SOFTWARE ENGINEERING

SOFTWARE DESIGN SPECIFICATION

DISCUSSION PORTAL
WEB APPLICATION

BY:-

NEHAL SINGH (IIT2018119)

TABLE OF CONTENT

The Software Design Specification

1. Introduction

DISCUSSION PORTAL(DP) is a discussion forum that gives information about various programming languages, general knowledge related questions,and course related questions of students by faculty.

This Document contains class models, sequence diagrams, collaboration models, object behaviour models, and other supporting requirement information.

1.1 Purpose of this document

To create a fully functional and user interactive online web portal for student query discussion which will help in minimizing the gap between the student and their

instructor. Whenever a question is asked by the end-user to get information, it is received by the instructor. Any user can post doubts and can reply for the other user

doubts. There is a centralized database in which all the information is managed. The

administrator has the rights to update the database,delete the information.

. This is useful for a small school or a department or for that matter any group who is

interested in organizing it effectively.

1.2 Scope of the development project

This software system will be an online portal for IIT ALLAHABAD, or any university

wishing to create a discussion forum for its students. More specifically to design and develop

a simple and intuitive system which shall cater the needs of the students to clear their doubts when not present in class or doubts arising during studying .

The system shall provide features to the user of an educational institute to ask questions,

search for a previously asked questions, reply to the

questions asked by different users, upvote/downvote the questions and answers.

Further we can add more functionalities to make it better and update according to future needs.

In Scope:

a. Ask questions by students,Replies by faculty and already existing questions can be searched.

b.Admin has all access to delete the inappropriate queries posted by users, which are reported by users.

Out of Scope:

a.Any online conduction of test,assignments etc. b.Personal chats among users

1.3 Definitions, Acronyms and Abbreviation

Admin – Administrator.

HTML - HyperText Markup Language

PHP - Hypertext Preprocessor

CSS - Cascading Style Sheets

MySQL - Database Management System

1.4 References

1. IEEE SDS Format.

1.5 Technologies to be used

1. Application Architecture - HTML
2. Database Application - MySQL
3. Web Deployment Server - PHP
4. Development tool - Rational System Developer
5. Designing Language - CSS

1.6 Overview of Document

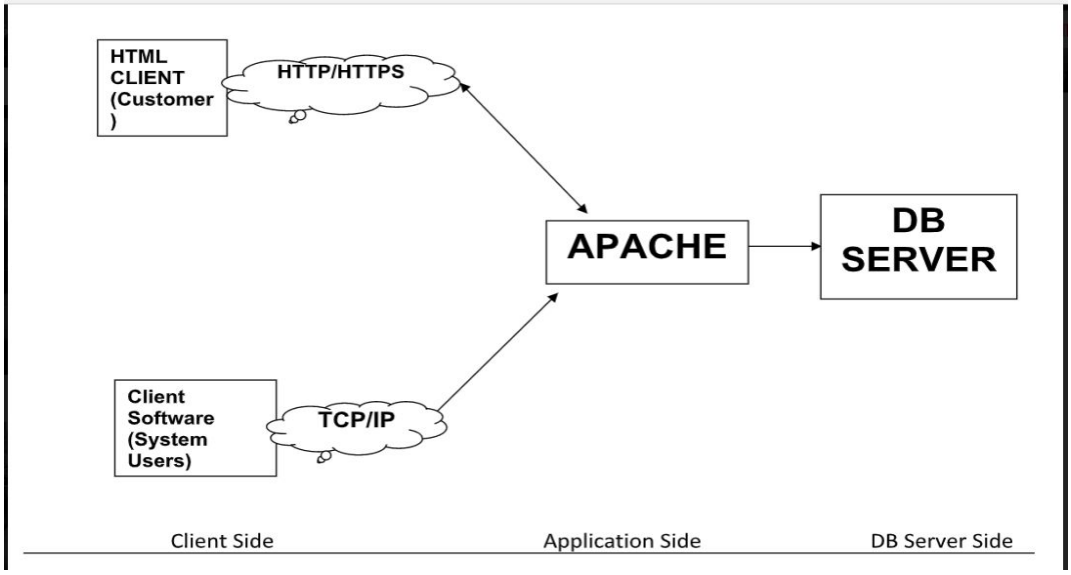
Student Query Discussion Portal is the technology which increases the interaction of
student and instructor which greatly extends classroom learning. It helps in more
democrartic exchange due to participation of more students in the discussion who are
unable to voice their opinion in the class.
This SDS is divided into seven sections with various sub-sections. The sections of the
Software Design Document are:

1. Introduction: describes about the document, purpose, scope of development
   project definitions and abbreviations used in the document.
2. Conceptual Architecture/Architecture Diagram: describes the overview of components, modules, structure and relationships and user interface issues.
3. Logical Architecture: describes Logical Architecture Description and Components.
4. Execution Architecture: defines the runtime environment, processes, deployment view.
5. Design Decisions and Trade-offs: describes the decisions taken along with the reason as to why they were chosen over other alternatives.
6. Code Snippets: contains some code snippets as name indicates.
7. Appendices: describes subsidiary matter if any.

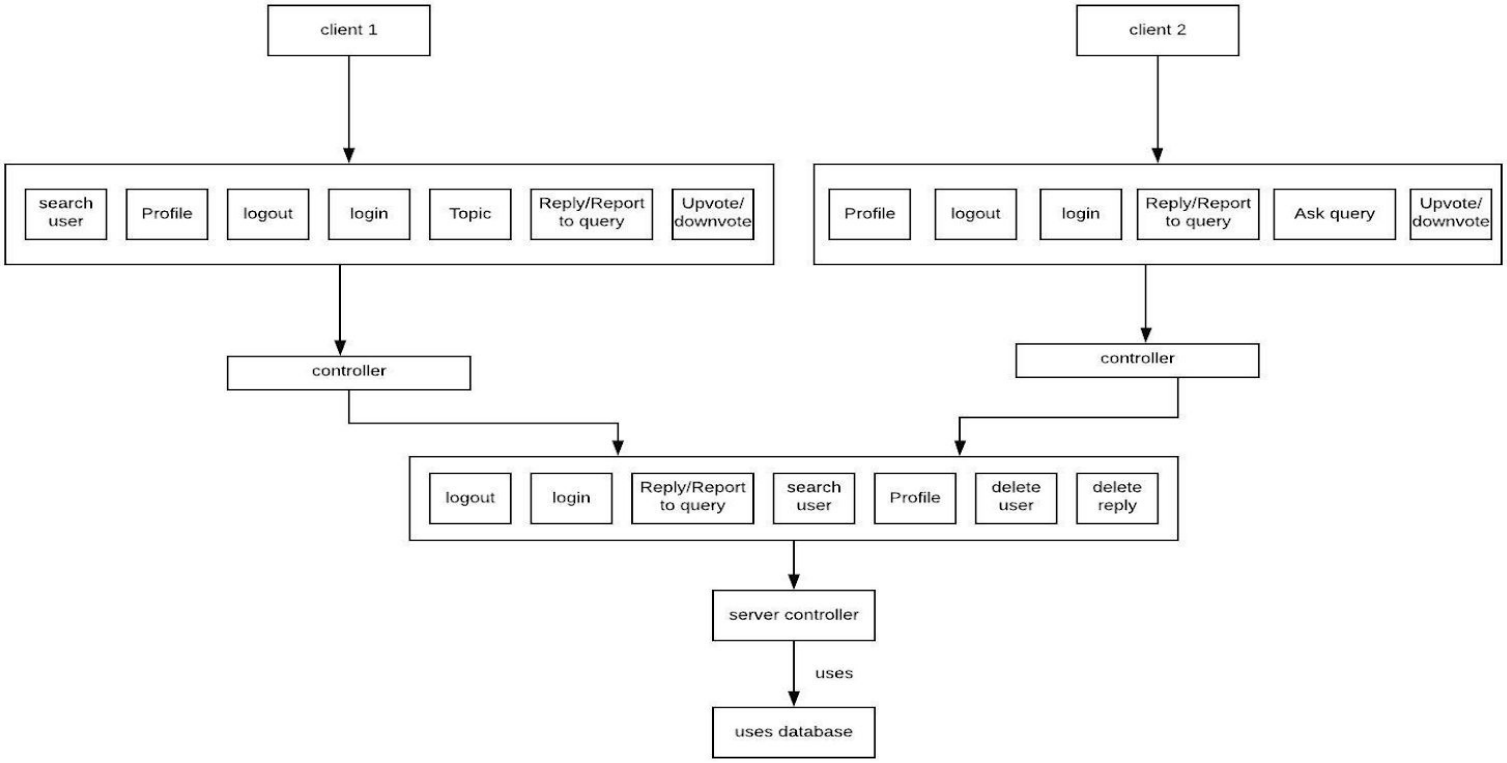
2. Conceptual Architecture/Architecture Diagram

It uses three tier architecture dig where client side application side and DB server are three different components that improve security in the system.

Architecture 1:

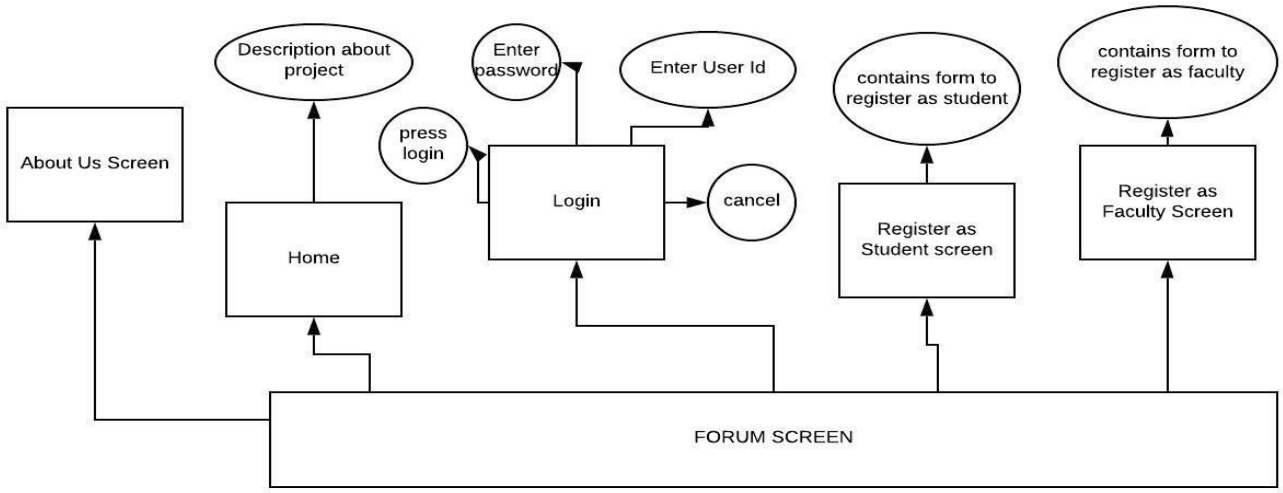


Architecture 2:

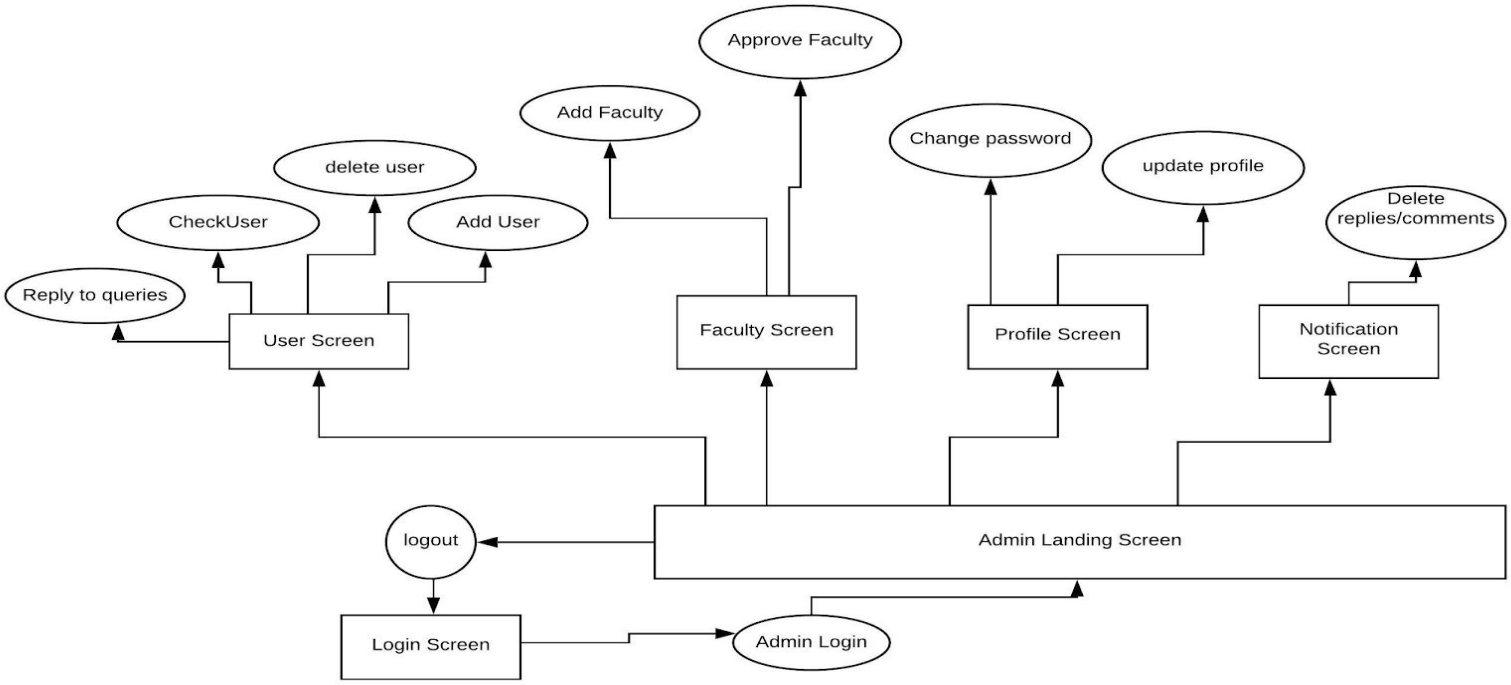


2.1 Structure and relationships

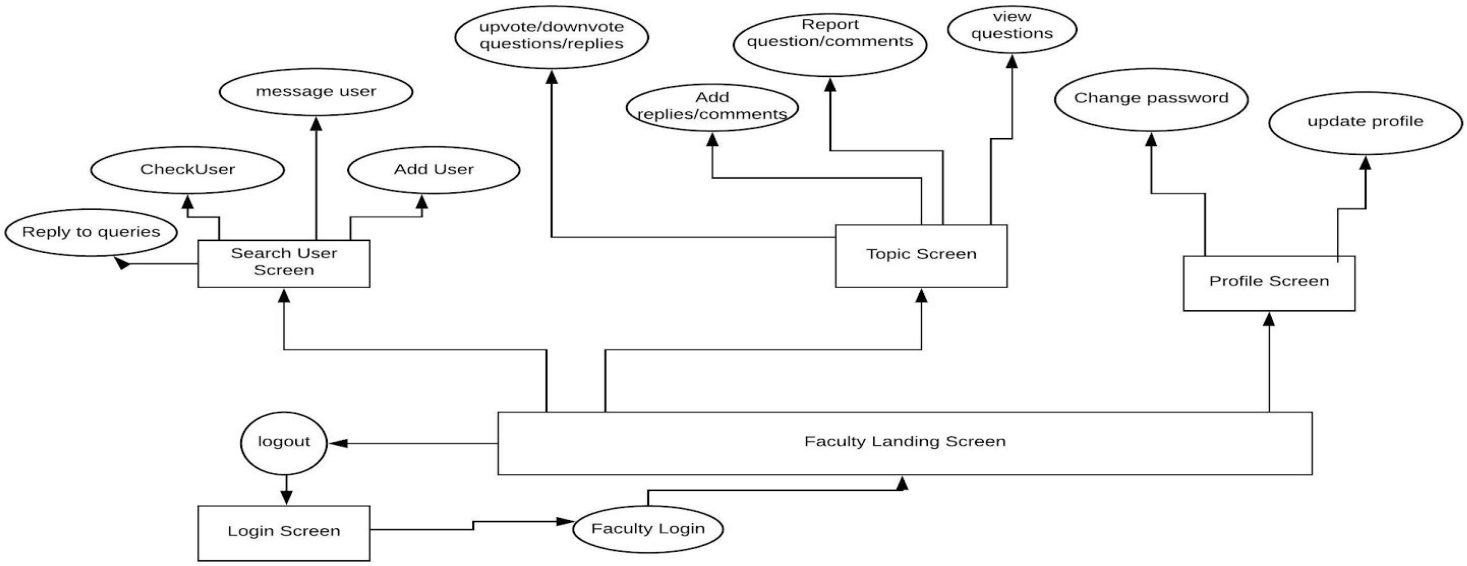
Forum Screen(Front page)



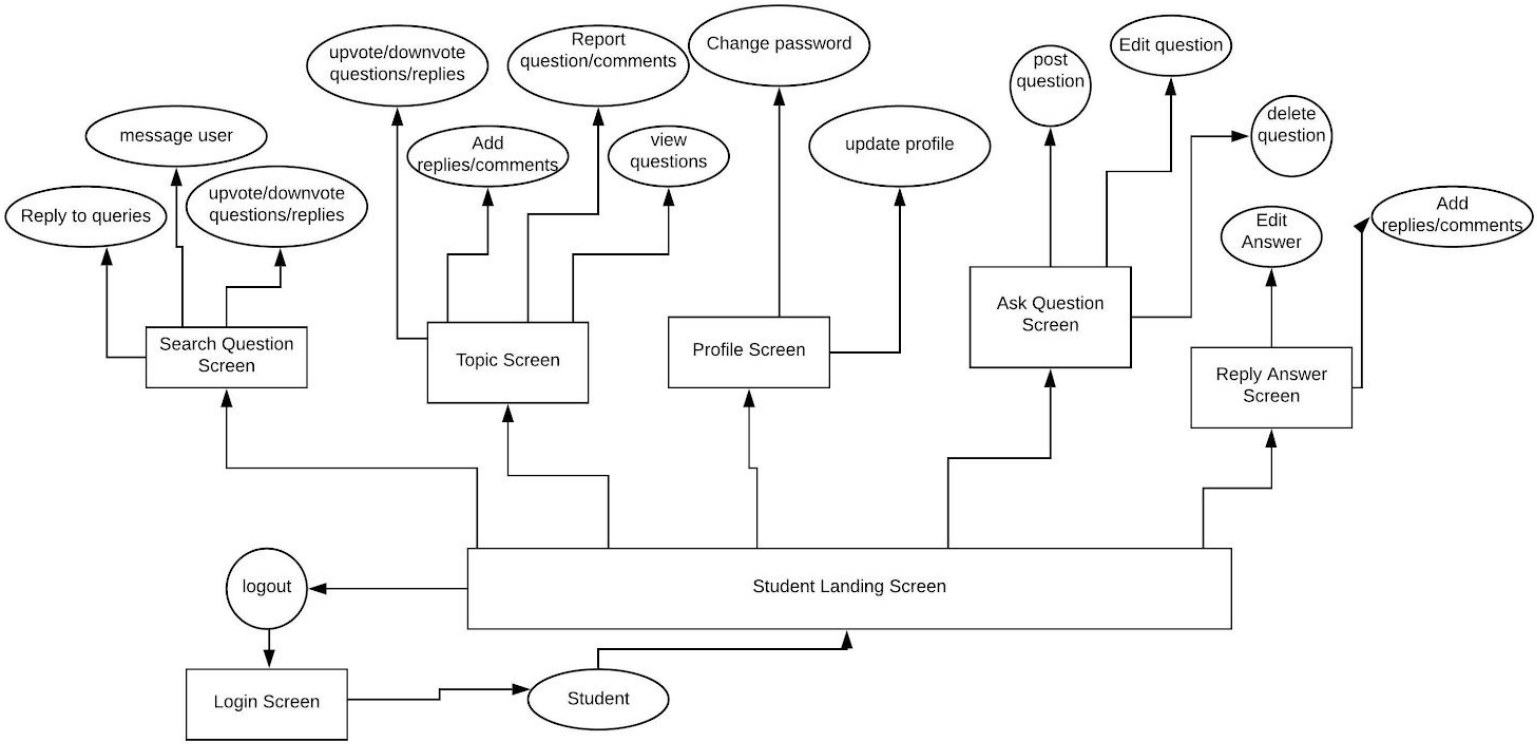
Admin's Side



Faculty's Side



Student's Side



NOTE:

The boxes represent individual screens.
The circles represent actions that do not have screens.
The arrows represent navigation between screens.

2.2 User interface issues

This section will address User Interface issues as they apply to the following hypothetical users of the Query Discussion Portal.

- User A is a 20-year-old female, a student of IIIT Allahabad, IInd year, who is fairly comfortable with technology. She is proficient with using most common computer applications.

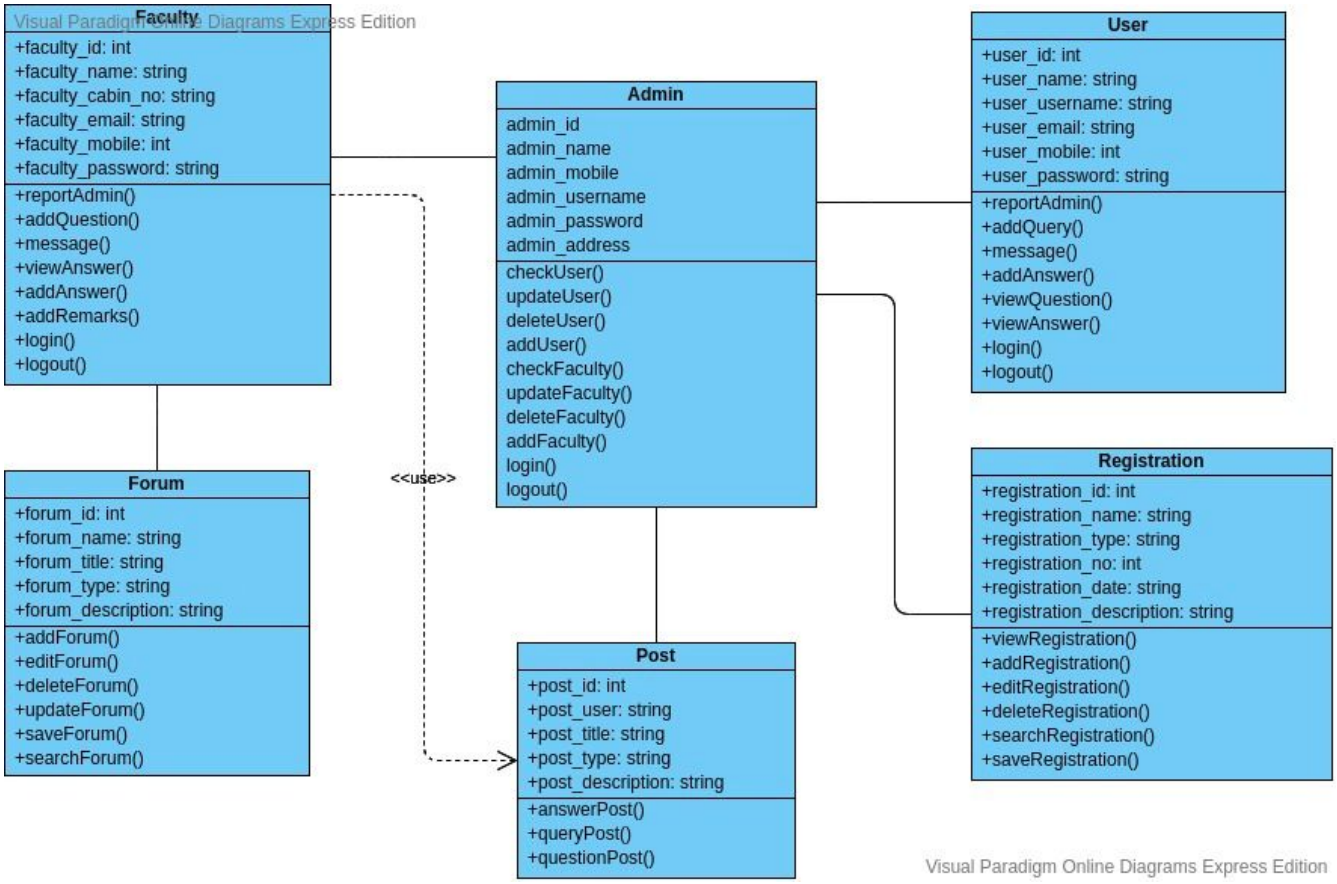
Since User A is familiar with web applications, like QDP will use common user interface conventions. For example, links between screens will use ordinary, easy to understand descriptions such as “Login”, ”Home”, ‘ASk queries” etc. To maintain consistency, any other links will also appear in the bottom half of the screen.

- User B is a 19-year-old male, a student of IIIT Allahabad, Ist year, He is although comfortable using web applications, he has never used any portal like, QDP. He may or may not acknowledge if this android application is really helpful.

  Since User B is not fond of such a technology oriented(asking questions via forums), it is imperative that he be given clear on-screen directions. Consequently, the user interface has been tried to be such that grabs the attention of the user. It has been taken care that user should not consider this application as a burden/difficult to use. On-clicks task implementation has been done .

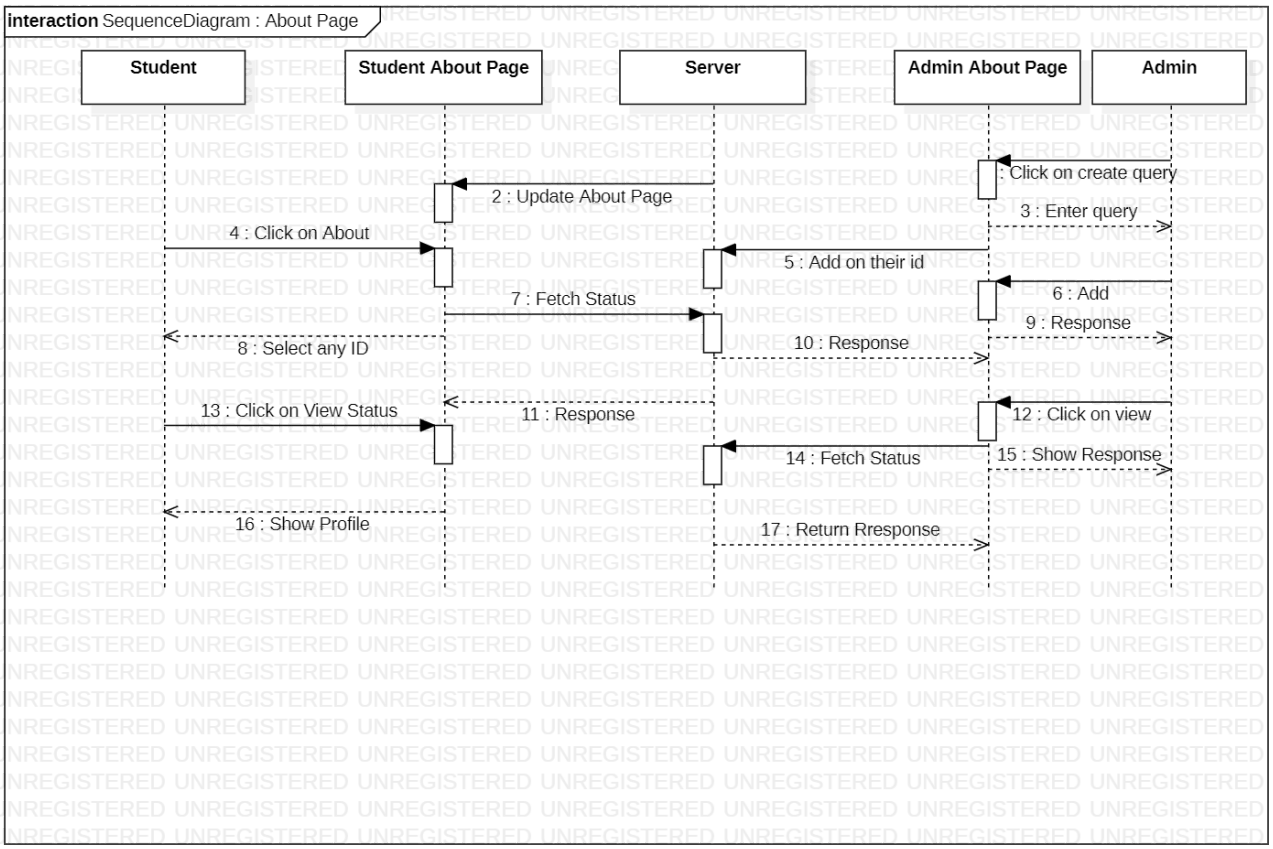
3. Logical Architecture (Class Diagram, Sequence Diagram, State Diagram)

Class Diagram.

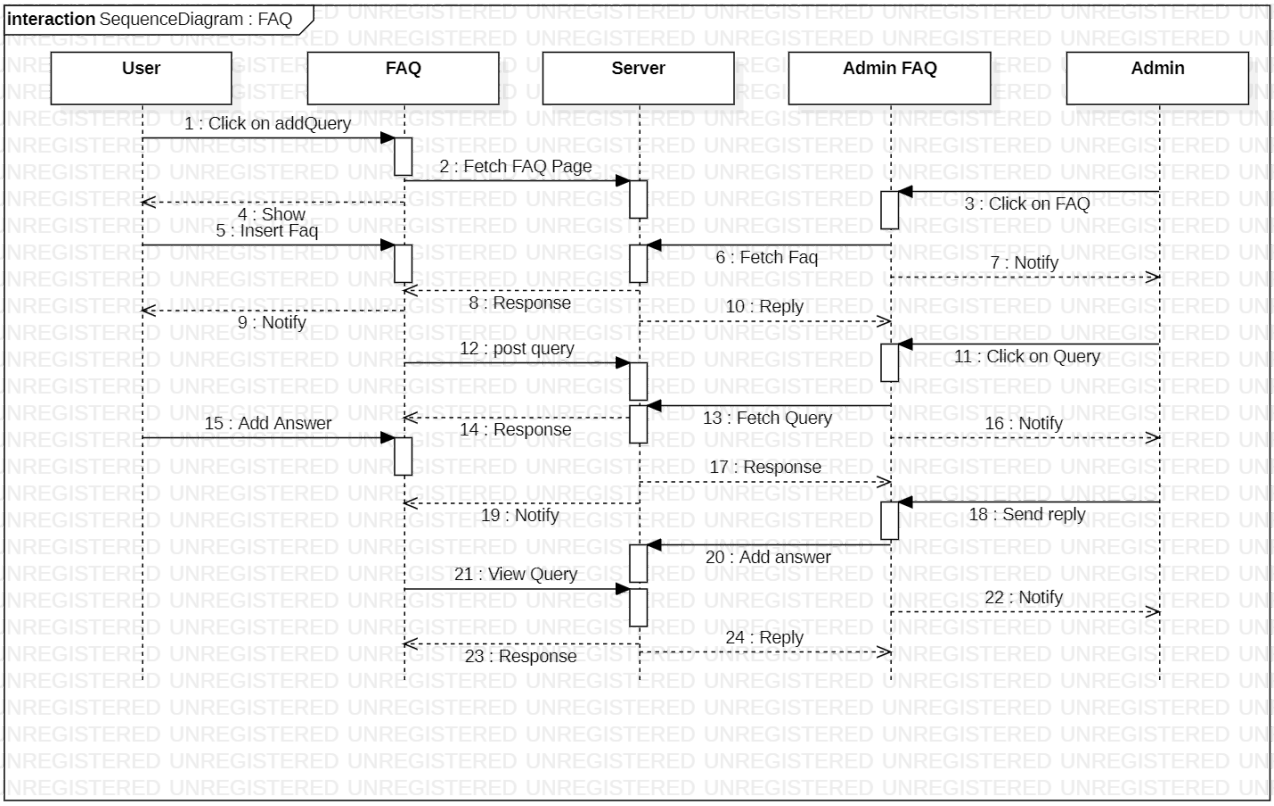


Sequence Diagrams:

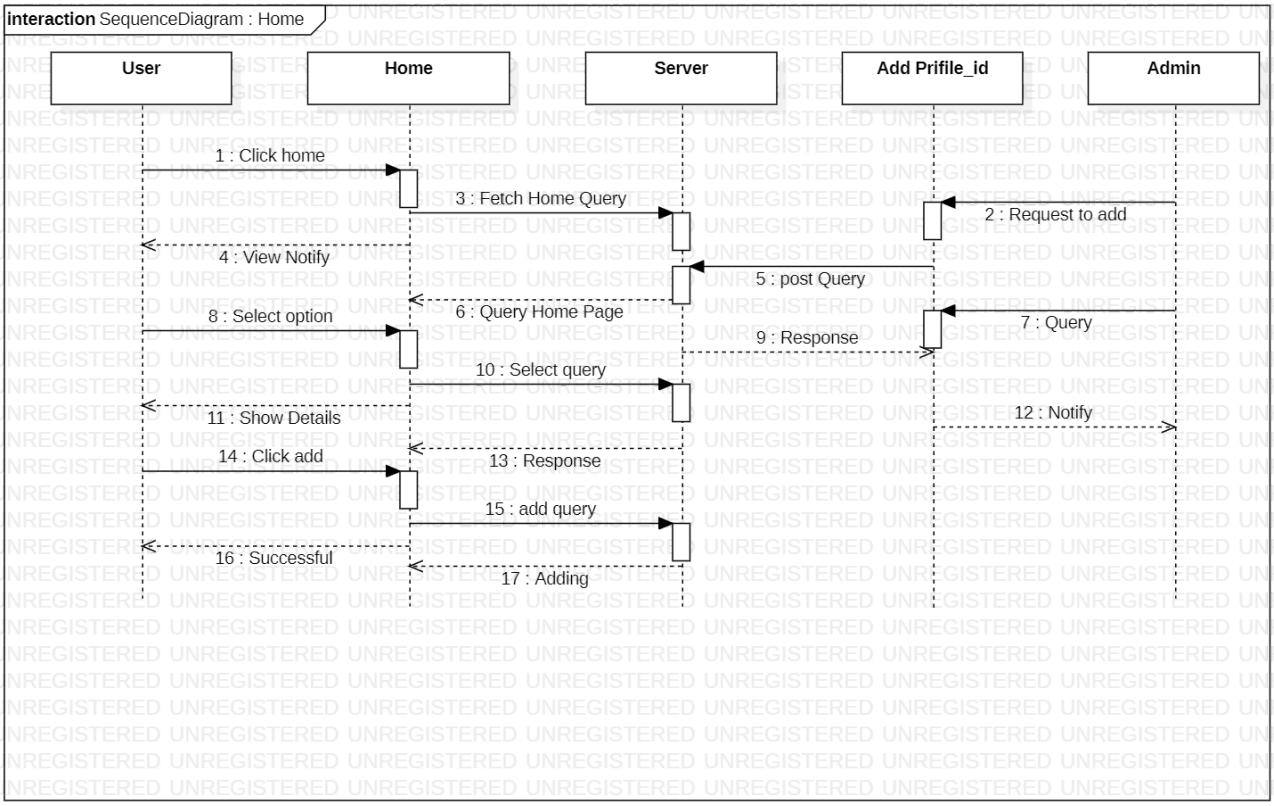
Sequence Diagram : About Page



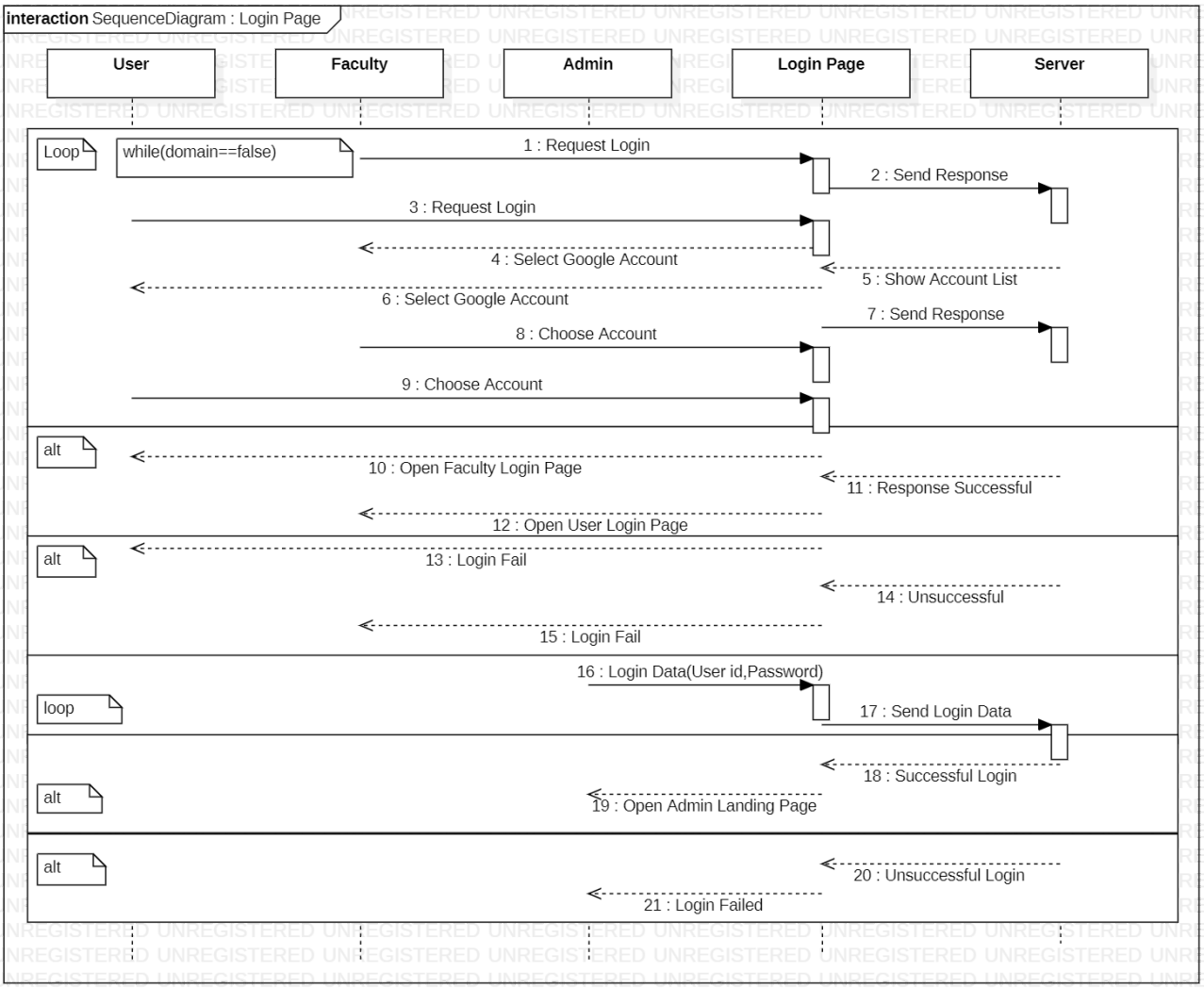
Sequence Diagram : FAQ



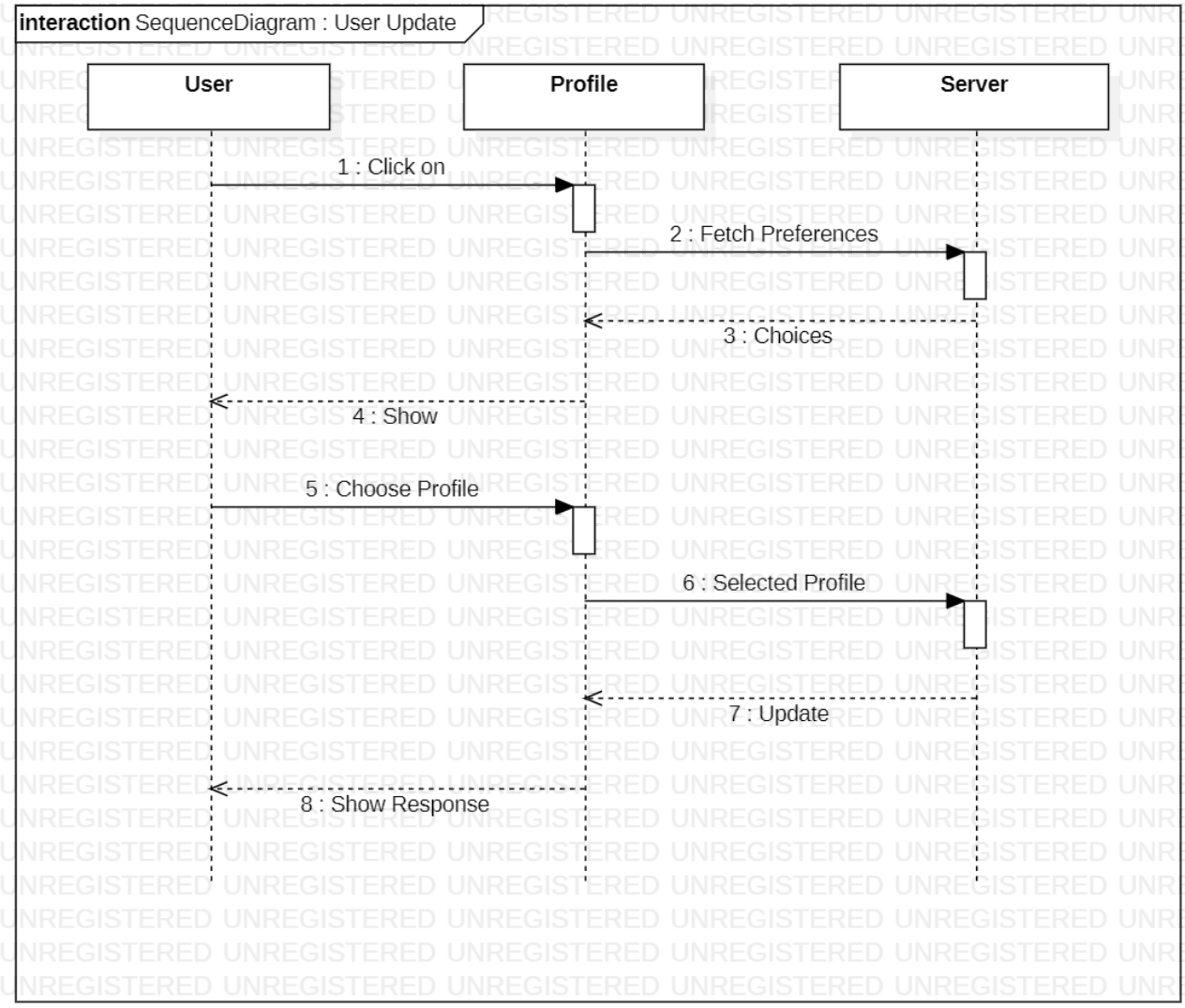
Sequence Diagram : Home



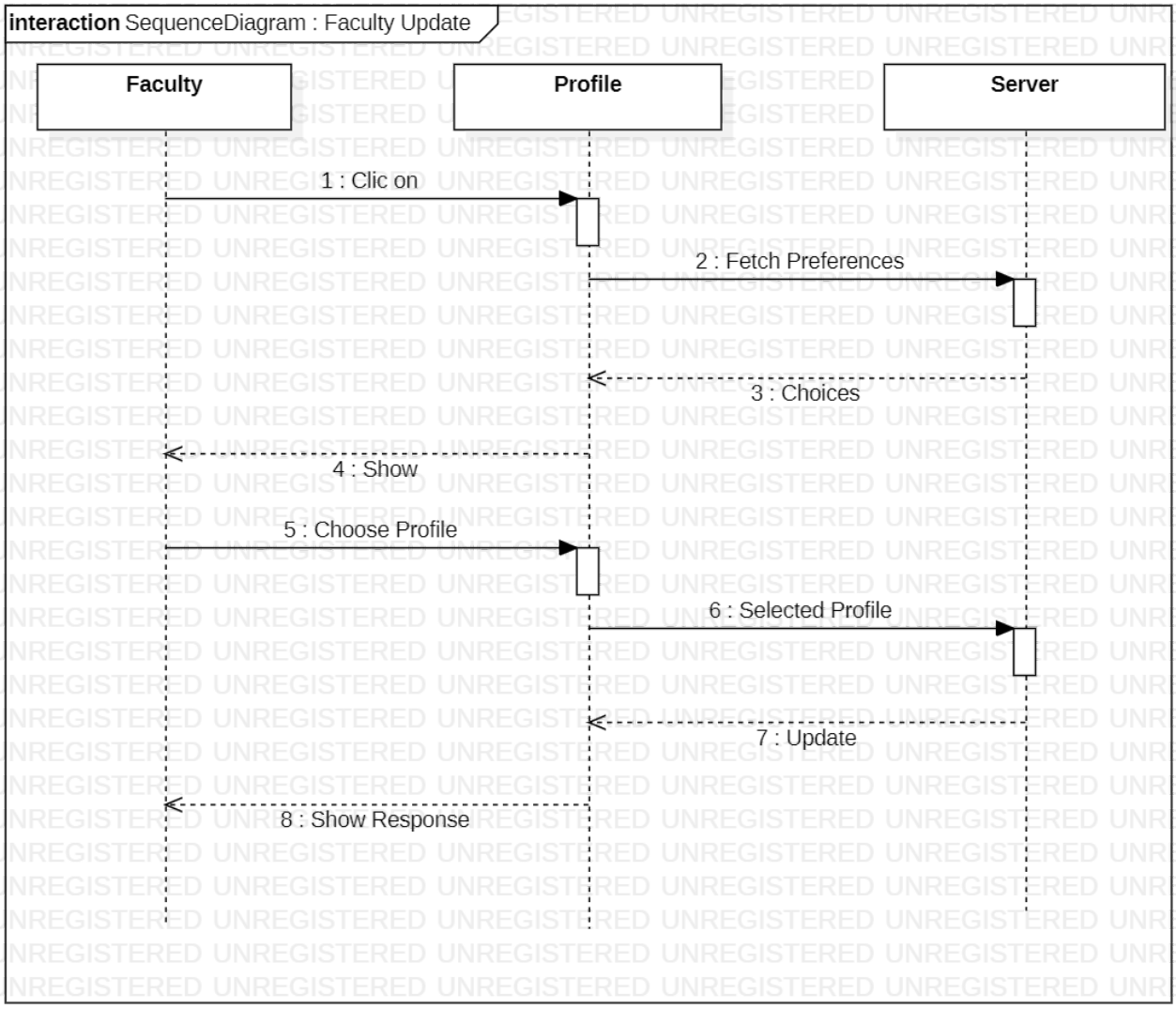
Sequence Diagram : Login Page



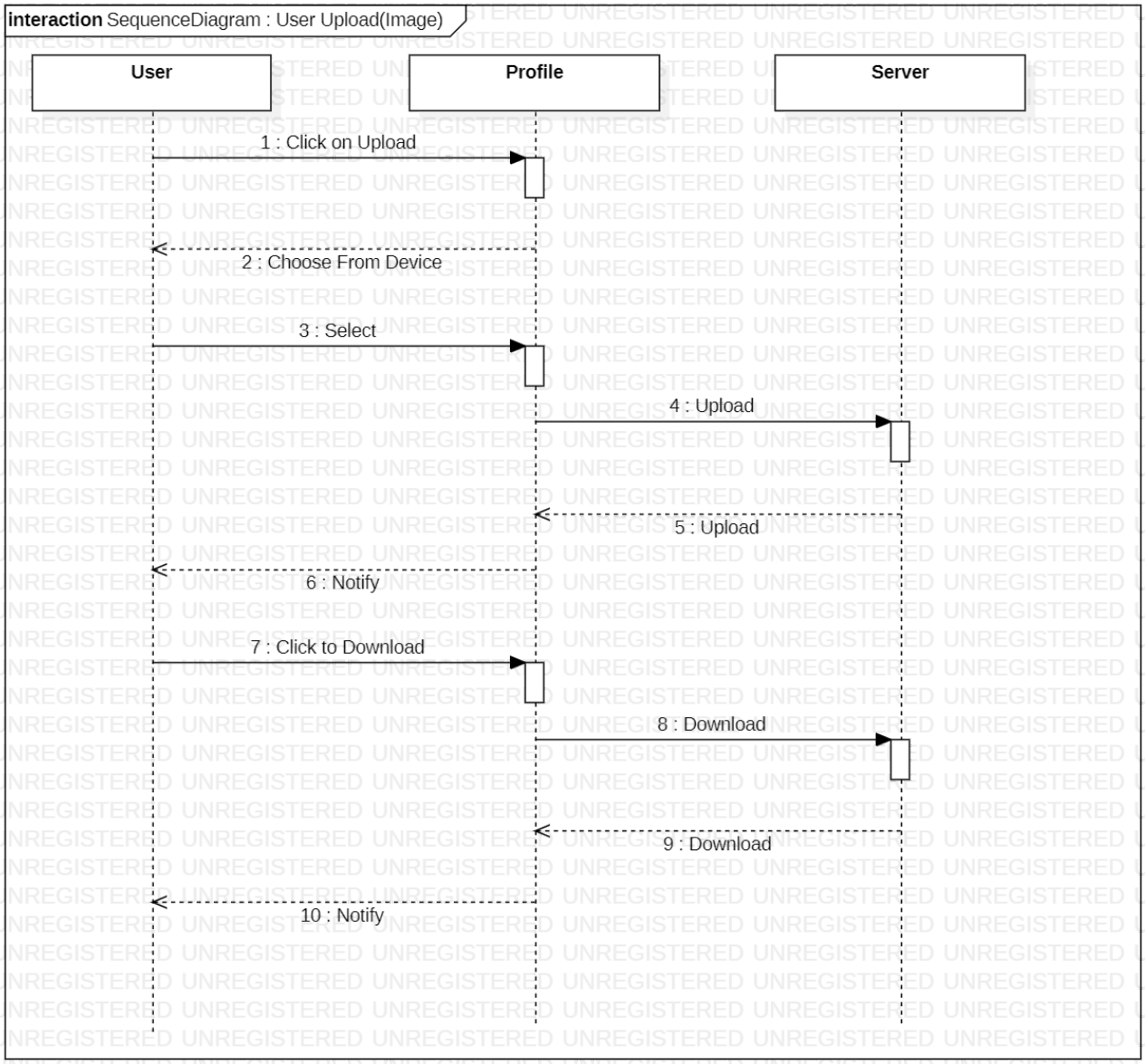
Sequence Diagram : User Update



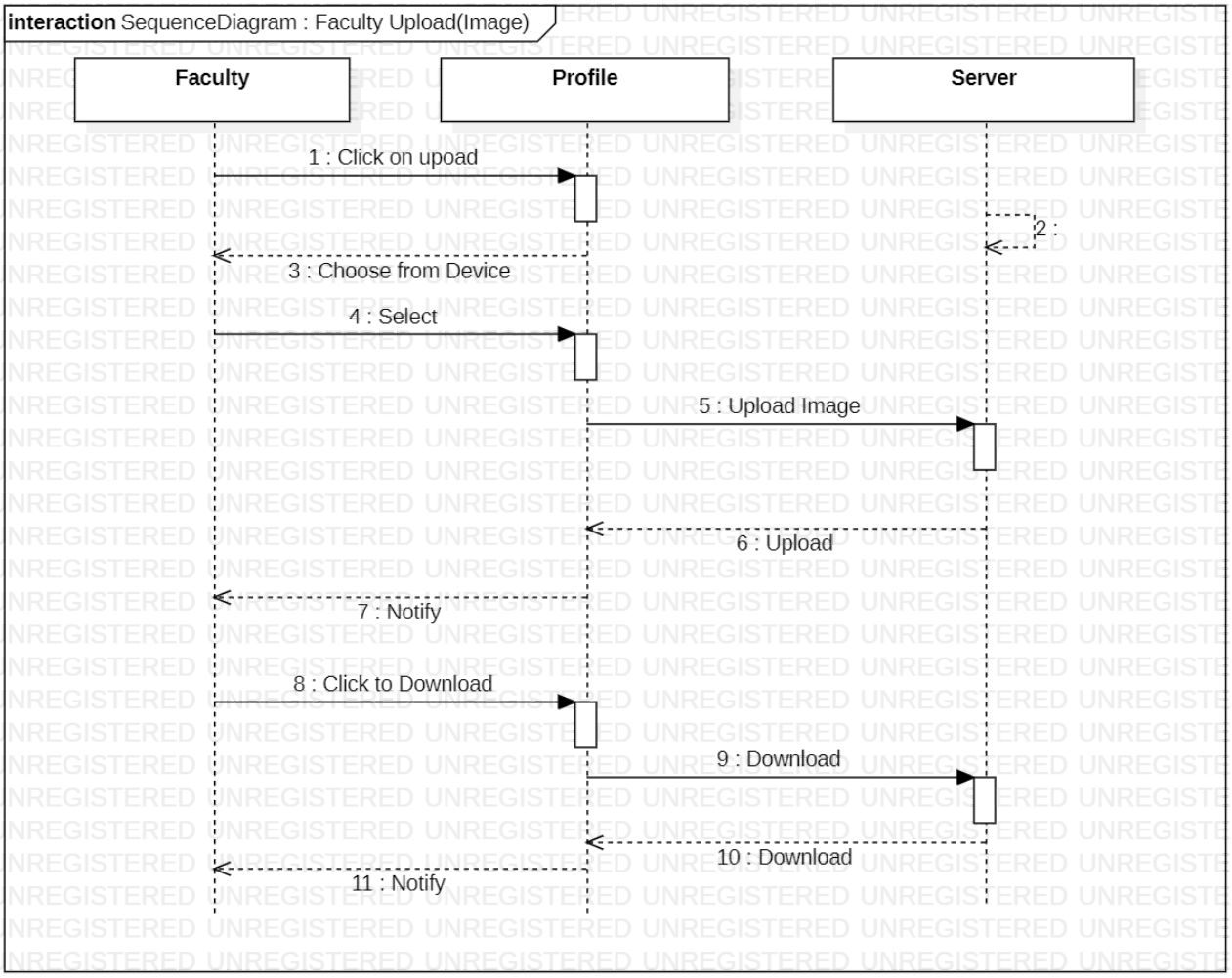
Sequence Diagram : Faculty Update



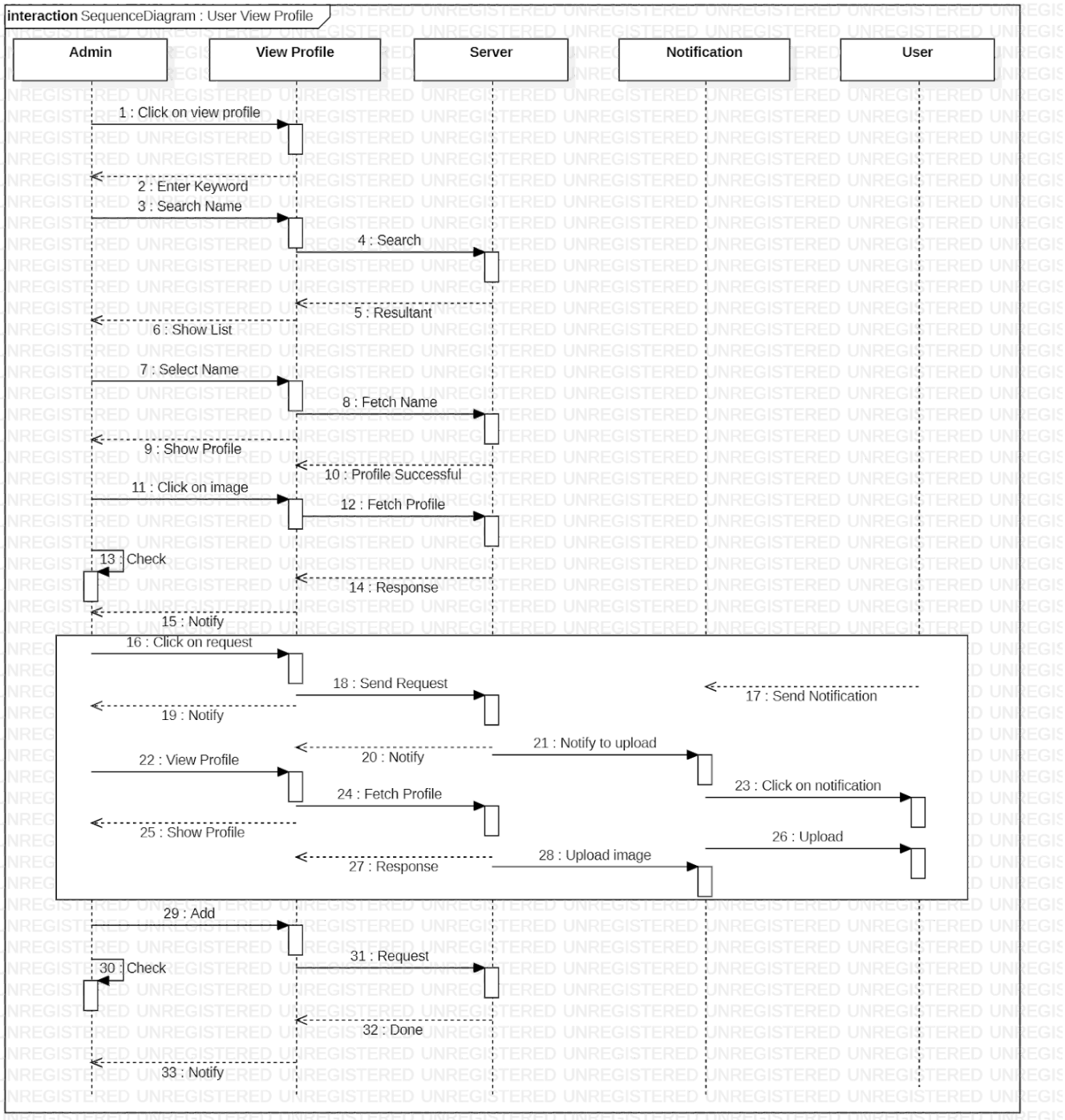
Sequence Diagram : User Upload(Image)


interaction SequenceDiagram : User Upload(Image)
User
Profile
Server
1 : Click on Upload
2 : Choose From Device
3 : Select
4 : Upload
5 : Upload
6 : Notify
7 : Click to Download
8 : Download
9 : Download
10 : Notify

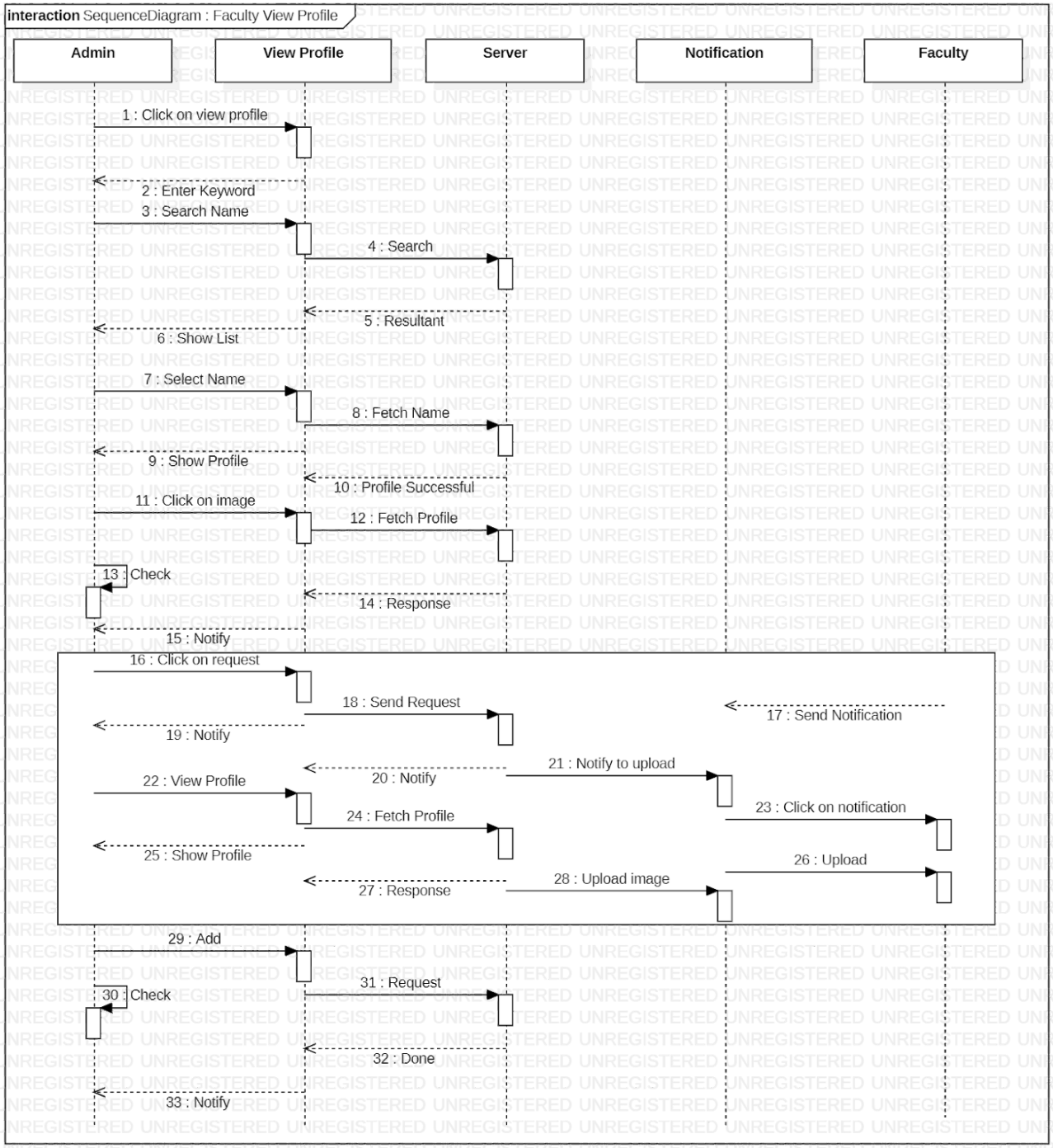
Sequence Diagram: Faculty Upload(Image)



Sequence Diagram : User View Profile

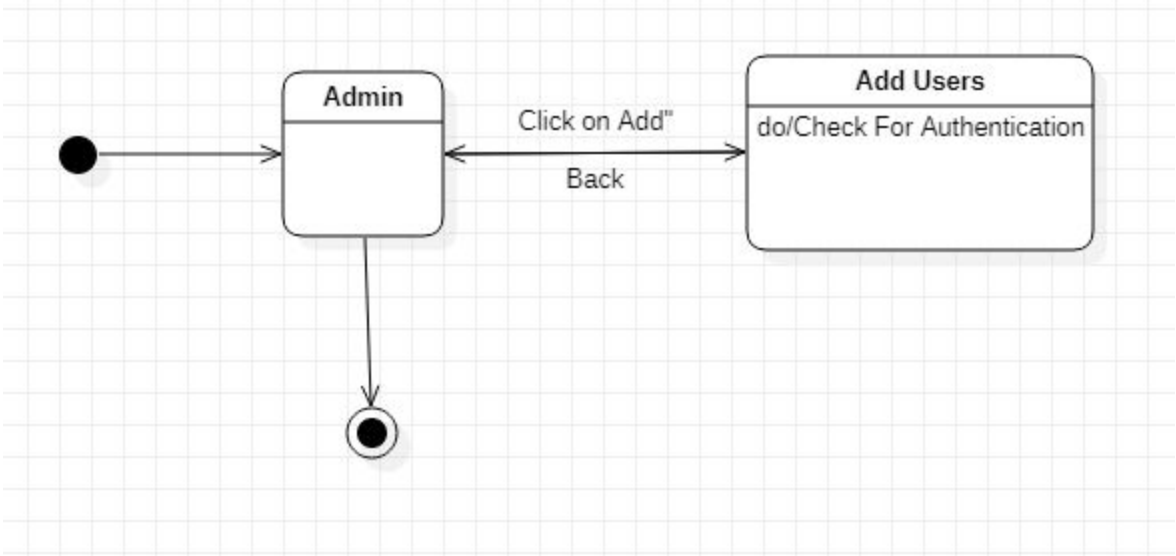


Sequence Diagram : Faculty View Profile

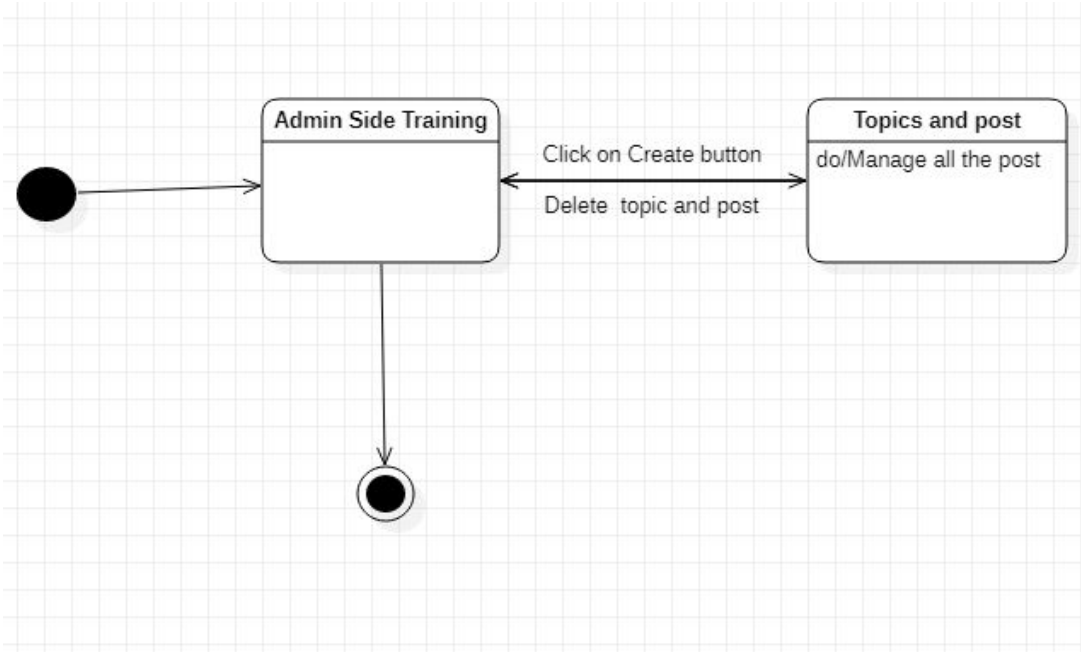


State Diagrams:

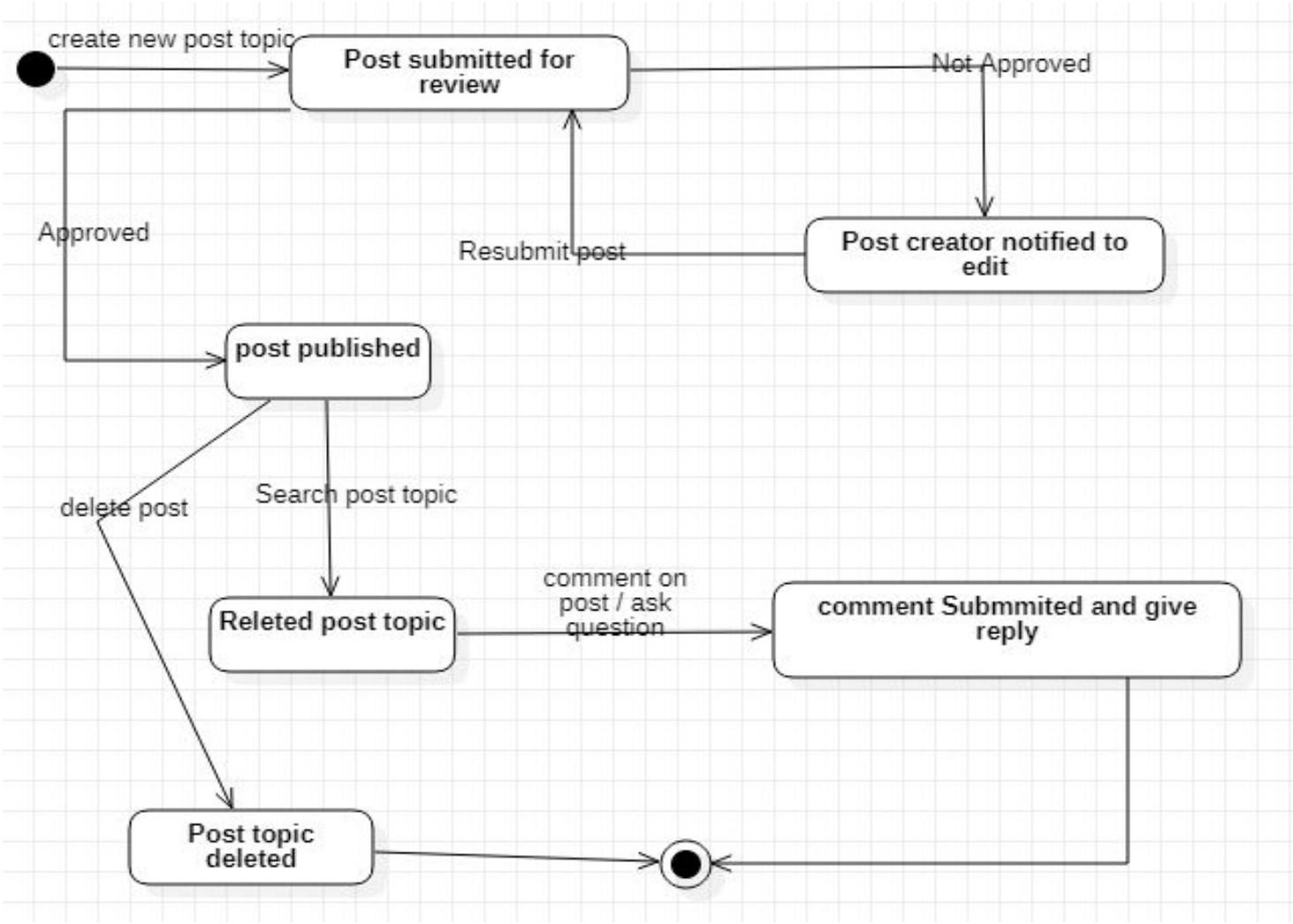
State Diagram:Admin About page



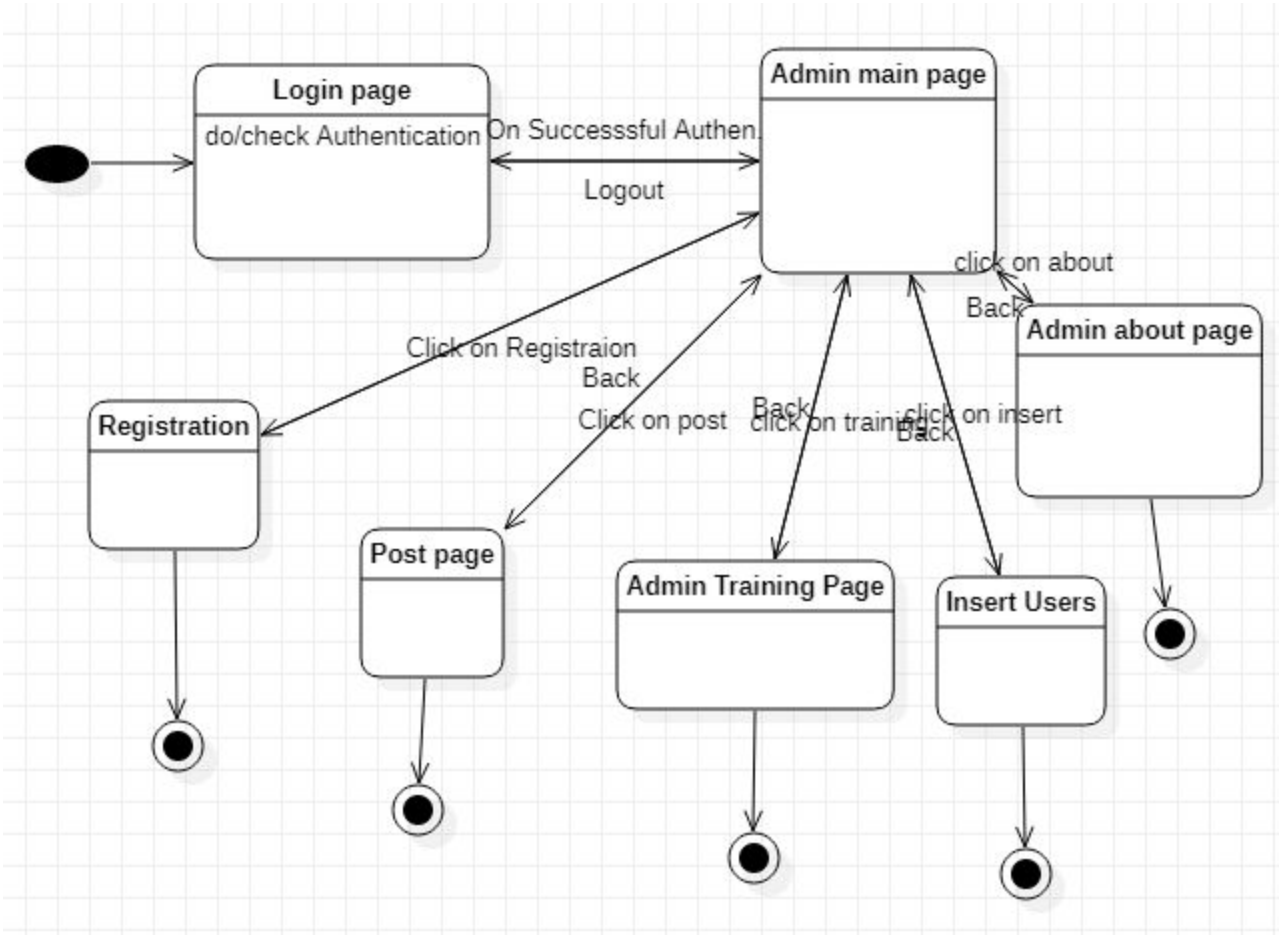
State Diagram:Admin Side Training



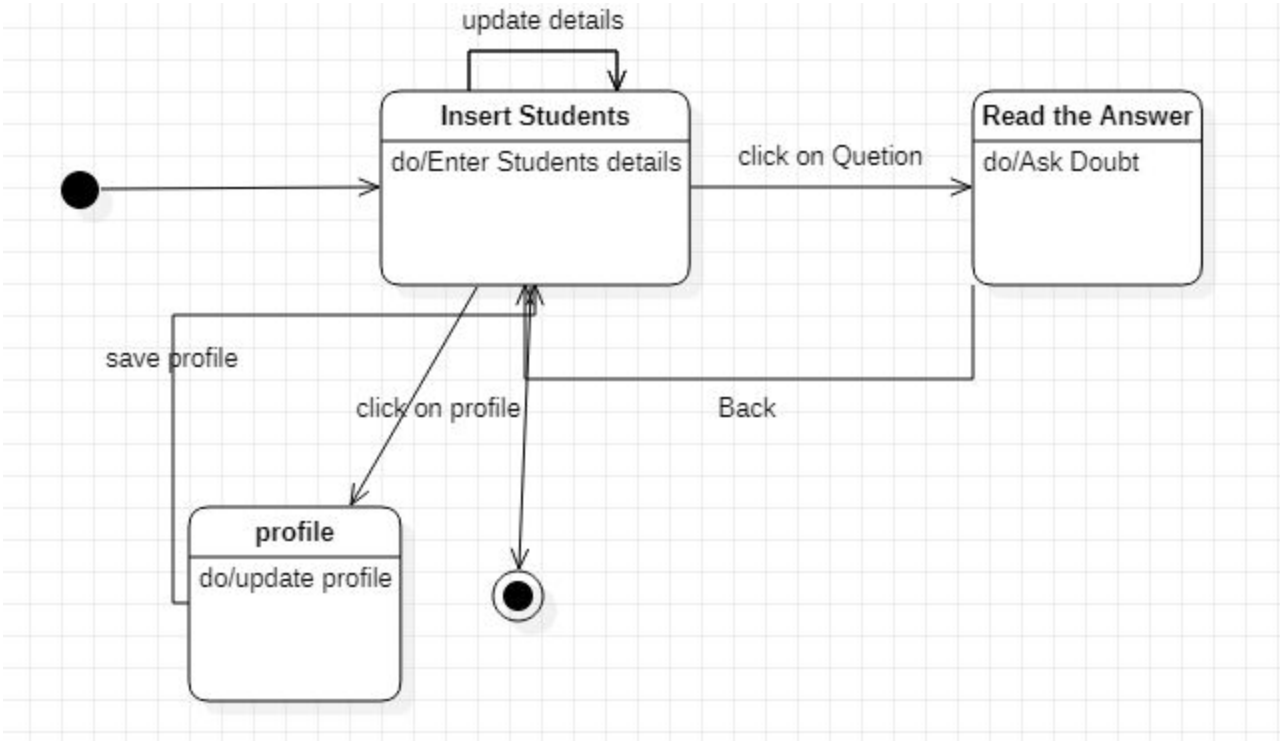
State Diagram: Post



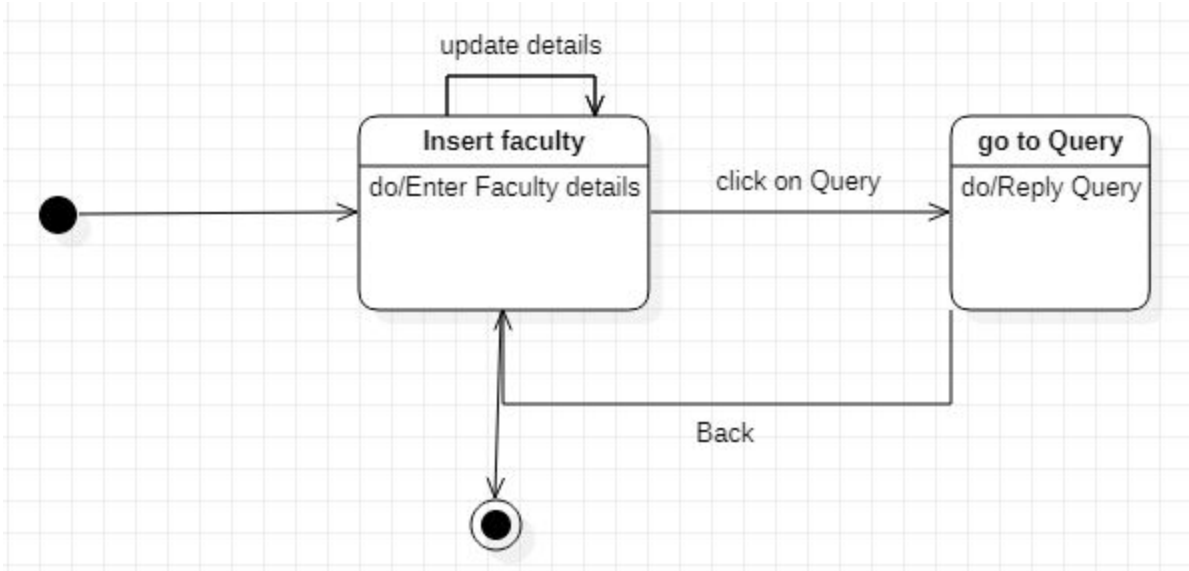
State Diagram: Admin main login page



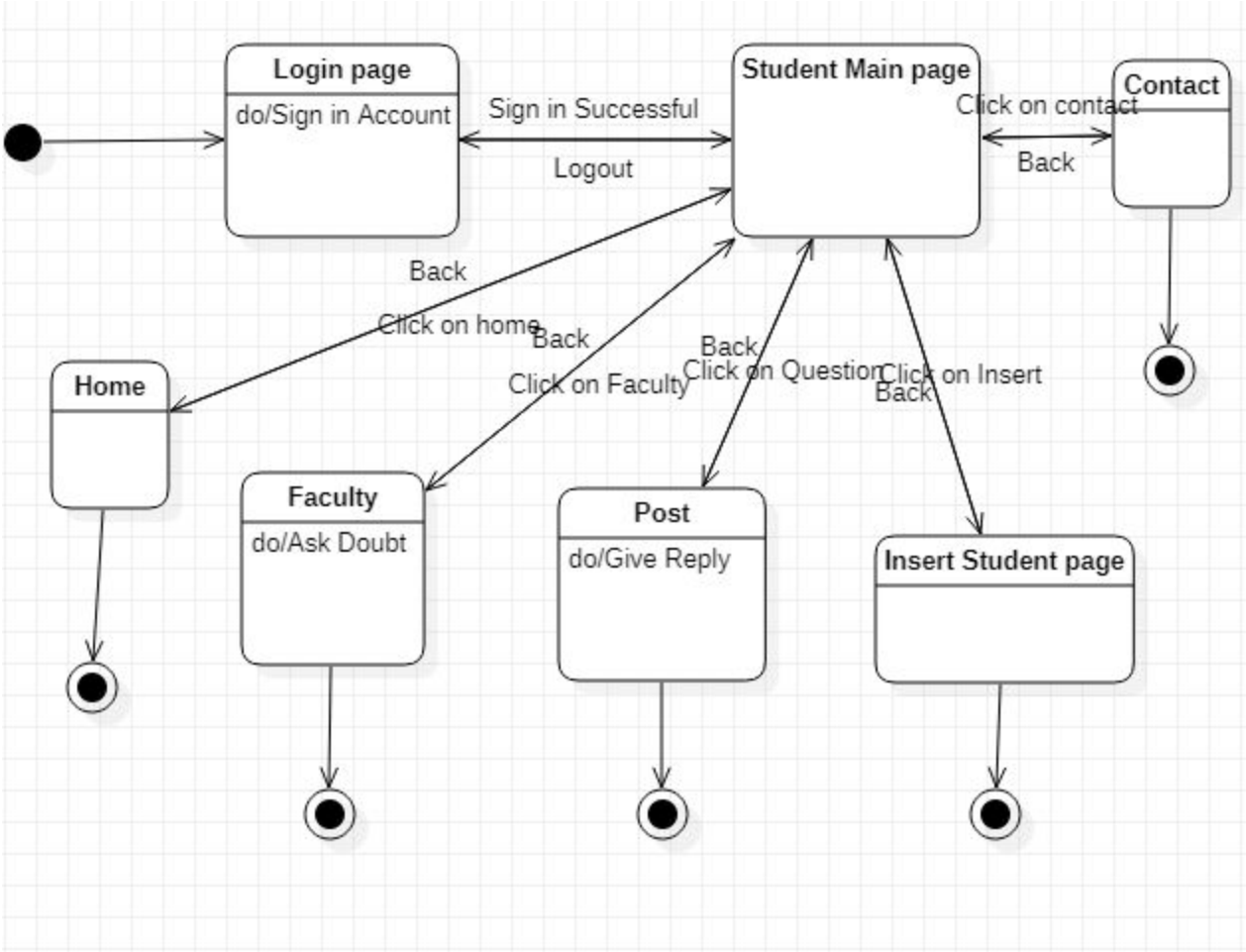
State Diagram:Insert Student



State Diagram : Faculty



State Diagram: Student Main Page



3.1 Logical Architecture Description

3.1.1 Sequence Diagram explanation:

Sequence Diagram:

Sequence Diagrams are interaction diagrams that detail how operations are carried out.
They capture the interaction between objects in the context of a collaboration.
Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent .

Purpose of sequence diagram:

a.)Model high-level interaction between active objects in a system
b.)Model the interaction between object instances within a collaboration that realizes a use case
c.)Model the interaction between objects within a collaboration that realizes an operation
d.)Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction)

Object Dimension:

a.)The horizontal axis shows the elements that are involved in the interaction
b.)Conventionally, the objects involved in the operation are listed from left to right according to when they take part in the message sequence.
However, the elements on the horizontal axis may appear in any order

Time Dimension:

The vertical axis represents time proceedings (or progressing) down the page.

Note that: Time in a sequence diagram is all about ordering, not duration.

The vertical space in an interaction diagram is not relevant for the duration of the interaction.

Sequence diagram at a glance:

Sequence Diagrams show elements as they interact over time and they are organized according to object (horizontally) and time (vertically):

Arrow line signifies there is a send message taken place.Response is being shown by dotted arrows.

3.1.2.1 AboutPage:

Admin(Administrator) is responsible for maintaining the website and monitors all the activities happening on the website. On clicking the Add button, it lands up the user to the Add Data page, which takes input from the students and faculty and creates some buttons like Create_topic, delete_topic,delete question/answer . On clicking Back, it returns to the parent page.

3.1.2.2 FAQ:

Student posts question on this page, which later can be viewed by all other students and can be answered by the Admin. On being answered by Admin, the student also get notification for the same.admin add topics ,post and manage all the posts,if posts are not approved then also delete the post.

3.1.2.3 Home:

Admin post any changes or any doubt topic , which later is visible on the Home Page of the students. It also allows students to answer after being their profile.Allows users to enter credentials, which are being checked for authentication in the back-end. After being authenticated successfully, it lands up on the Admin main page. It has various buttons like post, Registration,admin about page,admin training page, and About to land up on their respective pages, and click on Back to return to the Admin main page. On clicking on Logout, it returns to the parent page.

3.1.2.4 User Login Page:

It allows students to login with their college enrollmentID(college emaill id) and Admin to login with the username and password that are being registered in the database already. It loops being on same page until the correct information is not given. Allows users to choose google account, which is being checked for authentication in the back-end. Students update your profile, go to Search topic,search_question/answer,view_question/answer,ask_questions, . Users On clicking Logout, it returns to the parent page.

3.1.2.5 Faculty Login Page:

It allows students to login with their college enrollmentID(college emaill id) and Admin to login with the username and password that are being registered in the database already.Allows users to choose google account, which is being checked for authentication in the back-end. On being authenticated successfully, it lands up the user on the Faculty main page. Faculties can give replies to the student Query.

3.1.2.6 Faculty Update Profile:

Faculty can update their profile by either editing or adding to it.

3.1.2.7 User Update Profile:

User can update their profile by either editing or adding to it.

3.1.2.8 User Upload Image :

Users can upload image by choosing the appropriate file from their pc,which then can be viewed by Admin also.

3.1.2.9 View profile of others:

User can view profile.He/She can add query or can view the query of others.He/She can also view answer of others or can add their reply to it.

3.1.2.10 View Profile:

Admin can search for the name of the student, by the data being searched on the database and shown in response. Then the Admin can also

3.1.2 State Diagram Explanation:

A state diagram used to describe the behavior of a system considering all the possible states of an object when an event occurs. This behavior is represented and analyzed in a series of events that occur in one or more possible states. Each diagram represents objects and tracks the various states of these objects throughout the system.

Initial state is being shown by starting with a black dot. Final State is being shown by the black dot surrounded by an empty circle.

3.1.2.1 Admin About page: On clicking the Add button, it lands up the user to the Add Data page, which takes input from the students and faculty and creates some buttons like Create_topic, delete_topic,delete question/answer . On clicking Back, it returns to the parent page.

3.1.2.2 Admin Side Training: On Admin Training page admin add topics ,post and manage all the posts,if posts are not approved then also delete the post.

3.1.2.3 Admin main page: Allows users to enter credentials, which are being checked for authentication in the back-end. After being authenticated successfully, it lands up on the Admin main page. It has various buttons like post, Registration,admin about page,admin training page, and About to land up on their respective pages, and click on Back to return to the Admin main page. On clicking on Logout, it returns to the parent page.

3.1.2.4 Student Login: Allows users to choose google account, which is being checked for authentication in the back-end. Students update your profile, go to Search topic,search_question/answer,view_question/answer,ask_questions, . Users On clicking Logout, it returns to the parent page.

3.1.2.5 Faculty Login:Allows users to choose google account, which is being checked for authentication in the back-end. On being authenticated successfully, it lands up the user on the Faculty main page. Faculties can give replies to the student Query.

3.1.2.6 Student main page: After being authenticated successfully, it lands up on the Student main page.Now there are three button Contact Faculty,Post,Home page .After click on post they ask their Question ,post **Comments on Faculties replies.**

3.1.2.7 Post: On clicking the post button, it lands on the post page, which requires input data from users(faculty/students).In post page clicking on Question and answer button corresponding,it lands up in the question/answer page,here we can put question on corresponding topics,and on answer button type answer for it(question).

Class Descriptions:-

3.2 Class name: Admin

Description: Manages all logins, logouts, reports and other details of authentication

3.2.1 Method1: checkUser()

Input: user_id, user_password

Output: Launch the activity

Method Description: It will check if the user details are valid or not and then it will allow access to the details of given username.

3.2.2 Method2: deleteUser()

Input: user_password

Output: Launch the activity

Method Description: It will deactivate the account and will delete all the information of the user permanently.

3.2.3 Method3: updateUser()

Input:user_id, user_name, user_mobile, user_email, user_username, user_password, user_address

Output: Launch the activity

Method Description: It will update the account details that have been edited by the user in the database.

3.2.4 Method4: addUser()

Input: user_id, user_name, user_mobile, user_email, user_username, user_password, user_address (all details of user)

Output: Launch the activity, Create Account

Method Description: This will add a new user to the database.

3.2.5 Method5: checkFaculty()

Input: faculty_id, faculty_password

Output: Launch the activity

Method Description: It will check if the faculty details are valid or not and then it will allow access to the details.

3.2.6 Method6: deleteFaculty()

Input: faculty_password

Output: Launch the activity

Method Description: It will deactivate the account and will delete all the information of the faculty permanently.

3.2.7 Method7: updateFaculty()

Input: faculty_id, faculty_name, faculty_cabin_no, faculty_email, faculty_mobile, faculty_password

Output: Launch the activity

Method Description: It will update the account details that have been edited by the faculty in the database.

3.2.8 Method8: addFaculty()

Input:faculty_id, faculty_name, faculty_cabin_no, faculty_email, faculty_mobile, faculty_password (all details of faculty)

Output: Launch the activity, Create Account

Method Description: This will add a new faculty to the database.

3.2.9 Method9: logout()

Input: admin_id

Output: Logging out

Method Description: This will logout admin from the application.

3.2.10 Method10: login()

Input: admin_id,admin_password

Output: Logging in

Method Description: This will login admin from the application.

3.3 Class name: Faculty

Description: Manage all the operations of the faculties.

3.3.1 Method1: reportAdmin

Input: report, id

Output: Report is sent to admin

Method Description: This will help communicating with the admin.

3.3.2 Method2: addQuestion

Input: question, id

Output: Question is posted.

Method Description: This will put the question in the forum.

3.3.3 Method3: message

Input: message, sender_id, receiver_id

Output: Message is sent.

Method Description: Help communicating with the individual.

3.3.4 Method4: viewAnswer

Input: id

Output: Answer can now be seen

Method Description: This will show the answer.

3.3.5 Method5 :addAnswer

Input: answer, id

Output: Answer is posted.

Method Description: This will put the answer in the forum.

3.3.6 Method6: addRemarks

Input: remarks, sender_id, getters_id

Output: Remarks are given to post, query or answer.

Method Description: This will give remarks to a post, query or answer.

3.3.7 Method7: logout()

Input: faculty_id

Output: Logging out

Method Description: This will logout faculty from the application.

3.3.8 Method8: login()

Input: faculty_id, faculty_password

Output: Logging in

Method Description: This will login faculty from the application.

3.4 Class name: Users

Description: Manage all the operations of the users.

3.4.1 Method1: reportAdmin

Input: report, id

Output: Report is sent to admin

Method Description: This will help communicating with the admin.

3.4.2 Method2: addQuery

Input: query, id

Output: Query is posted.

Method Description: This will put the query in the forum.

3.4.3 Method3: message

Input: message, sender_id, receiver_id

Output: Message is sent.

Method Description: Help communicating with the individual.

3.4.4 Method4: viewAnswer

Input: id

Output: Answer can now be seen

Method Description: This will show the answer.

3.3.5 Method5 :addAnswer

Input: answer, id

Output: Answer is posted.

Method Description: This will put the answer in the forum.

3.4.6 Method6: viewQuestion

Input: id

Output: Question

Method Description: This will make users view the question.

3.4.7 Method7: logout()

Input: user_id

Output: Logging out

Method Description: This will logout users from the application.

3.4.8 Method8: login()

Input: user_id, user_password

Output: Logging in

Method Description: This will login users from the application.

3.5 Class name: Registration

Description: Manage all the operations of the registrations.

3.5.1 Method1: viewRegistration()

Input: registration_id

Output: Launch the activity

Method Description: This will make it view the registration.

3.5.2 Method2: addRegistration()

Input: registration_id, registration_user_id, registration_name, registration_type, registration_number, registration_date, registration_description

Output: Launch the activity

Method Description: This will add the registration.

3.5.3 Method3: editRegistration()

Input: registration_id, registration_user_id, registration_name, registration_type, registration_number, registration_date, registration_description

Output: Launch the activity

Method Description: This will edit the registration.

3.5.4 Method4: deleteRegistration()

Input: registration_id

Output: Launch the activity

Method Description: This will delete the registration.

3.5.5 Method5: searchRegistration()

Input: registration_id

Output: Launch the activity

Method Description: This will search the registration.

3.5.6 Method6: saveRegistration()

Input: registration_id, registration_user_id, registration_name, registration_type, registration_number, registration_date, registration_description

Output: Launch the activity

Method Description: This will save the registration.

3.6 Class name: Post

Description: Manage all the operations of the posts.

3.6.1 Method1: answerPost()

Input: post_id, post_user_id, post_description

Output: Launch the activity

Method Description: Post is answered by someone.

3.6.2 Method2: queryPost()

Input: post_user_id, post_title, post_type, post_description

Output: Launch the activity

Method Description: A query has been posted.

3.6.3 Method3: questionPost()

Input: post_user_id, post_title, post_type, post_description

Output: Launch the activity

Method Description: A question is posted.

3.7 Class name: Forum

Description: Manage all the operations of the forum.

3.7.1 Method1: addForum()

Input: forum_id, forum_name, forum_type, forum_description

Output: Launch the activity

Method Description: This will add the forum.

3.7.2 Method2: editRegistration()

Input: forum_id, forum_name, forum_type, forum_description

Output: Launch the activity

Method Description: This will edit the forum.

3.7.3 Method3: deleteForum()

Input: forum_id

Output: Launch the activity

Method Description: This will delete the forum.

3.7.4 Method4: updateForum()

Input: forum_id, forum_name, forum_type, forum_description

Output: Launch the activity

Method Description: This will update the forum.

3.7.5 Method5: searchForum()

Input: forum_id

Output: Launch the activity

Method Description: This will search the forum.

3.7.6 Method6: saveForum()

Input: forum_id, forum_name, forum_type, forum_description

Output: Launch the activity

Method Description: This will save the forum.

4.0 Execution Architecture

Runtime environment required is any device supporting PHP, mysql, Html and features to connect mysql to php files.

4.1 Reuse and relationships to other products

NIL

5. Design decisions and tradeoffs

The design decision to use three screens separately for admin ,faculty and student is to provide encapsulation. It may have been possible to get all the information on one screen. However, using more than one screen will keep the data of admin separate from the data being accessed by students.

A possible tradeoff when considering links is to use buttons instead of items in the menu. This design decision - to use buttons for navigating between screens - is to enhance visibility. Screen readers handle links slightly differently than they do buttons. All links and buttons are tab-able, but pressing the Space key or Enter triggers a button, whereas pressing the Enter key only triggers a link.

Text links in the menu bar located at the bottom of the PDA's screen can be hard to see. The tradeoff for buttons with descriptive labels rather than text links in the menu bar will be that navigation from screen to screen will be easier. Descriptive labels will let the user know where he is navigating. Buttons are larger than the text links located in the menu bar of the PDA. Therefore, it is easier for the user to locate the mechanisms needed to navigate from screen to screen

6.Some Code Snippets

Below are the some code snippets which we have written:

Classes:-

6.1 Class name: Admin

6.1.1 Method1: checkUser()

```
<?php
	if(!isset($_SESSION["fn"]))
		header("location:../index.php");
?>

<span style="float:right">
welcome<a href="aedit.php"><img src="../res/images/useri.jpg" class="imagedel"><?php echo
$_SESSION["fn"];?></a>, [ <a href="logout.php">log-out</a> ]
</span>
```

6.1.2 Method2: deleteUser()

```
//require("checkUser.php");
?>
<?php
$uid = $_GET['id'];
$qry = "DELETE FROM user WHERE user_id='".$uid."'";
$result = ExecuteQuery($qry);
if($result)
{header ("location:student_account.php");}
	else
	{echo "Not Done";}

?>
<?php require("footer.php")?>
```

6.1.3 Method3: updateUser()

```
<?php
	 $sql = "UPDATE user  SET  username = '".$_POST['un']."', fullname =
'".$_POST['fn']."',password='".$_POST['pwd']."',e_mail='".$_POST['e_mail']."',gender = '".$_POST['gender']."',
dob = '".$_POST['dob']."',uimg = '".$_POST['ima']."',address = '".$_POST['add']."',state =
'".$_POST['sta']."',country = '".$_POST['cou']."' WHERE user_id =$_SESSION[uid]";
//echo $sql;
$result=ExecuteNonQuery($sql);
if($result == 1)
{header("location:apupdate.php");}
	else
	{header("location:aedit.php");}
?>
<?php require("footer.php")?>
```

6.1.4 Method5: checkFaculty()

```
<?php session_start();
 require("header.php");
require("checkUser.php");
echo "<body style='background: url(../res/images/im6.jpg);'>";?>
<script type="text/javascript">
```

```
        document.getElementById("amanage").className="active";
</script>
<?php
                $sql="SELECT user_id, username, fullname, dob, e_mail from user where user.user_type =
'faculty' and user.user_acc_active = 0";
                $rows=ExecuteQuery($sql);
                echo "<table border='9'>";
                echo "<strong><tr><th>User
ID</th><th>Username</th><th>Fullname</th><th>DOB</th><th>Email</th><th>Approve</th></tr> </strong>";
                while($name_row=mysql_fetch_array($rows))
                {echo "<tr>";
                echo
"<td>$name_row[user_id]</td><td>$name_row[username]</td><td>$name_row[fullname]</td><td>$name_row[
dob]</td><td>$name_row[e_mail]</td><td><a href='a_fac.php?id=".$name_row[0]."'><img
src='../res/images/tick.png'  class='imagedel'/></a></td>";
                        }
                echo "</table>";
?>
<?php require("footer.php")?>
```

6.1.5 Method5: logout()

```
<?php require("header.php");
echo "<body style='background: url(../res/images/im6.jpg);'>";?>
<?php
session_start();
session_destroy();
?>
<h1>log out</h1>
<p>You have logged out.  <a href="../index.php">Click here</a> to login again.
</p>
<?php require("footer.php");?>
```

6.2 Class name: Faculty

6.2.1 Method 1: login()

```
<?php
if (isset ($_GET["act"]))
if ($_GET["act"] == "invalid")
echo "Invalid User Id / password";
elseif ($_GET["act"] == "not_fac")
            echo "Faculty not approved yet";

?>
```

6.2.2 Method 2: add Question()

```
<script type="text/javascript">
function check(f)
{
if(f.head.value=="")
{
document.getElementById("a").innerHTML="Please,Enter the heading";
//alert("Please,Enter The Heading");
f.head.focus();
return false;

}
else if(f.ta.value=="")
{
document.getElementById("b").innerHTML="Please,Enter The Question";
//alert("Please,Enter The Question")}
f.ta.focus();
return false;
}
  else
  return true;
}
```

6.2.3 Method 3: add Answer()

```
<script type="text/javascript">
function check(f)
{
if(f.ata.value=="")
{
document.getElementById("spuid").innerHTML = "Please, Enter Answer.";
f.ata.focus();
return false;
}
else
return true;
}
</script>
```

6.2.4 Method 4 : message()

```
<script type="text/javascript">
function check(f)
```

```
{
if(f.tt.value=="")
{
document.getElementById("a").innerHTML="Please,Enter the message";
//alert("Please,Enter The Question");
f.tt.focus();
```

6.2.5 Method 5 : logout()

```
<?php require("header.php");
echo "<body style='background: url(../res/images/im6.jpg);'>";?>
<?php
session_start();
session_destroy();
?>
<h1>log out</h1>
<p>
        You have logged out.  <a href="../index.php">Click here</a> to login again.
</p>
<?php require("footer.php");?>
```

6.3 Class name: Forum

6.3.1 Method 1 : addForum()

```
<?php session_start();
 require("header.php");

 if ($_SESSION["fn"] == null){
        header("location:unreg.php");
        exit();
 }

 require("checkUser.php");
 echo "<body style='background: url(res/images/im6.jpg);'>";
 ?>

 <script type="text/javascript">
        document.getElementById("aforum").className="active";
</script>

<?php
        $topic = ExecuteQuery ("SELECT * FROM topic");
```

```
        while ($r1 = mysql_fetch_array($topic))
        {echo "<div class='heading'>$r1[topic_name]</div>";
                $stopic = ExecuteQuery ("SELECT * FROM subtopic WHERE topic_id=$r1[topic_id]");
        while ($r2 = mysql_fetch_array ($stopic) )
                        {echo "<div class='box'>";
                                echo "<div class='sub-heading'>
        <a href='questions.php?id=$r2[subtopic_id]'> $r2[subtopic_name]</a>
                                        </div>";
                                echo "<p>$r2[subtopic_description]</p>";
                                echo "</div>";
                        }
        }


?>

<?php require("footer.php"); ?>
```

6.3 Class nameRegistration

6.3.1 Method 1 : addRegistration()

```
<?php  require("header.php");
echo "<body style='background: url(res/images/im6.jpg);'>"; ?>
<script type="text/javascript">
        function check(form1)
        {
                if(
                form1.u_name.value == "" ||
                form1.f_name.value == "" ||
                form1.pwd.value == "" ||
                form1.e_mail.value == ""||
                form1.gender.value == ""||
                form1.dob.value == ""||
                form1.add.value == ""||
                form1.sta.value == "" ||
                form1.cou.value == "" )
                {

                        if (form1.u_name.value == "")
                        {
                                document.getElementById("a").innerHTML = "Please, Enter user name.";
                                //alert("Please, Enter The Username");
                                form1.u_name.focus();
```

```
}
else
{
	document.getElementById("a").innerHTML = "";
	//alert("Please, Enter The Username");
	form1.u_name.focus();

}
 if (form1.f_name.value == "")
{
	document.getElementById("b").innerHTML = "Please, Enter full name.";
	//alert ("Please,Please Enter The Fullname");
	form1.f_name.focus();
}
else
{
	document.getElementById("b").innerHTML = "";
	//alert ("Please,Please Enter The Fullname");
	form1.f_name.focus();
}
 if (form1.pwd.value == "")
{
	document.getElementById("c").innerHTML = "Please, Enter password.";
	//alert ("Please,Please Enter The Password");
	form1.pwd.focus();
}
else
{
	document.getElementById("c").innerHTML = "";
	//alert ("Please,Please Enter The Password");
	form1.pwd.focus();

}

 if (form1.e_mail.value == "")
{
	document.getElementById("d").innerHTML = "Please, Enter e-mail
address.";
	//alert ("Please,Please Enter The E-mail Address");
	form1.e_mail.focus();
}
else
{
```

```
                document.getElementById("d").innerHTML = "";
                //alert ("Please,Please Enter The E-mail Address");
                form1.e_mail.focus();
        }

         if (form1.dob.value == "")
        {
                document.getElementById("e").innerHTML = "Please, Enter date of
birth.";
//              alert ("Please,Please Enter The Date Of Birth");
                form1.dob.focus();

        }
        else
        {
                document.getElementById("e").innerHTML = "";
//              alert ("Please,Please Enter The Date Of Birth");
                form1.dob.focus();

        }
         if (form1.add.value == "")
        {
                document.getElementById("f").innerHTML = "Please, Enter address.";
                //alert ("Please,Please Enter The Address");
                form1.add.focus();

        }
        else
        {
                document.getElementById("f").innerHTML = "";
                //alert ("Please,Please Enter The Address");
                form1.add.focus();
        }
         if (form1.sta.value == "")
        {
                document.getElementById("g").innerHTML = "Please, Enter state.";
                //alert ("Please,Please Enter The State");
                form1.sta.focus();

        }
        else
        {
                document.getElementById("g").innerHTML = "";
```

```
                        form1.sta.focus();

                }
                 if (form1.cou.value == "")
                {
                        document.getElementById("h").innerHTML = "Please, Enter country.";
                        form1.cou.focus();

                }
                else
                {
                        document.getElementById("h").innerHTML = "";
                        form1.cou.focus();
                }
                return false;
        }
        else
                return true;
    }
</script>



<h1>Register Student</h1>
<form action="registerH.php" method="post" onsubmit="return check(form1)"
enctype="multipart/form-data" name = "form1">
<table>
<tr><td>Username</td><td>:</td><td><input type="text" name="u_name" ><span id='a' style="color:
red;"></span></td></tr>
<tr><td>Fullname</td><td>:</td><td><input type="text" name="f_name"><span id='b' style="color:
red;"></span></td></tr>
<tr><td>Password</td><td>:</td><td><input type="password" name="pwd"><span id='c' style="color:
red;"></span></td></tr>
<tr><td>E_Mail</td><td>:</td><td><input type="email"
pattern="[a-zA-Z0-9.-_]{1,}@[a-zA-Z.-]{2,}[.]{1}[a-zA-Z]{2,}" name="e_mail"><span id='d' style="color:
red;"></span></td></tr>
<tr><td>Gender</td><td>:</td><td><input type="radio" name="gender" value="1"
checked="checked">male <input type="radio" name="gender" value="2">female<span id='spuid'
style="color: red;"></span></td></tr>
<tr><td>Date Of Birth</td><td>:</td><td><input type="text" name="dob" ><span id='e' style="color:
red;"></span></td></tr>
<tr><td>Image</td><td>:</td><td><input type="file" name="ima"></td></tr>
```

```
<tr><td>Address</td><td>:</td><td><textarea rows="3" cols="15" name="add"></textarea><span
id='f' style="color: red;"></span></td></tr>
<tr><td>State</td><td>:</td><td><input type="text" name="sta"><span id='g' style="color:
red;"></span></td></tr>
<tr><td>Country</td><td>:</td><td><input type="text" name="cou"><span id='h' style="color:
red;"></span></td></tr>

<tr><td><input type="submit" value="Submit"></td><td></td><td><input type="reset"
value="Reset"></td></tr></table></form>

<?php require("footer.php"); ?>
```

Java Script : Script.js

```
/* begin Page */



var artEventHelper = {
	'bind': function(obj, evt, fn) {
		if (obj.addEventListener)
			obj.addEventListener(evt, fn, false);
		else if (obj.attachEvent)
			obj.attachEvent('on' + evt, fn);
		else
			obj['on' + evt] = fn;
	}
};

var userAgent = navigator.userAgent.toLowerCase();
var browser = {
	version: (userAgent.match(/.+(?:rv|it|ra|ie)[\/: ]([\d.]+)/) || [])[1],
	safari: /webkit/.test(userAgent) && !/chrome/.test(userAgent),
	chrome: /chrome/.test(userAgent),
	opera: /opera/.test(userAgent),
	msie: /msie/.test(userAgent) && !/opera/.test(userAgent),
	mozilla: /mozilla/.test(userAgent) && !/(compatible|webkit)/.test(userAgent)
};

var artLoadEvent = (function() {
```

```
var list = [];

var done = false;
var ready = function() {
	if (done) return;
	done = true;
	for (var i = 0; i < list.length; i++)
		list[i]();
};

if (document.addEventListener && !browser.opera)
	document.addEventListener('DOMContentLoaded', ready, false);

if (browser.msie && window == top) {
	(function() {
		try {
			document.documentElement.doScroll('left');
		} catch (e) {
			setTimeout(arguments.callee, 10);
			return;
		}
		ready();
	})();
}

if (browser.opera) {
	document.addEventListener('DOMContentLoaded', function() {
		for (var i = 0; i < document.styleSheets.length; i++) {
			if (document.styleSheets[i].disabled) {
				setTimeout(arguments.callee, 10);
				return;
			}
		}
		ready();
	}, false);
}

if (browser.safari) {
	var numStyles;
	(function() {
		if (document.readyState != 'loaded' && document.readyState != 'complete') {
```

```
                        setTimeout(arguments.callee, 10);
                        return;
                }
                if ('undefined' == typeof numStyles) {
                        numStyles = document.getElementsByTagName('style').length;
                        var links = document.getElementsByTagName('link');
                        for (var i = 0; i < links.length; i++) {
                                numStyles += (links[i].getAttribute('rel') == 'stylesheet') ? 1 : 0;
                        }
                        if (document.styleSheets.length != numStyles) {
                                setTimeout(arguments.callee, 0);
                                return;
                        }
                }
                ready();
        })();
    }

    artEventHelper.bind(window, 'load', ready);

    return ({
        add: function(f) {
            list.push(f);
        }
    })
})();

(function() {
    // fix ie blinking
    var m = document.uniqueID && document.compatMode && !window.XMLHttpRequest &&
document.execCommand;
    try { if (!!m) { m('BackgroundImageCache', false, true); } }
    catch (oh) { };
})();

function xGetElementsByClassName(clsName, parentEle, tagName) {
    var elements = null;
    var found = [];
    var s = String.fromCharCode(92);
    var re = new RegExp('(?:^|' + s + 's+)' + clsName + '(?:$|' + s + 's+)');
    if (!parentEle) parentEle = document;
    if (!tagName) tagName = '*';
    elements = parentEle.getElementsByTagName(tagName);
```

```
	if (elements) {
		for (var i = 0; i < elements.length; ++i) {
			if (elements[i].className.search(re) != -1) {
				found[found.length] = elements[i];
			}
		}
	}
	return found;
}

var styleUrlCached = null;
function GetStyleUrl() {
	if (null == styleUrlCached) {
		var ns;
		styleUrlCached = '';
		ns = document.getElementsByTagName('link');
		for (var i = 0; i < ns.length; i++) {
			var l = ns[i];
			if (l.href && /style\.css(\?.*)?$/.test(l.href)) {
				return styleUrlCached = l.href.replace(/style\.css(\?.*)?$/, '');
			}
		}

		ns = document.getElementsByTagName('style');
		for (var i = 0; i < ns.length; i++) {
			var matches = new
RegExp('import\\s+"([^"]+\\/)style\\.css"').exec(ns[i].innerHTML);
			if (null != matches && matches.length > 0)
				return styleUrlCached = matches[1];
		}
	}
	return styleUrlCached;
}

function fixPNG(element) {
	if (/MSIE (5\.5|6).+Win/.test(navigator.userAgent)) {
		var src;
		if (element.tagName == 'IMG') {
			if (/\.png$/.test(element.src)) {
				src = element.src;
				element.src = GetStyleUrl() + 'images/spacer.gif';
			}
		}
```

```
		else {
			src = element.currentStyle.backgroundImage.match(/url\("(.+\.png)"\)/i);
			if (src) {
				src = src[1];
				element.runtimeStyle.backgroundImage = 'none';
			}
		}
		if (src) element.runtimeStyle.filter =
"progid:DXImageTransform.Microsoft.AlphaImageLoader(src='" + src + "')";
	}
}

function artHasClass(el, cls) {
   return (el && el.className && (' ' + el.className + ' ').indexOf(' ' + cls + ' ') != -1);
}/* end Page */

/* begin Menu */
function Insert_Separators() {
	var menus = xGetElementsByClassName("art-menu", document);
	for (var i = 0; i < menus.length; i++) {
		var menu = menus[i];
		var childs = menu.childNodes;
		var listItems = [];
		for (var j = 0; j < childs.length; j++) {
			var el = childs[j];
			if (String(el.tagName).toLowerCase() == "li") listItems.push(el);
		}
		for (var j = 0; j < listItems.length - 1; j++) {
			var span = document.createElement('span');
			span.className = 'art-menu-separator';
			var li = document.createElement('li');
			li.appendChild(span);
			listItems[j].parentNode.insertBefore(li, listItems[j].nextSibling);
		}
	}
}
artLoadEvent.add(Insert_Separators);

function Menu_IE6Setup() {
	var isIE6 = navigator.userAgent.toLowerCase().indexOf("msie") != -1
   && navigator.userAgent.toLowerCase().indexOf("msie 7") == -1;
	if (!isIE6) return;
	var aTmp2, i, j, oLI, aUL, aA;
```

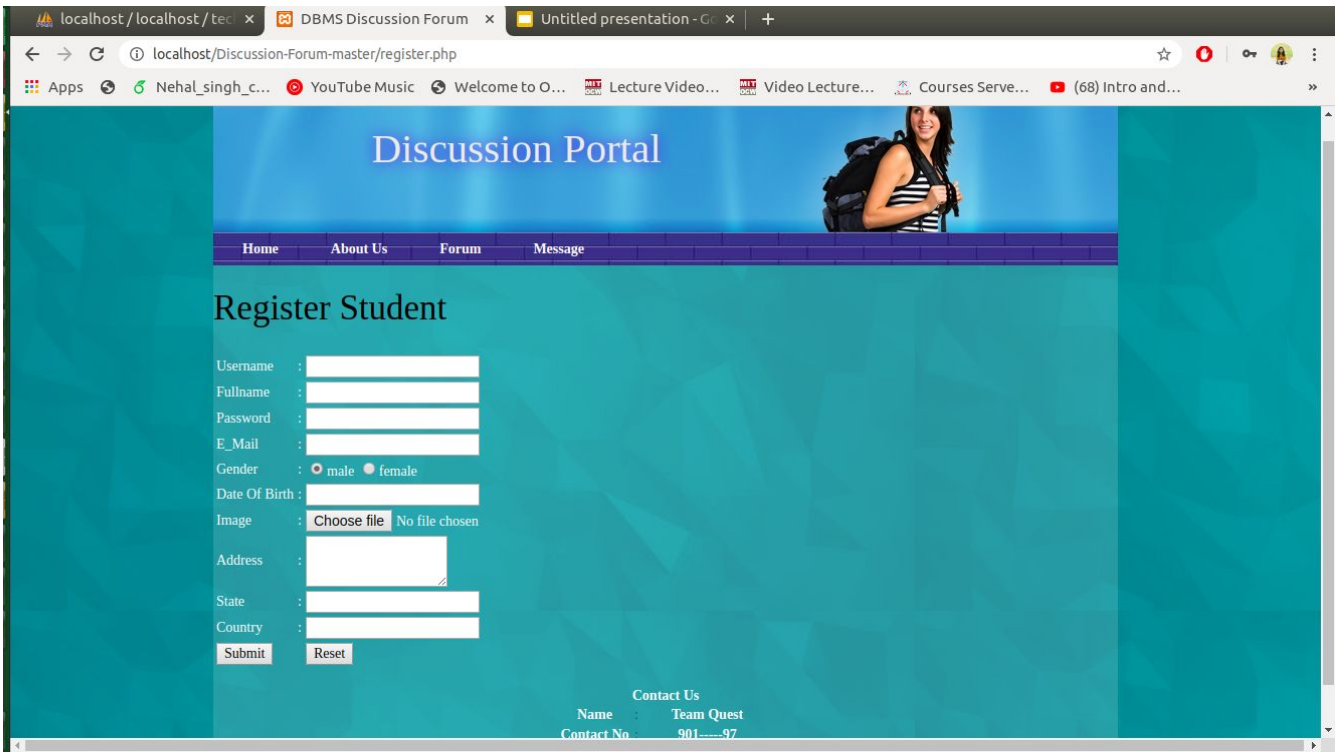
```
	var aTmp = xGetElementsByClassName("art-menu", document, "ul");
	for (i = 0; i < aTmp.length; i++) {
		aTmp2 = aTmp[i].getElementsByTagName("li");
		for (j = 0; j < aTmp2.length; j++) {
			oLI = aTmp2[j];
			aUL = oLI.getElementsByTagName("ul");
			if (aUL && aUL.length) {
				oLI.UL = aUL[0];
				aA = oLI.getElementsByTagName("a");
				if (aA && aA.length)
					oLI.A = aA[0];
				oLI.onmouseenter = function() {
					this.className += " art-menuhover";
					this.UL.className += " art-menuhoverUL";
					if (this.A) this.A.className += " art-menuhoverA";
				};
				oLI.onmouseleave = function() {
					this.className = this.className.replace(/art-menuhover/, "");
					this.UL.className =
this.UL.className.replace(/art-menuhoverUL/, "");
					if (this.A) this.A.className =
this.A.className.replace(/art-menuhoverA/, "");
				};
			}
		}
	}
}
artLoadEvent.add(Menu_IE6Setup);
/* end Menu */

/* begin Button */
function artButtonsSetupJsHover(className) {
	var tags = ["input", "a", "button"];
	for (var j = 0; j < tags.length; j++){
		var buttons = xGetElementsByClassName(className, document, tags[j]);
		for (var i = 0; i < buttons.length; i++) {
			var button = buttons[i];
			if (!button.tagName || !button.parentNode) return;
			if (!artHasClass(button.parentNode, 'art-button-wrapper')) {
				if (!artHasClass(button, 'art-button')) button.className += ' art-button';
				var wrapper = document.createElement('span');
				wrapper.className = "art-button-wrapper";
				if (artHasClass(button, 'active')) wrapper.className += ' active';
```

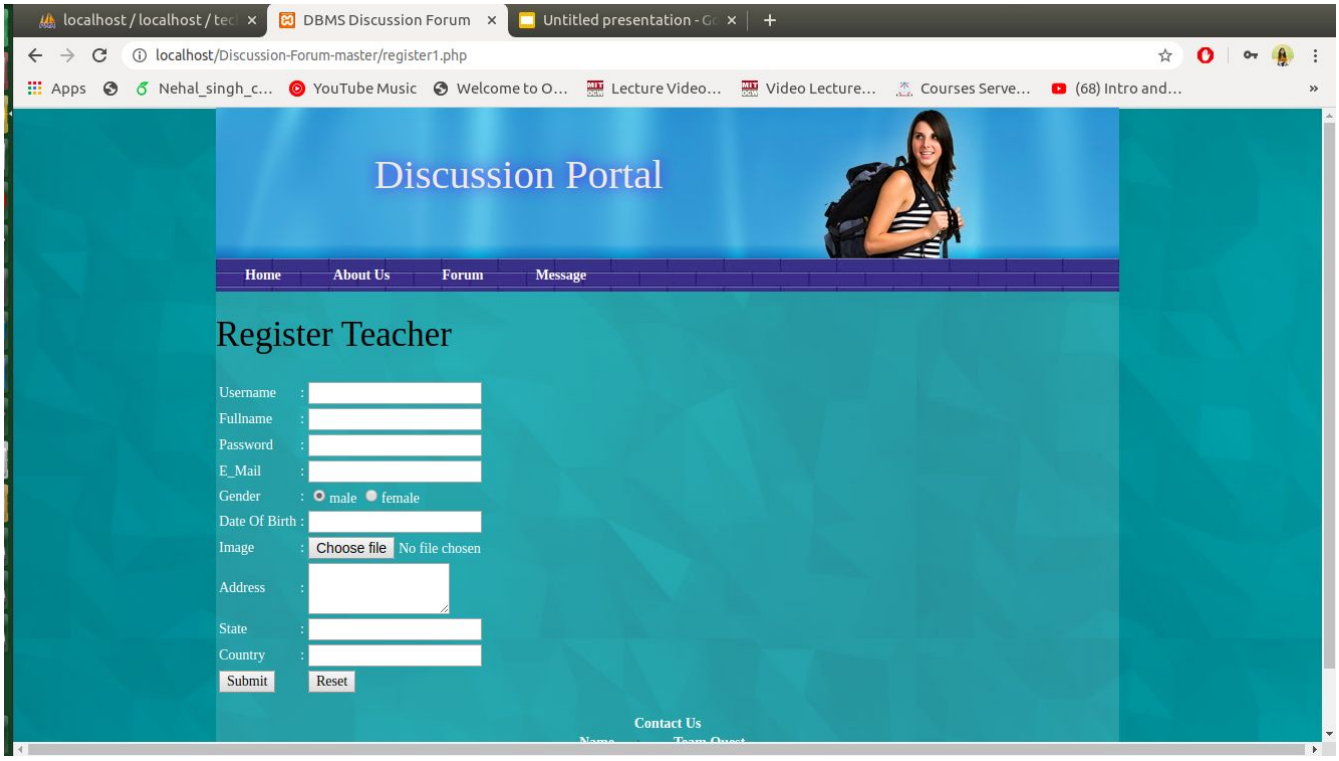
```
                        var spanL = document.createElement('span');
                        spanL.className = "l";
                        spanL.innerHTML = " ";
                        wrapper.appendChild(spanL);
                        var spanR = document.createElement('span');
                        spanR.className = "r";
                        spanR.innerHTML = " ";
                        wrapper.appendChild(spanR);
                        button.parentNode.insertBefore(wrapper, button);
                        wrapper.appendChild(button);
                }
                artEventHelper.bind(button, 'mouseover', function(e) {
                        e = e || window.event;
                        wrapper = (e.target || e.srcElement).parentNode;
                        wrapper.className += " hover";
                });
                artEventHelper.bind(button, 'mouseout', function(e) {
                        e = e || window.event;
                        button = e.target || e.srcElement;
                        wrapper = button.parentNode;
                        wrapper.className = wrapper.className.replace(/hover/, "");
                        if (!artHasClass(button, 'active')) wrapper.className =
wrapper.className.replace(/active/, "");
                });
                artEventHelper.bind(button, 'mousedown', function(e) {
                        e = e || window.event;
                        button = e.target || e.srcElement;
                        wrapper = button.parentNode;
                        if (!artHasClass(button, 'active')) wrapper.className += " active";
                });
                artEventHelper.bind(button, 'mouseup', function(e) {
                        e = e || window.event;
                        button = e.target || e.srcElement;
                        wrapper = button.parentNode;
                        if (!artHasClass(button, 'active')) wrapper.className =
wrapper.className.replace(/active/, "");
                });
        }
    }
}
artLoadEvent.add(function() { artButtonsSetupJsHover("art-button"); });
/* end Button */
```

ScreenShots for some GUI/forms created.

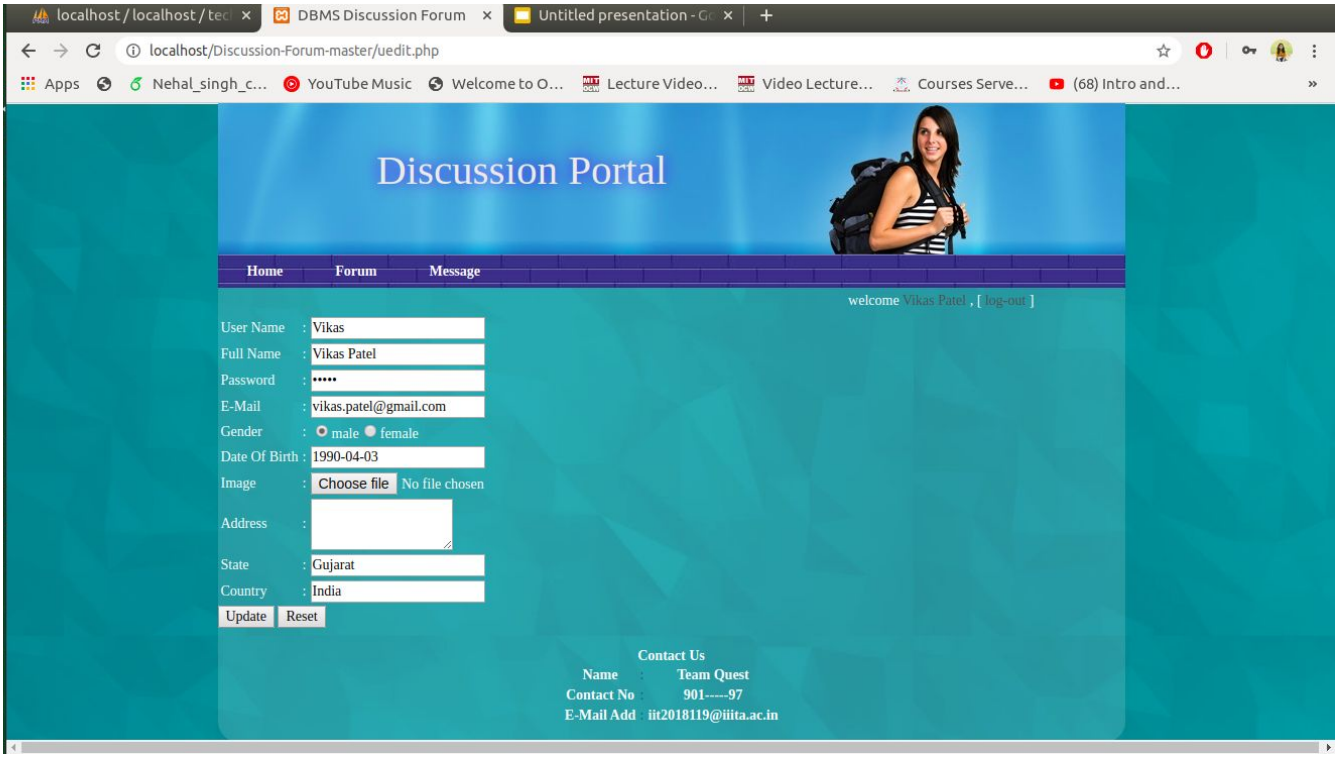
- **FORM FOR STUDENT TO REGISTER**



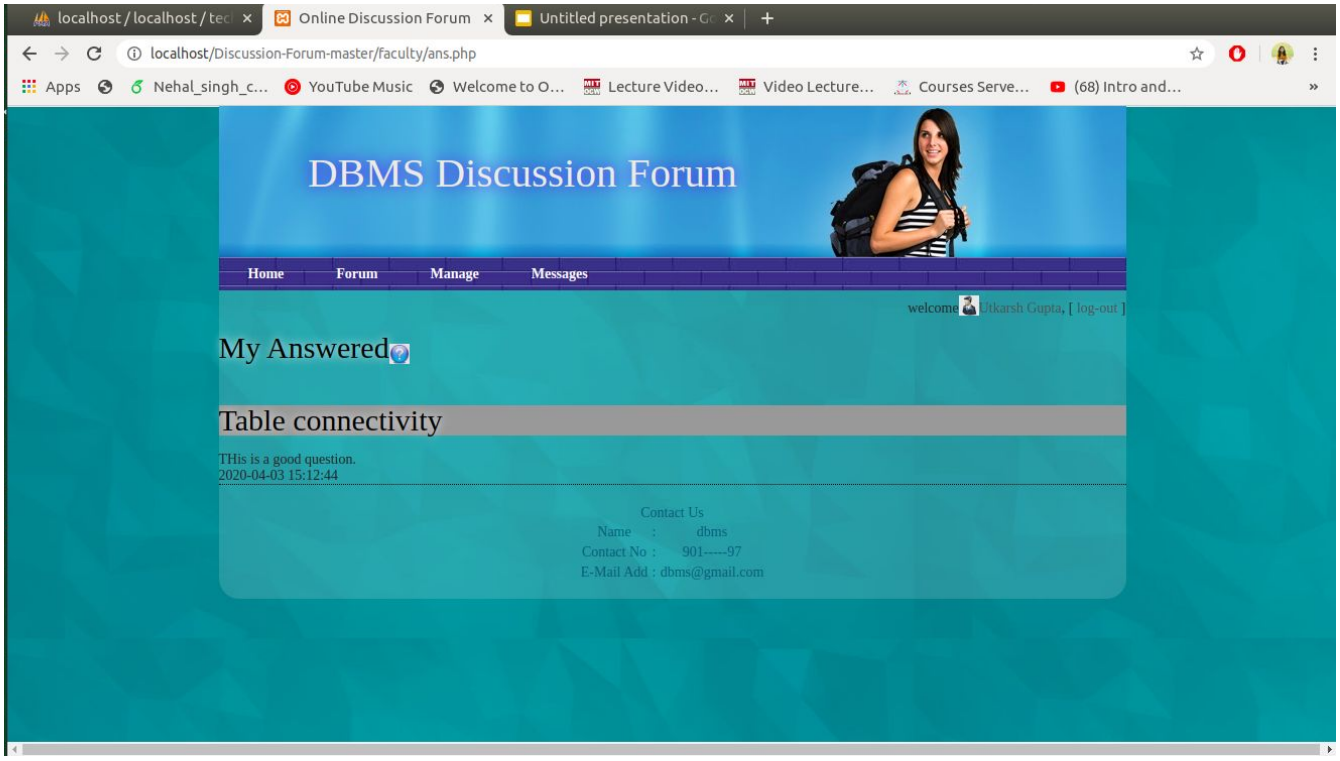
- **FORM FOR FACULTY TO REGISTER**



- **Registration process:**



- **After Login through Registered User**



- **We can go to “reported post”, post may be deleted by admin**