==0){
    echo "invalide";   }
   else{
$simple_string = "$pas";
//echo "Original String: " . $simple_string;
// Store the cipher method
$ciphering = "AES-128-CTR";
```

```
// Use OpenSSl Encryption method
$iv_length = openssl_cipher_iv_length($ciphering);
$options = 0;
// Non-NULL Initialization Vector for encryption
$encryption_iv = '1234567891011121';

// Store the encryption key
$encryption_key = "$puse";
// Use openssl_encrypt() function to encrypt the data
$encryption = openssl_encrypt($simple_string, $ciphering,
                              $encryption_key, $options, $encryption_iv);

   mysqli_query($con,"INSERT INTO `stud_account`(`stu_username`, `stu_email`, `stu_password`, `public_username`, `stu_controlid`, `stu_rollno`) VALUES ('$use','$email','$encryption','$puse','$cid','$rno')");
   mysqli_query($con,"UPDATE `student` SET `registered_no`='$head' WHERE control_id='$cid'");

   echo "NU";
  }
}
?>
```

adst_change.php

```
<?php
session_start();
//$con = mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_spss_db');
```

```
$con =
mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_sp
ss_db');

$cid=$_POST['cid'];

$use=$_POST['use'];

 $rno=$_POST['rno'];

$pas=$_POST['pas'];

$email=$_POST['mai'];

//$reg=$_POST['reg'];

$res=mysqli_query($con,"SELECT * FROM stud_account WHERE stu_controlid='$cid'");

$count=mysqli_num_rows($res);

if($count>0){

    $ress5=mysqli_query($con,"SELECT * FROM student WHERE roll_no='$rno'");

    $countt5=mysqli_num_rows($ress5);

    $ress6=mysqli_query($con,"SELECT * FROM student WHERE email='$email'");

    $countt6=mysqli_num_rows($ress6);


    if($countt5==0){

    echo "invalidr";

  }

   else if($countt6==0){

    echo "invalide";

  }

   else{

    mysqli_query($con,"UPDATE `student` SET `roll_no`='$rno' WHERE control_id='$cid'");

      mysqli_query($con,"UPDATE `student` SET `email`='$email' WHERE roll_no='$rno'");

    mysqli_query($con,"UPDATE `stud_account` SET `stu_rollno`='$rno' WHERE
stu_controlid='$cid'");

    mysqli_query($con,"UPDATE `stud_account` SET `stu_email`='$email' WHERE
stu_controlid='$cid'");

     mysqli_query($con,"UPDATE `stud_account` SET `stu_password`='$pas' WHERE
stu_controlid='$cid'");
```

```
    echo "P";
   }
}
else{
    echo "NF";
}
?>
```

adst_fetch.php

```
<?php
session_start();
$con = mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_spss_db');
$use=$_POST['use'];
$res=mysqli_query($con,"SELECT * FROM stud_account");
$count=mysqli_num_rows($res);
if($count>0){
   $data=array();
  while($mypost= mysqli_fetch_assoc($res)){
   array_push($data,$mypost);
  }
  echo json_encode($data);
}
else if($count==0){
    echo "zero";
}
else{
    echo "NF";
}
?>
```

adst_delete.php

```
<?php

session_start();

$con =
mysqli_connect('sql112.infinityfree.com','if0_34419009','dCXhuPfmbSMj','if0_34419009_sp
ss_db');

$cid=$_POST['cid'];

$res=mysqli_query($con,"SELECT * FROM `student` WHERE control_id='$cid'");

$count=mysqli_num_rows($res);

if($count>0){

    mysqli_query($con,"DELETE FROM `student` WHERE control_id='$cid'");

   echo("P");//existing username

}

else{

    echo "NF";}

?>
```

5.2.2 Coding Efficiency

I have tried to keep the codes as short as possible but functionalities and reliability are not compromised. Efficiency is an important aspect of the system as the usability by reducing the complexity. Wherever there was repetition of code, I used functions. Login page is common for both user and admin. So, the functions were the called instead of writing the whole code again and again.

5.3 Testing Approaches

The approach of testing used are functional testing and user-acceptance testing. The focus of functional testing is to verify that the software functions correctly according to the documented requirements Unlike functional testing, and user-acceptance testing which focuses on verifying technical requirements, evaluates the software's usability, accessibility, and overall user experience.

5.4 Unit Testing

For all users:

Login:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter username: franky_super77 Password: hi@123456	Logged in successfully	Correct credentials-student	Passed
2	Enter username: franky_super77 Password: stude@77	Invalid Password	Incorrect Password	Passed
3	Enter username: frankysuper7	Invalid username	Username does not exist	Passed
4	Enter username: null Password: hi@123456	Please enter details	Please enter username	Passed
5	Enter username: franky_super77 Password: null	Please enter details	Please enter password	Passed

Table 5.4.1: Login Test Case

Edit Profile:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	New Password: bye@12345	Details updated	Details have been updated	Passed
2	New Password: bye@	Enter valid password	Invalid password	Passed
3	Password: null	Please enter password	Invalid password	Passed

Table 5.4.2: Edit Profile Test Case

Contact Us:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Email: rujiihelpdesk@gmail.com Select Issue: Login issue Description: forgot password	Your mail has been sent	The form was submitted successfully	Passed
2	Email: rujigmail.com	Invalid email	Please include '@' in the email address	Passed
3	Email: null	Please enter details	Please fill out this field	Passed

Table 5.4.3: Contact Us Test Case

Forgot Password:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter username: franky_super77 Enter email: null	Password sent to mail. Please check spam.	Password sent to mail. Please check spam.	Passed
2	Enter username: null Enter_email: steven77thomas.2003@gmail.com	Password sent to mail. Please check spam.	Password sent to mail. Please check spam.	Passed
3	Enter username: franky_super77 Enter_email: steven77thomas.2003@gmail.com	Password sent to mail. Please check spam.	Password sent to mail. Please check spam.	Passed
4	Enter username: null Enter email: null	Please enter details	Please enter username	Passed

Table 5.4.4: Forgot Password Test Case

For Student:

Registration:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter control id:2019060866 Roll no: 4045A100 Email: rujiihelpdesk@gmail.com Otp: 467856 username: frankysuper78 password: bye#04566	Registered Successfully	Correct credentials	Passed
2	control id:201906086	Invalid Control id	control id does not exist	Passed
3	control id:2019060868	Already registered control id	Already registered	Passed
4	Roll no: 4045A222	Invalid Rollno	roll no does not match with control id	Passed
5	Email: new	Invalid email	Please include '@' in the email address	Passed
6.	Email: test@gmail.com	Already used mail	email does not match with control id	Passed
7.	Otp:43	Invalid otp	Incorrect otp	Passed
8	Enter username: frankysuper*77	Invalid username should not consist special characters, except underscore() and length should be	In username, No space or special character allowed, except underscore(_)	Passed

		between 10 and 20		
9	Enter username: franky_super77	Already registered Username	Already registered Username	Passed
10	Password: student@	Password should consist a letter, a digit and special character, and length should be between 8 and 30	In password, Minimum one number required	Passed
11	Enter control id: null Roll no: 4045A100 Email: rujiihelpdesk@gmail.com	Please enter details	fill control id	Passed
12	Enter control id: 2019060866 Roll no: null Email: rujiihelpdesk@gmail.com	Please enter details	fill roll no	Passed
13	Enter control id: 2019060866 Roll no: 4045A100 Email: null	Please enter details	fill email	Passed
14	Otp:null	Enter otp	Enter otp	Passed
15	username: null password: bye#04566	Username length must not be less than 10 characters	Username length must not be less than 10 characters	Passed
16	username: frankysuper78 password: null	Password should not be less than 8 characters	Correct credentials	Failed

Table 5.4.5: Registration Test Case

Send Post:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter Title: Filter non-availability Enter Description: Filter is not available in every floor of the college. It should be made available. So please vote for this post.	Post sent successfully, will be posted after verified.	This will be posted after it gets verified	Passed
2	Enter Title: This is title of 4th post Enter Description: This is description of 4th post.	Cannot post more than 3 post per week	Too many posts. Only three most allowed per week to a user.	Passed
3	Enter Title: dummy Enter Description: Filter is not available in every floor of the college. It should be made available. So please vote for this post.	You are posting post against our guidelines. Please use right language and valid content	You are posting post against our guidelines. Please use right language and valid content	Passed
4	Enter Title: null Enter Description: Filter is not available in every floor of the college. It should be made available. So please vote for this post.	Please enter details	Please enter title	Passed
5	Enter Title: Filter non-availability Enter Description: null	Please enter details	Please enter password	Passed

Table 5.4.6: Send Post Test Case

Home(Vote Post):

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Vote post	Voted	vote added	Passed
2	Vote same post again	You have already voted	already voted	Passed

Table 5.4.7: Vote Post Test Case

For handler and admin

Verify Post:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Verify post	Verified	Verified	Passed

Table 5.4.8: Verify Post Test Case

Search Post:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	washroom	Posts to be displayed which contains word 'washroom'	Same as expected output	Passed
2	null	no post's available on the current search	no post's available on the current search	Passed

Table 5.4.9: Search Post Test Case

Remove Post:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Remove post	Post removed	Removed post	Passed

Table 5.4.10: Remove Post Test Case

Delete Post:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Delete post	Post deleted	Post deleted	Passed

Table 5.4.11: Delete Post Test Case

For Admin:

Add student account:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter control id:2021060111 Roll no: 4045A111 Email: studentproblemsolversystem05@gmail.com username: student_1111 password: student@1111	User added	User created successfully	Passed
2	control id:201906086	Invalid Control id	Invalid Control id	Passed
3	control id:2019060866	Registered control id	control id already exist	Passed
4	Roll no: 4045A222	Invalid Rollno	invalid roll no	Passed
5	Email: new	Invalid email	invalid email	Passed

6.	Email: stephotos8@gmail.com	Already used mail	email already exist	Passed
7	Enter username: frankysuper*77	Invalid username, should not consist special characters and length should be between 10 and 20	Invalid username	Passed
8	Enter username: franky_super77	Registered Username	Already Registered	Passed
9	Password: student@	Password should consist a letter, a digit and special character, and length should be between 8 and 30	Invalid password	Passed
10	Enter control id:null Roll no: 4045A111 Email: studentproblemsolversystem05@gmail.com username: student_1111 password: student@1111	fill control id	User created successfully	Failed
11	Enter control id: 2021060111 Roll no: null Email: studentproblemsolversystem05@gmail.com	fill roll no	fill roll no	Passed

	username: student_1111 password: student@1111			
12	Enter control id:2021060111 Roll no: 4045A111 Email: null username: student_1111 password: student@1111	fill email	fill email	Passed
13	Enter control id:2021060111 Roll no: 4045A111 Email: studentproblemsolversystem05@gmail.com username: null password: student@1111	invalid username	invalid username	Passed
14	Enter control id:2021060111 Roll no: 4045A111 Email: studentproblemsolversystem05@gmail.com username: student_1111 password: null	invalid password	invalid password	Passed

Table 5.4.12: Add student account Test Case

Edit student account:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Roll no: 4045A045 Email: rujiihelpdesk3@gmail.com	User data updated	Details have been updated.	Passed
2	Roll no: 4045A222	Invalid Rollno	Invalid Rollno	Passed
3	Email: new	Invalid email	Invalid email	Passed
4	Roll no: null Email: null	Please enter details	invalid email	Passed

Table 5.4.13: Edit student account Test Case

Add handler:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter username: hand_123456 Email:handlerrr@gmail.com Password: hand@hand12	User added	User created successfully	Passed
2	Enter username: hand_	Invalid username, should not consist special characters and length should be between 10 and 20	Invalid username	Passed
3	Password: hand12	Password should consist a letter, a digit and special character, and length should be between 8 and 30	Invalid password	Passed
4	Enter username: null Email: null Password: null	Please enter details	Invalid username	Passed

Table 5.4.14: Add handler Test Case

Add Admin:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter username: admi_123456 Email: adminnn@gmail.com Password: admin@admi12	User added	User created successfully	Passed
2	Enter username: admi_	Invalid username, should not consist	Invalid username	Passed

		special characters and length should be between 10 and 20		
3	Password: admi12	Password should consist a letter, a digit and special character, and length should be between 8 and 30	Invalid password	Passed
4	Enter username: null Email: null Password: null	Please enter details	Invalid username	Passed

Table 5.4.15: Add Admin Test Case

Edit handler:

Test No	**Test Case Name**	**Expected Output**	**Actual Output**	**Remark**
1	Enter mail: handdd@gmail.com Password: hand@hand13	User data updated	Details have been updated.	Passed
2	Password: hand13	Password should consist a letter, a digit and special character, and length should be between 8 and 30	Invalid password	Passed
3	Enter mail: null Password: null	Please enter details	Invalid username	Passed

Table 5.4.16: Edit handler Test Case

Edit Admin:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter mail: adminnn@gmail.com Password: admi@admi13	User data updated	Details have been updated.	Passed
2	Password: admi13	Password should consist a letter, a digit and special character, and length should be between 8 and 30	Invalid password	Passed
3	Enter mail: null Password: null	Please enter details	Invalid username	Passed

Table 5.4.17: Edit admin Test Case

Delete user:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Delete user	User deleted	Details have been deleted.	Passed

Table 5.4.18: Delete user Test Case

5.5 Integration Testing

Integrating testing of Registration, Profile and Manage Users:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Registered action	User data updated in Profile of Student and Manage Users Module of Admin	Same as expected Output	Passed
2	New password Edit action	User password updated in Manage Users Module of Admin	Same as expected Output	Passed

Table 5.5.1: Registration and Profile Integrated Test Case

Integrating testing of Send Post and Post Module of Handler:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Send Post action	Post available on Post Module of Handler	Same as expected output	Passed

Table 5.5.2: Send Post Integrated Test Case

Integrating testing of Home(student) and Home (other users):

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Vote action	Vote count of post is updated in Home tabs of All users	Same as expected output	Passed

Table 5.5.3: Home(student)Integrated Test Case

Integrating testing of Post (Handler) and Home (other users):

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Verify action	The post is added to the Home tabs of All users	Same as expected output	Passed
2	Remove action	The post is added to the Post tab under 'removed post' section of Admin and removed from Home Page tab of All users	Same as expected output	Passed

Table 5.5.4: Verify and Remove Post Integrated Test Case

Integrating testing of Manage users, login, and Profile of student:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Add action	The added student is to be able to login and view details in profile	Same as expected output	Passed
2	Edit action	The student data is updated in the profile	Same as expected output	Passed
3	Delete action	The student will not be able to login	Same as expected output	Passed

Table 5.5.5: Add, Edit and Delete student account Integrated Test Case

Integrating testing of Manage users, login, and Profile of Handler:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Add action	The added handler is to be able to login and view details in profile	Same as expected output	Passed
2	Edit action	The handler data is updated in the profile	Same as expected output	Passed
3	Delete action	The handler will not be able to login	Same as expected output	Passed

Table 5.5.6: Add, Edit and Delete Handler Integrated Test Case

Integrating testing of Manage users, login, and Profile of Admin:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Add action	The added admin is to be able to login and view details in profile	Same as expected output	Passed
2	Edit action	The admin data is updated in the profile	Same as expected output	Passed
3	Delete action	The admin will not be able to login	Same as expected output	Passed

Table 5.5.7: Add, Edit and Delete Admin account Integrated Test Case

5.6 Modification and Implementation

Registration and Add admin modification were done.

Modification:

Test No	Test Case Name	Change in Code

1	Checking the null value validation of password field in registration	if(pass!=null)//extra nested outside block was created in javascript before sending data in ajax
		{
		//rest of the code
		}

2	Checking the null value validation of control id field in add admin section in manage users page.	if(cid!=null)//extra nested outside block was created in javascript before sending data in ajax
		{
		//rest of the code
		}

Table 5.6.1: Modification Table for registration(student) and add student of manage users(admin)

Retesting:

For Student:

Registration:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter control id:2019060866 Roll no: 4045A100 Email: rujiihelpdesk@gmail.com Otp: 467856 username: frankysuper78 password: bye#04566	Registered Successfully	Correct credentials	Passed
2	control id:201906086	Invalid Control id	control id does not exist	Passed
3	control id:2019060868	Already registered control id	Already registered	Passed

4	Roll no: 4045A222	Invalid Rollno	roll no does not match with control id	Passed
5	Email: new	Invalid email	Please include '@' in the email address	Passed
6.	Email: test@gmail.com	Already used mail	email does not match with control id	Passed
7.	Otp:43	Invalid otp	Incorrect otp	Passed
8	Enter username: frankysuper*77	Invalid username should not consist special characters, except underscore() and length should be between 10 and 20	In username, No space or special character allowed, except underscore(_)	Passed
9	Enter username: franky_super77	Already registered Username	Already registered Username	Passed
10	Password: student@	Password should consist a letter, a digit and special character, and length should be between 8 and 30	In password, Minimum one number required	Passed
11	Enter control id: null Roll no: 4045A100 Email: rujiihelpdesk@gmail.com	Please enter details	fill control id	Passed
12	Enter control id: 2019060866 Roll no: null Email: rujiihelpdesk@gmail.com	Please enter details	fill roll no	Passed

13	Enter control id: 2019060866 Roll no: 4045A100 Email: null	Please enter details	fill email	Passed
14	Otp:null	Enter otp	Enter otp	Passed
15	username: null password: bye#04566	Username length must not be less than 10 characters	Username length must not be less than 10 characters	Passed
16	username: frankysuper78 password: null	Password should not be less than 8 characters	Password should not be less than 8 characters	Passed

Table 5.6.2: Registration Test Case

Integrating testing of Registration, Profile and Manage Users:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Registered action	User data updated in Profile of Student and Manage Users Module of Admin	Same as expected Output	Passed
2	New password Edit action	User password updated in Manage Users Module of Admin	Same as expected Output	Passed

Table 5.6.3: Registration and Profile Integrated Test Case

For Admin:

Add student account:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Enter control id:2021060111 Roll no: 4045A111	User added	User created successfully	Passed

	Email: studentproblemsolversystem05@gmail.com username: student_1111 password: student@1111			
2	control id:201906086	Invalid Control id	Invalid Control id	Passed
3	control id:2019060866	Registered control id	control id already exist	Passed
4	Roll no: 4045A222	Invalid Rollno	invalid roll no	Passed
5	Email: new	Invalid email	invalid email	Passed
6.	Email: stephotos8@gmail.com	Already used mail	email already exist	Passed
7	Enter username: frankysuper*77	Invalid username, should not consist special characters and length should be between 10 and 20	Invalid username	Passed
8	Enter username: franky_super77	Registered Username	Already Registered	Passed
9	Password: student@	Password should consist a letter, a digit and special character,	Invalid password	Passed

		and length should be between 8 and 30		
10	Enter control id:null Roll no: 4045A111 Email: studentproblemsolversystem05@gmail.com username: student_1111 password: student@1111	fill control id	fill control id	Passed
11	Enter control id: 2021060111 Roll no: null Email: studentproblemsolversystem05@gmail.com username: student_1111 password: student@1111	fill roll no	fill roll no	Passed
12	Enter control id:2021060111 Roll no: 4045A111 Email: null username: student_1111 password: student@1111	fill email	fill email	Passed
13	Enter control id:2021060111 Roll no: 4045A111 Email: studentproblemsolversystem05@gmail.com username: null password: student@1111	invalid username	invalid username	Passed
14	Enter control id:2021060111 Roll no: 4045A111 Email: studentproblemsolversystem05@gmail.com username: student_1111 password: null	invalid password	invalid password	Passed

Table 5.6.4: Add student account Test Case

Integrating testing of Manage users, login, and Profile of student:

Test No	Test Case Name	Expected Output	Actual Output	Remark
1	Add action	The added student is to be able to login and view details in profile	Same as expected output	Passed
2	Edit action	The student data is updated in the profile	Same as expected output	Passed
3	Delete action	The student will not be able to login	Same as expected output	Passed

Table 5.6.5: Add, Edit and Delete student account Integrated Test Case

Chapter 6

Results and Discussions

6.1 Test Report

The testing part in project development is a very important phase. This phase helps to know whether all the functionalities are being performed the way that they are supposed to be executing.
The testing phase started with designing the test cases for each module as well as the designing process for integration test cases was performed. Each module was analysed and according to that test cases were formed. The test cases include input and the expected output to be seen after entering the values. After designing the test cases, the test cases were checked by entering the inputs and checking from different users if they are getting the expected output or not. If the expected outputs match the actual output, then the test cases were remarked to be successful or else, they were remarked as fail.
The input values of student while registering included controlid, rollno, email, otp, username and password, which on successfully being validated shows success message and if something incorrect, the issue is alerted to the user.
Not all values were tried and tested but the process made sure the system would be able to cope up with any values.
After performing all the test cases, there were no more errors. So, no further modifications were needed in the respective modules.

6.2 User Documentation

1. This is the first page of the website which gives, the new student option to sign up and existing user the option to sign in.

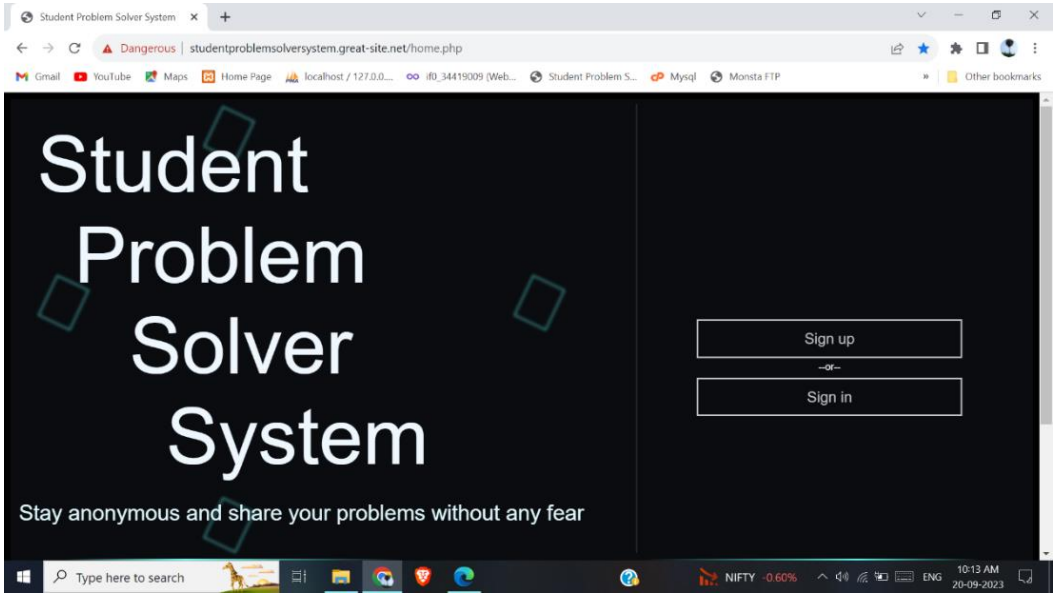


Figure 6.2.1: Home Page UI

2. The new user must register by filling their valid control id, roll no, email. An OTP would be send to the respective email. On entering correct OTP the user can enter valid username and password to create an account.

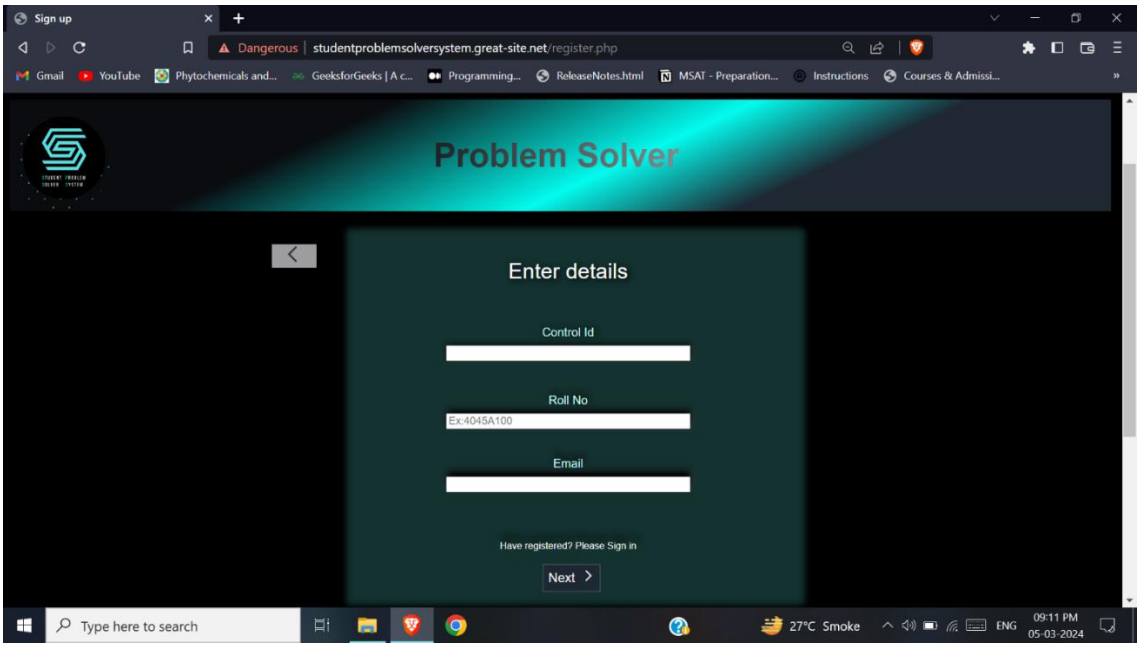


Figure 6.2.2.1: Register Page 1 UI

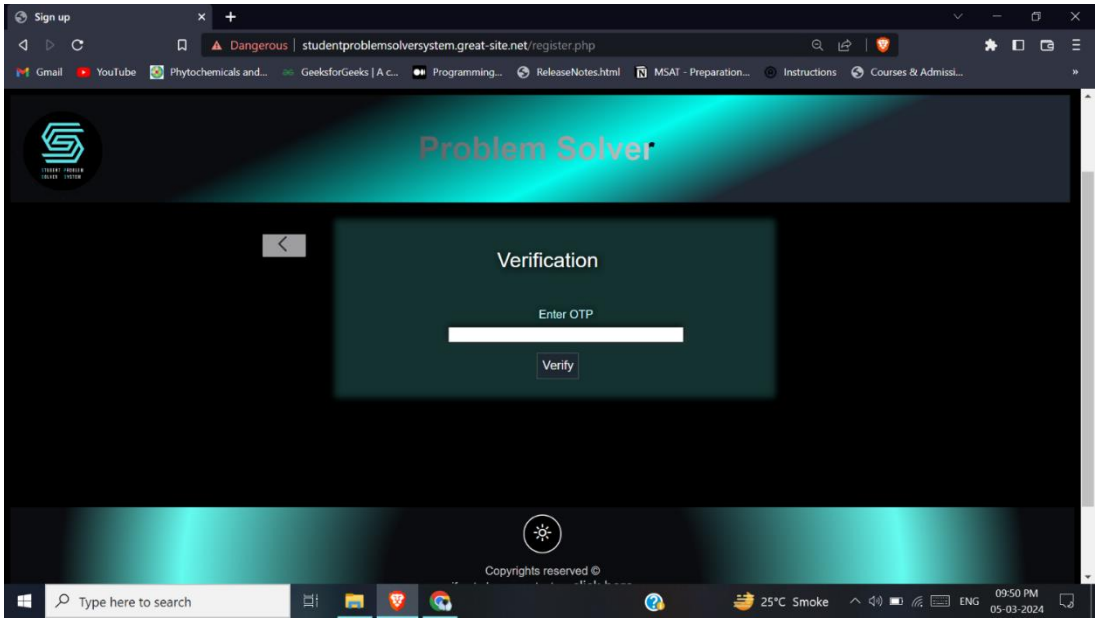


Figure 6.2.2.2: Register Page 2 UI

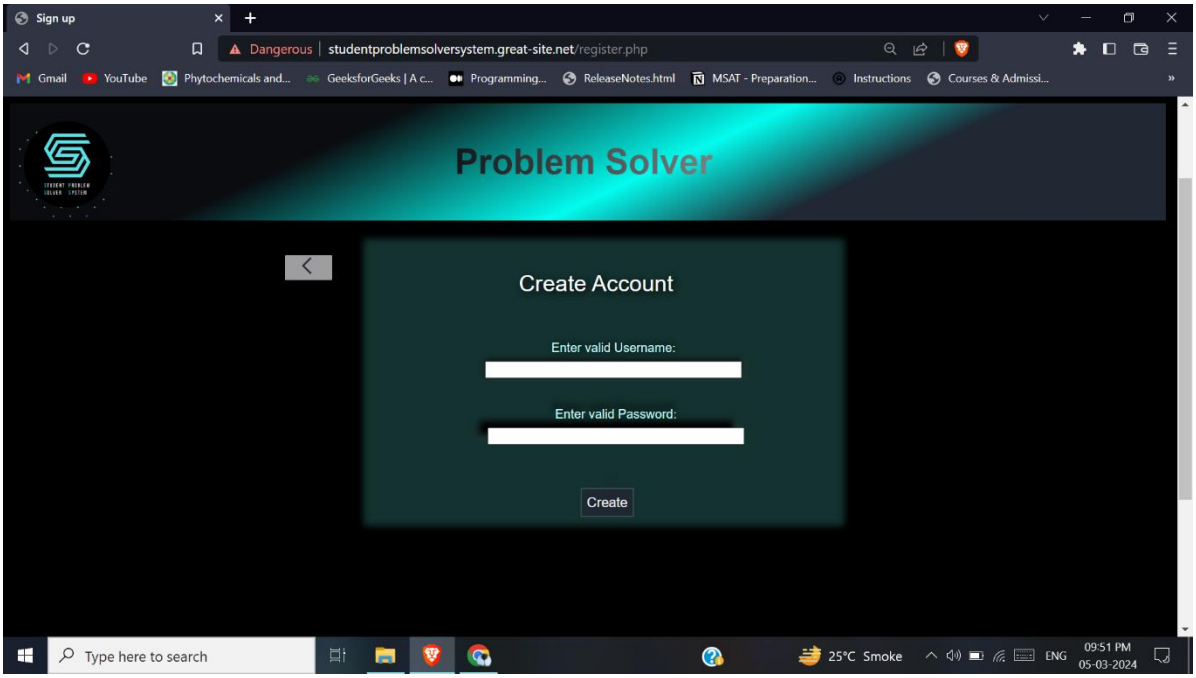


Figure 6.2.2.3: Register Page 3 UI

3. The registered users can login with their valid username and password.

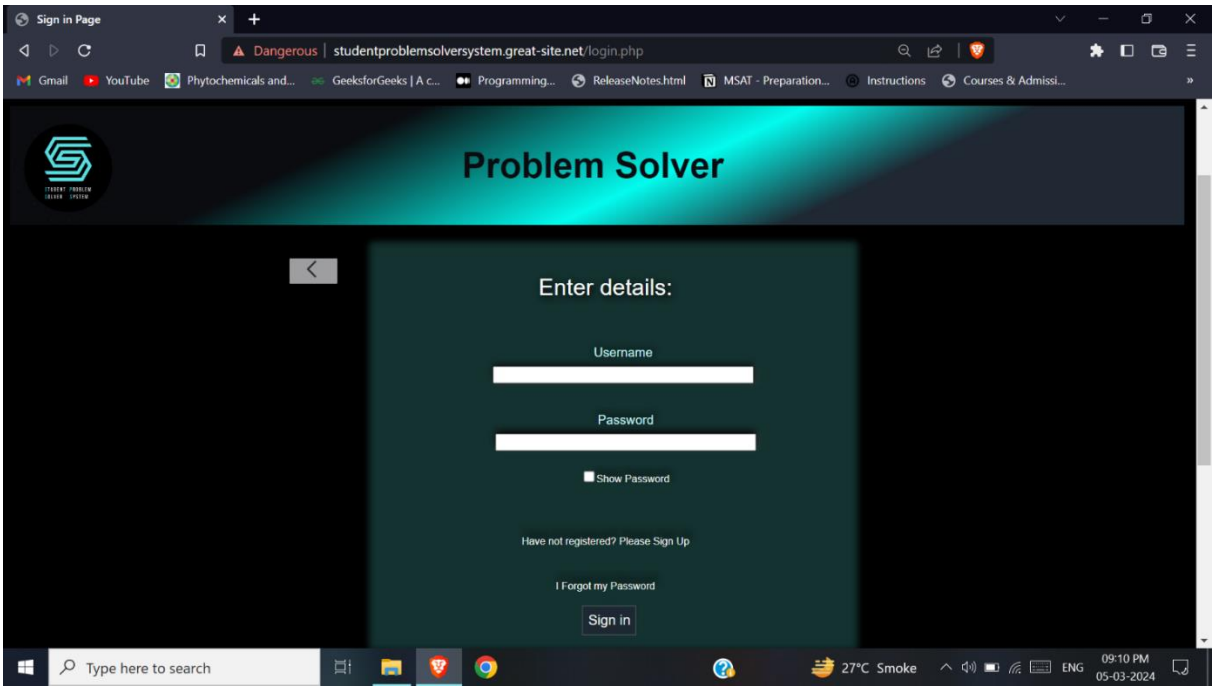


Figure 6.2.3: Login Page UI

Student:

4. If the student has logged in, this would be the first home page. Here, the student can view and vote the post according to their opinions.

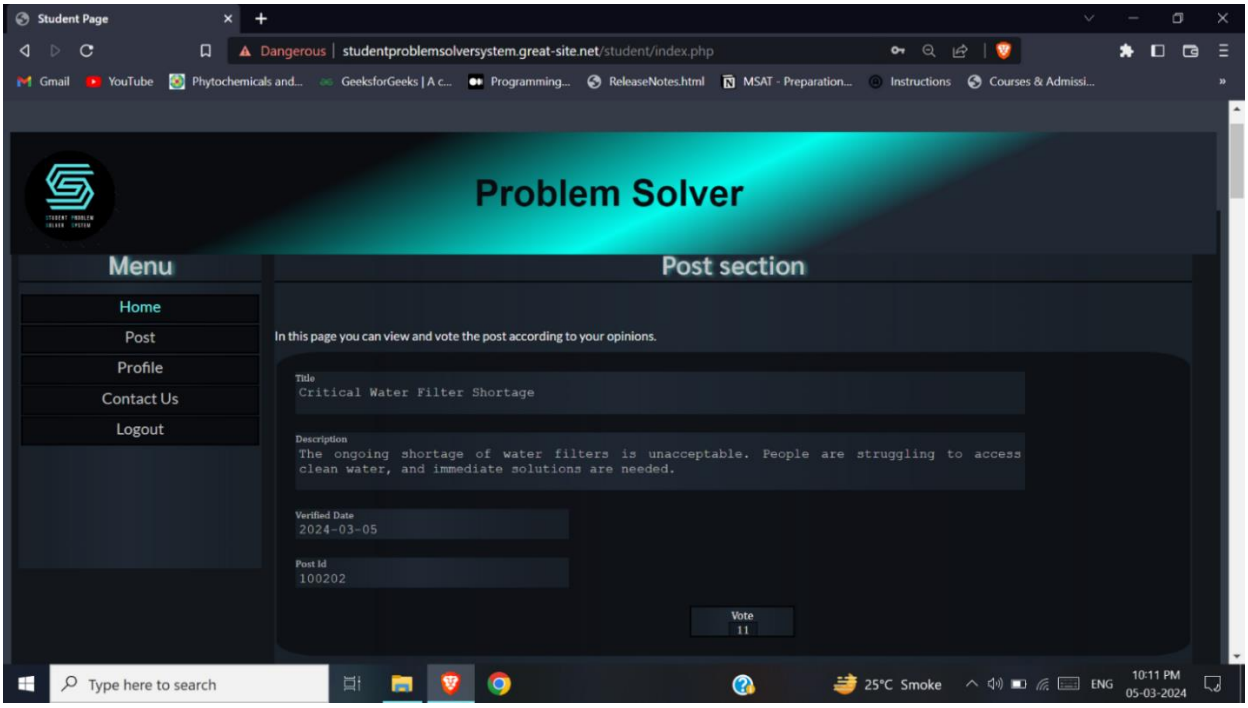


Figure 6.2.4: Student Home Page UI

5. On clicking the post tab, a page a will appear where student can fill the title and description of the problem they are facing and wanting to post.

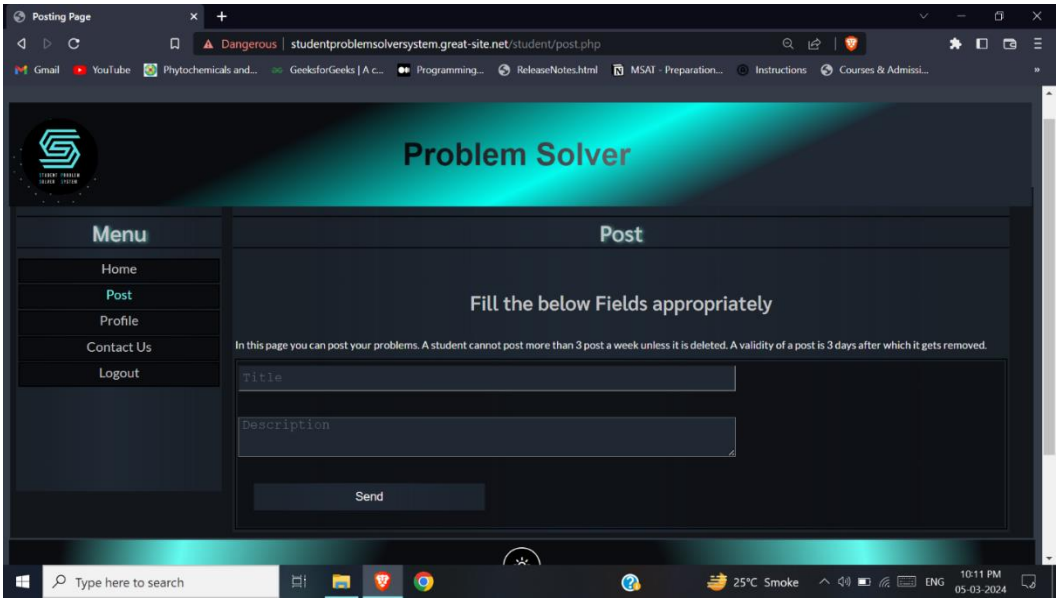


Figure 6.2.5: Student Send Post Page UI

6. On clicking the profile tab, a page a will appear where student can view and edit their profile. On clicking edit a modal will appear in which u can edit your password. To save the edited password click edit.

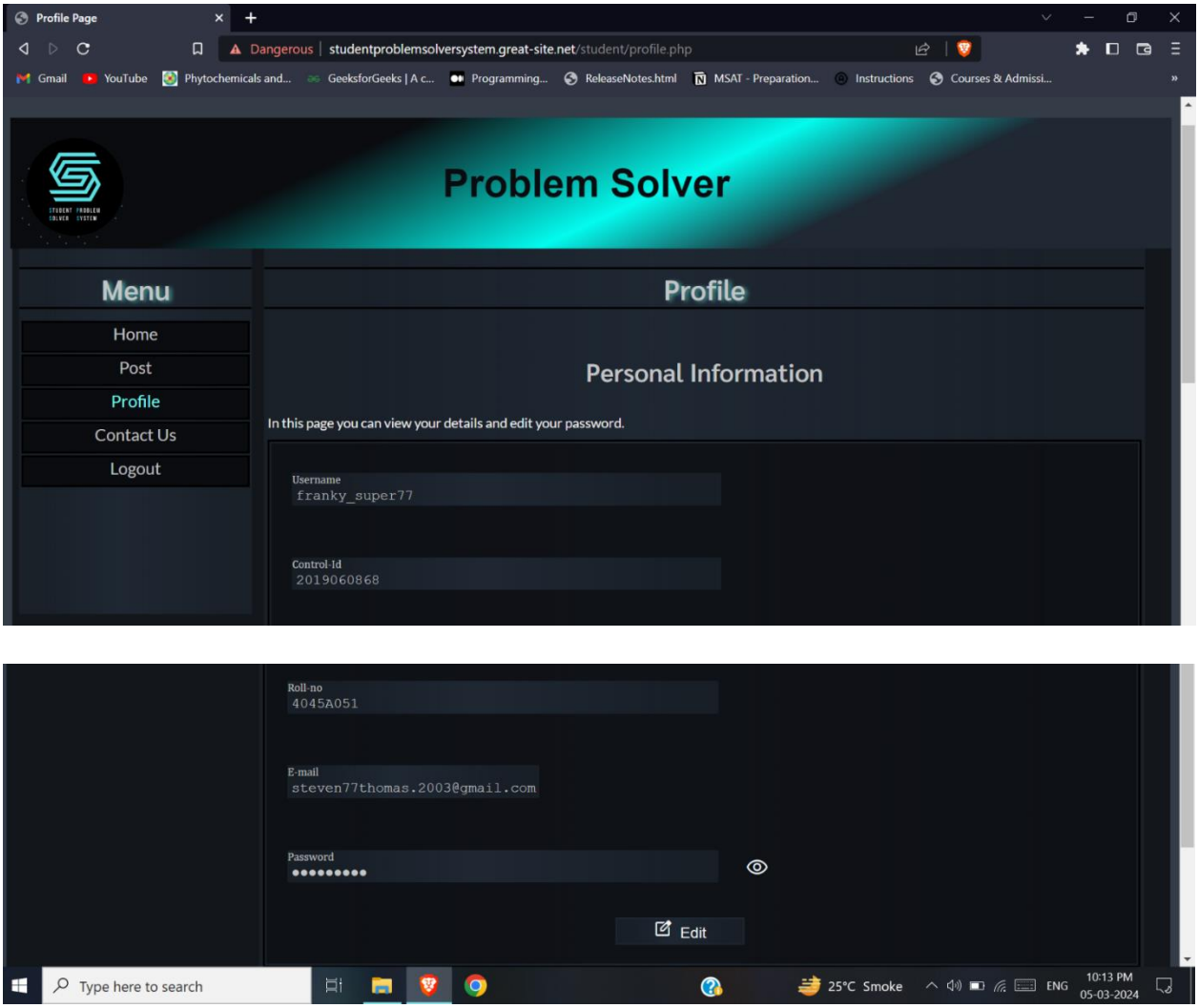


Figure 6.2.6: Student Profile Page UI

7. On clicking the Contact Us tab, a page a will appear where student can fill a form regarding any issues faced while using the website. The contact us of handler function's in the same way.

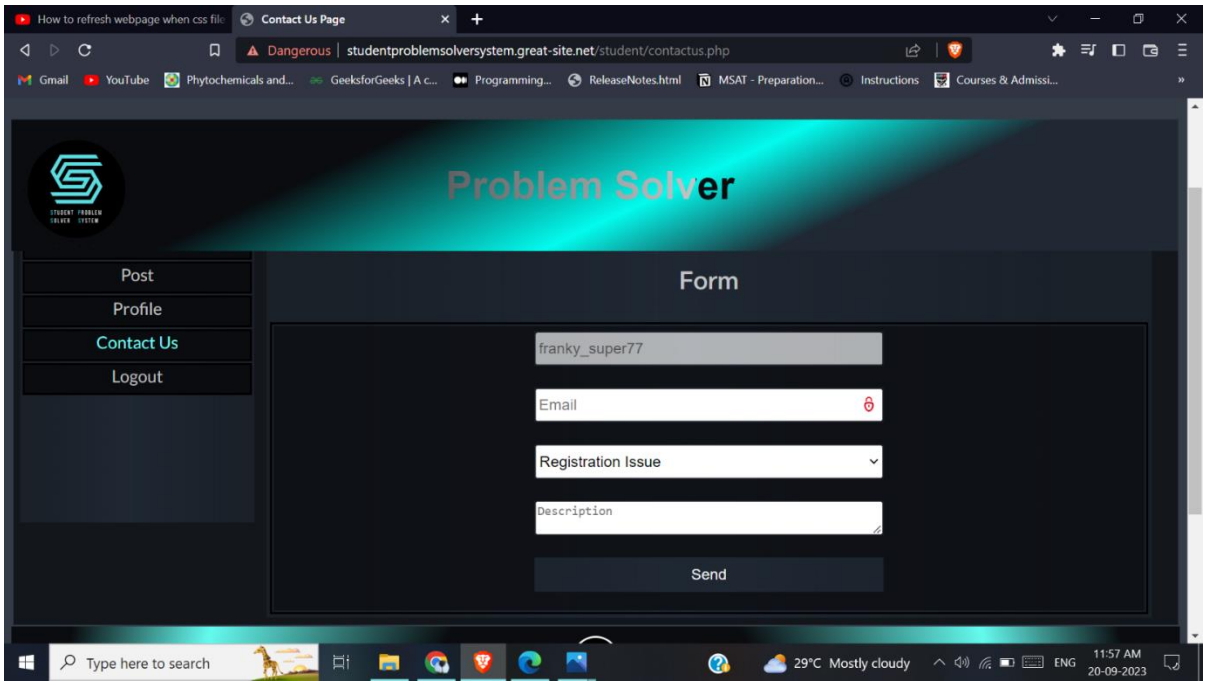


Figure 6.2.7: Student Contact Us Page UI

Handler:

8. If the handler has logged in, this would be the first home page. Here, the handler can view the most voted posts at the top.

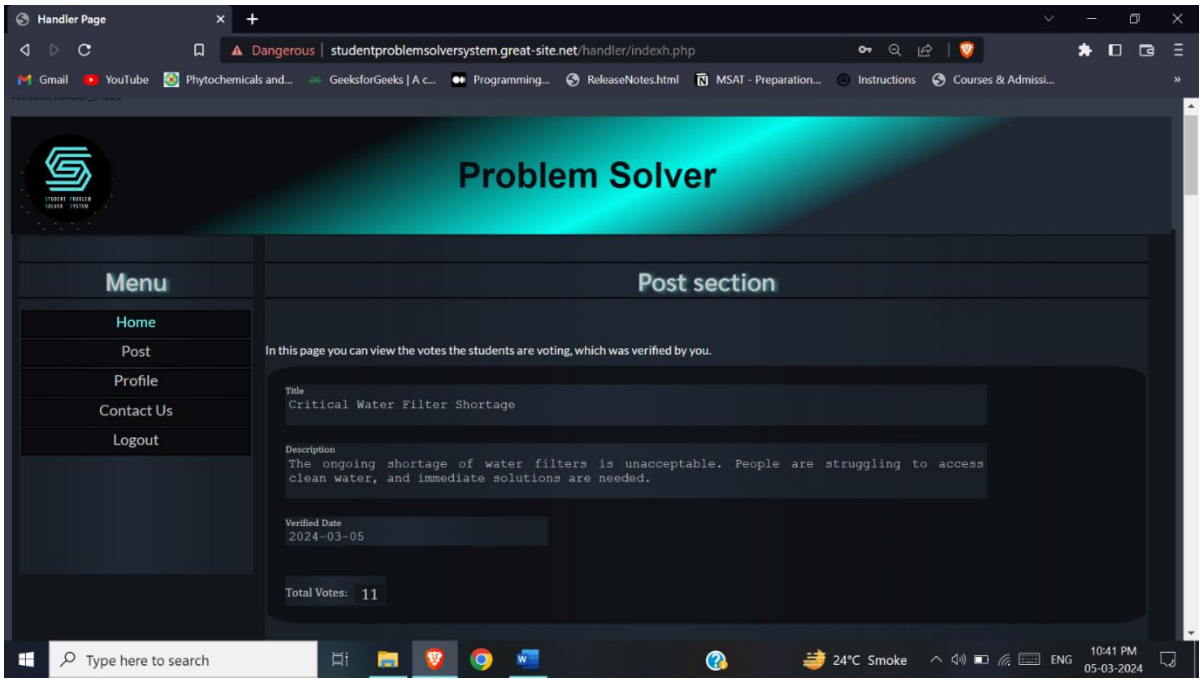


Figure 6.2.8: Handler Home Page UI

9. On clicking the post tab, a page will appear where the handler can view the posts which need to be either verified or removed. Here a search bar is present where u can search post by entering keyword and clicking the search button.

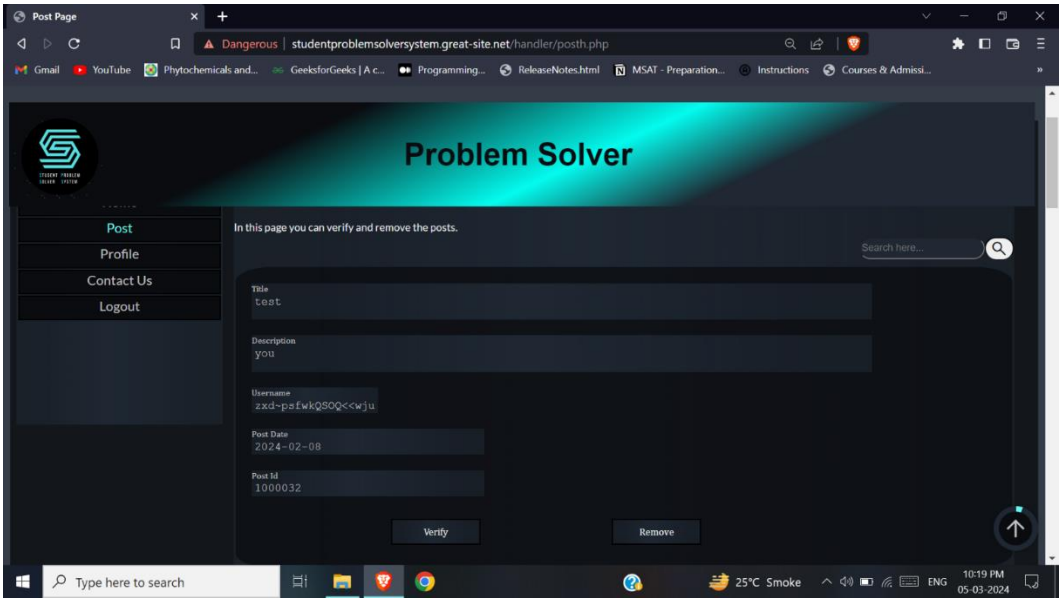


Figure 6.2.9: Handler Post Page UI

10. On clicking the profile tab, a page a will appear where handler can view and edit their profile. On clicking edit a modal will appear in which u can edit your password. To save the edited password click edit. The profile page of admin function's the same way.

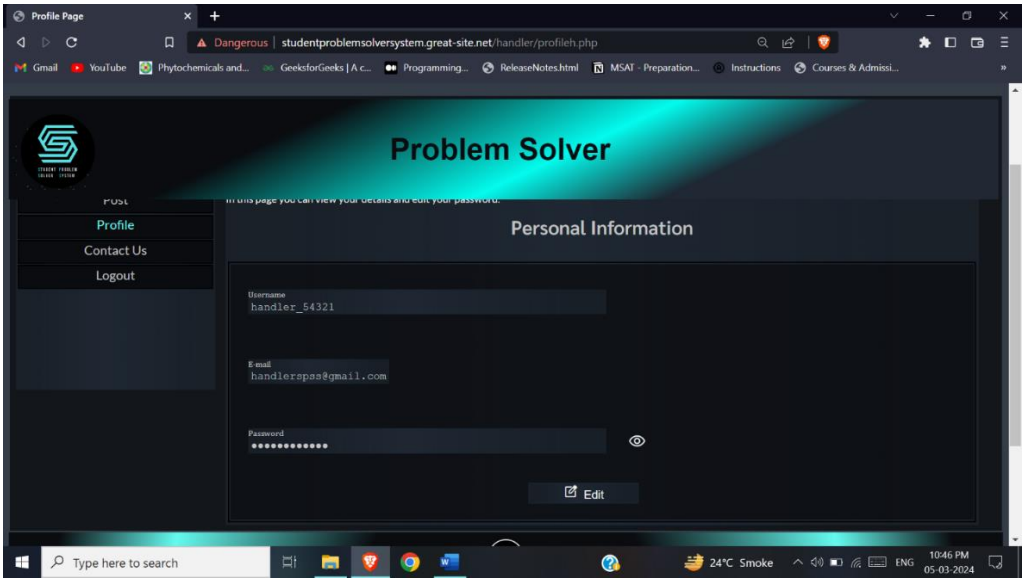


Figure 6.2.10: Handler Profile Page UI

Admin:

11. If the admin has logged in, this would be the first home page. Here, the admin can view the most voted posts at the top and remove the posted post.

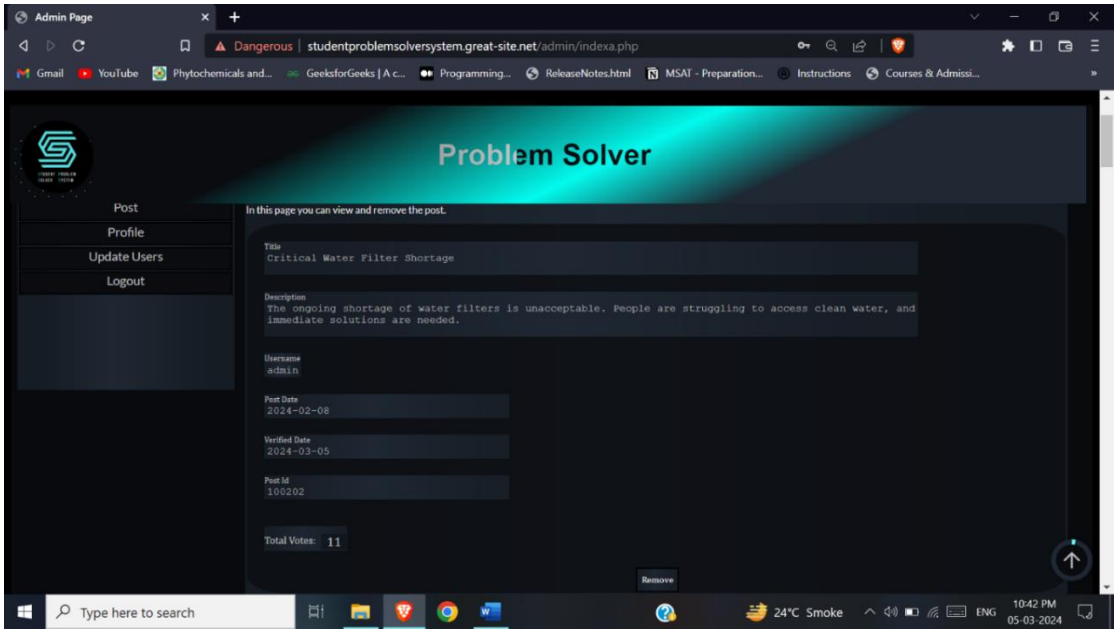


Figure 6.2.11: Admin Home Page UI

12. On clicking the post tab, a page will appear where the admin can view the posts which need to be verified and could be removed.

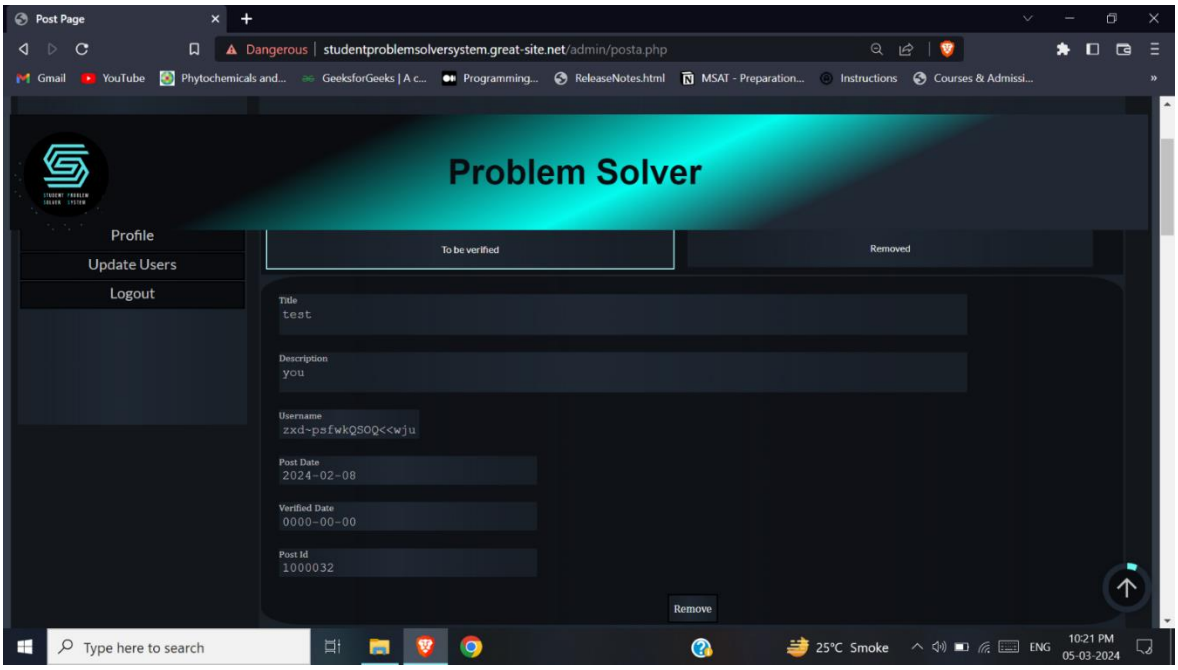


Figure 6.2.12.1: Admin Post Page 1 UI

13. On clicking the removed tab, admin can view all removed post which could either be deleted or verified.

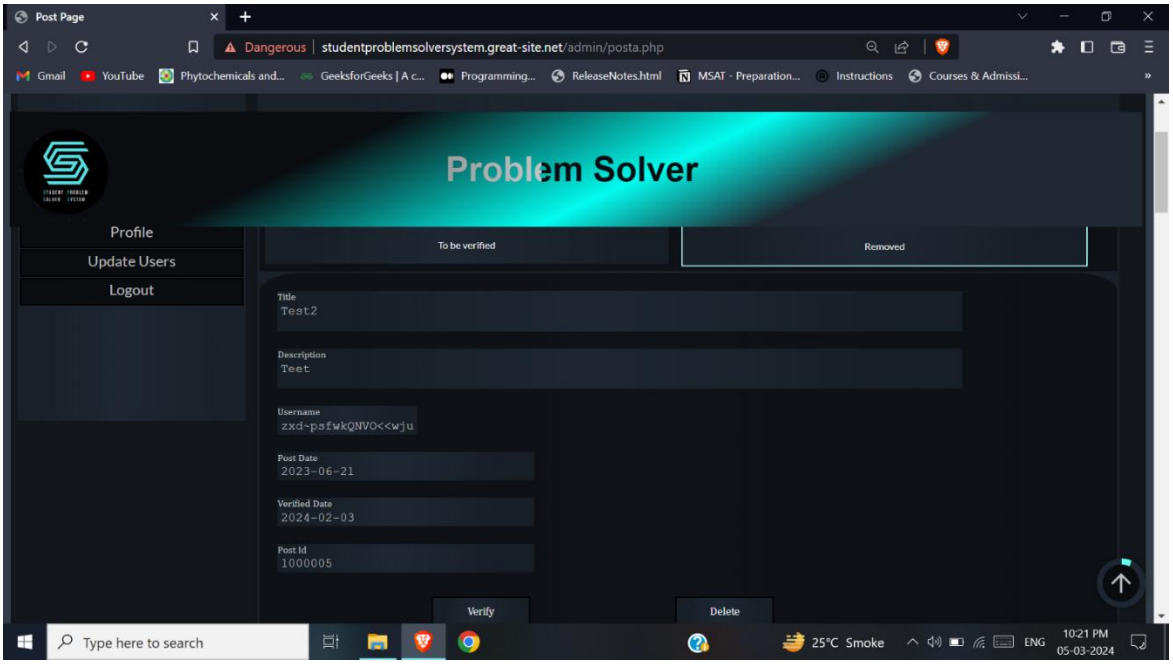


Figure 6.2.12.2: Admin Post Page 2 UI

Manage users Page:

14. On clicking the Manage users tab, admin can view all student users. On clicking the add symbol, a modal will appear where a user can be added. On clicking the edit button, a modal will appear where a user can be edited. On clicking the delete button, the user will be deleted. The modals appearing is applicable to the other user add and edit.

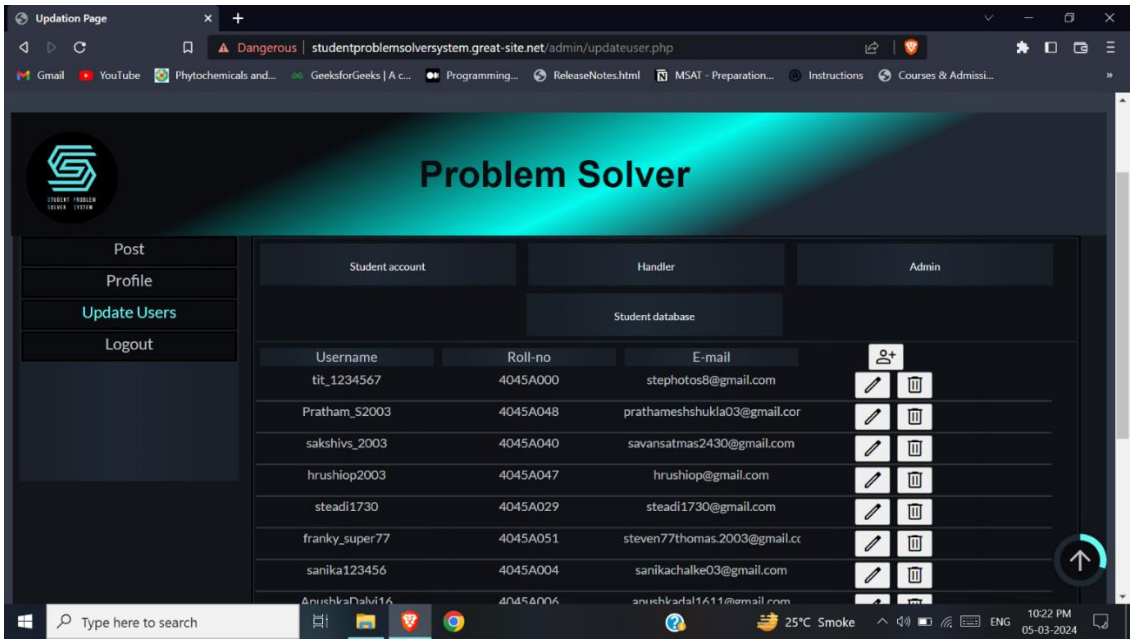


Figure 6.2.13.1: Admin Update Users Page 1 UI

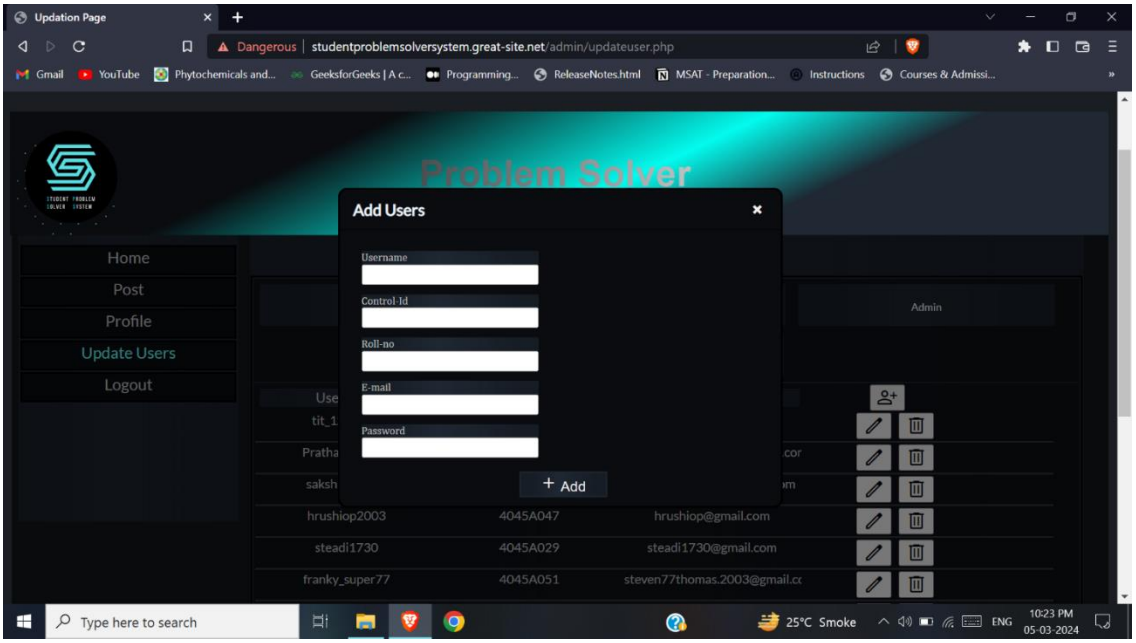


Figure 6.2.13.2: Add user model for student account UI

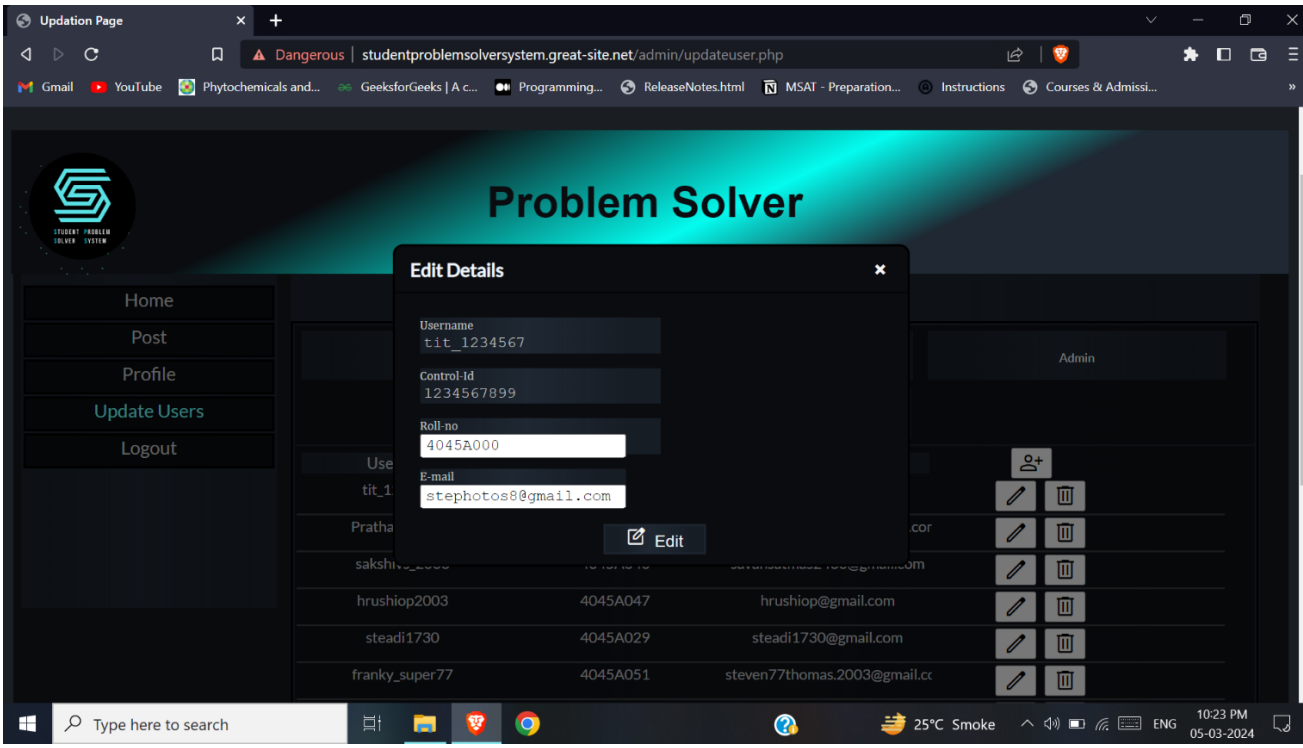


Figure 6.2.13.3: Edit user model for student database UI

15. On clicking the Handler tab, admin can view all handler users. On clicking the add symbol, a modal will appear where a user can be added. On clicking the edit button, a modal will appear where a user can be edited. On clicking the delete button, the user will be deleted.

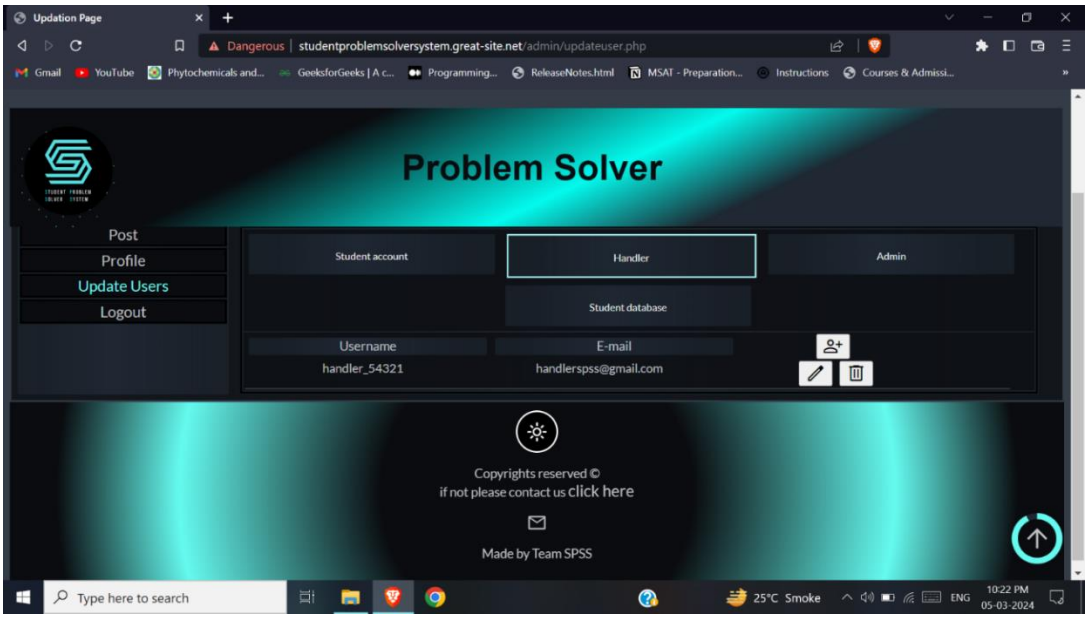


Figure 6.2.13.4: Admin Update Users Page 2 UI

16. On clicking the admin tab, admin can view all admin users. On clicking the add symbol, a modal will appear where a user can be added. On clicking the edit button, a modal will appear where a user can be edited. On clicking the delete button, the user will be deleted.

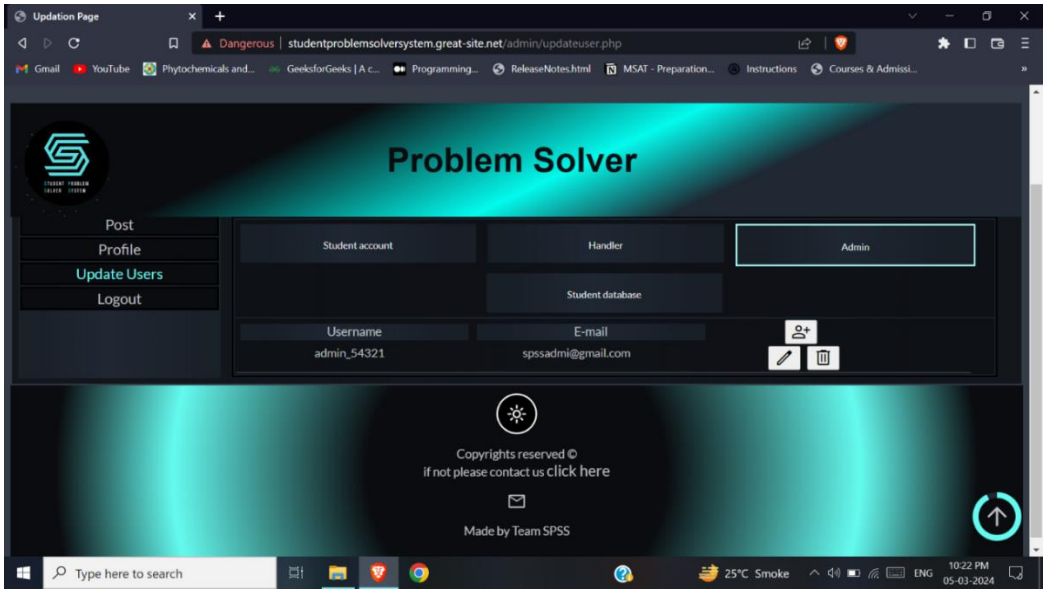


Figure 6.2.13.5: Admin Update Users Page 3 UI

16. On clicking the admin tab, admin can view all student information present in the college. On clicking the add symbol, a modal will appear where a user can be added. On clicking the edit button, a modal will appear where a user can be edited. On clicking the delete button, the user will be deleted.

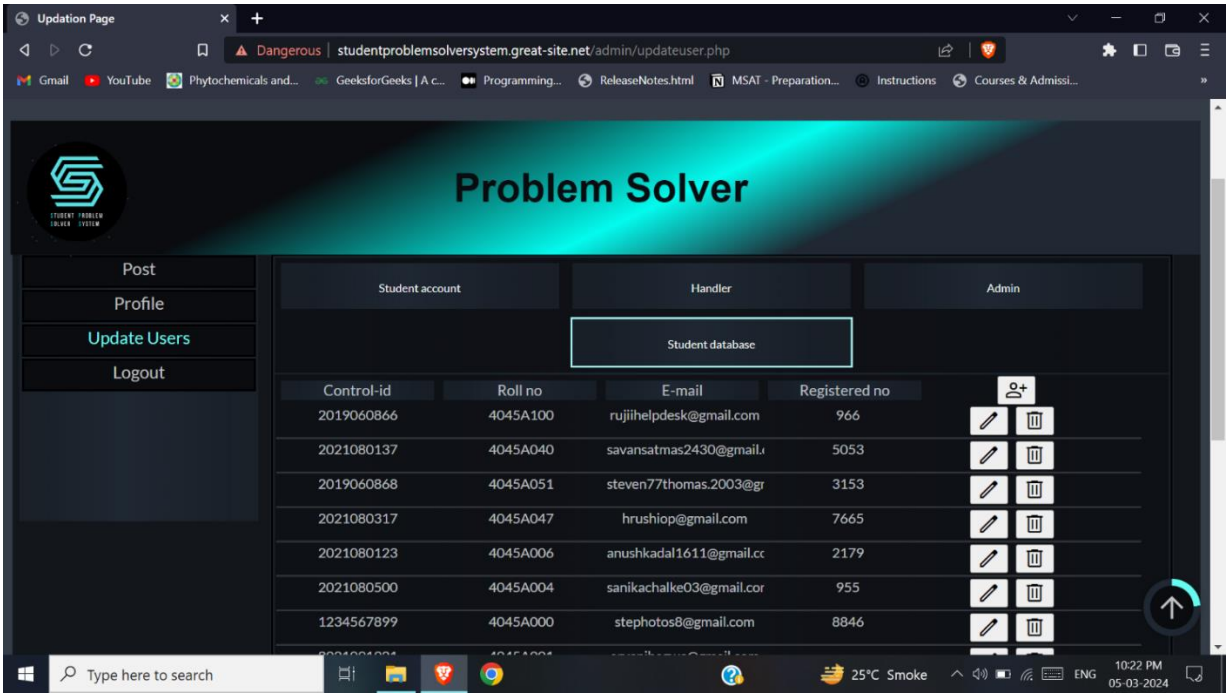


Figure 6.2.13.6: Admin Update Users Page 4 UI

17. After all the work has been completed the user can logout by clicking logout button.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

The student problem solver system will be helpful for the students of the college, as it will help them to communicate their problems easily and anonymously through this software to the college management.

In this website the students can post their problems and vote on post which has been posted. They can view and edit their profile details. In case of any other issues faced while using the software, the student can contact the issue through the contact us page to the admin.

Handler verifies the posts before being posted to the other students or is removed if found irrelevant. Handler can view the most voted most and convey to the college management. Profile and contact us page is available here too.

Admin can add, edit, and delete users. Admin can remove, verify, and delete posts under specific conditions. Profile page is available for editing details.

7.2 Limitation

- Students cannot update details apart from password without contacting the admin.
- Students cannot post 3 post per week.
- Students cannot undo a vote.
- Student cannot vote multiple times to a post
- Students cannot have multiple account.
- Handler cannot remove a verified post.
- Admin cannot verify post which has not been removed or not verified.
- Admin cannot edit the controlid and username of a student.

7.3 Future Scope

- To every post personalised notification could be sent.
- To prevent repeated posts from being posted Natural language Processing and Machine learning could be applied.
- All solved problem of year could be displayed after a year of software use.

References

Books Referred:

[1] Software Engineering"Ian Somerville", Pearson publisher 8 edition

[2] Database System Conceptes,"Henry F Korth, Abraham Silberschatz"

[3] James Rumbaugh Object-Oriented Modelling and Design with UML Second Edition

Website References:

The Links referred from the date 03/04/2023 to 25/04/2023:

[1] Student Friendly College Management System - 1000 Projects

[2] https://www.tutorialsduniya.com/software-engineering-projects-pdf/

[3] https://nevonprojects.com/online-diagnostic-lab-reporting-system/

[4] https://www.tutorialsduniya.com/software-engineering-projects-pdf/

[5] https://www.youtube.com/watch?v=1Rs2ND1ryYc&t=6210s

[6] https://www.php.net/docs.php

Referred for UML diagrams:

[1] Wondershare edrawmax software

Referred for Gant Chart:

[1] Microsoft Excel software