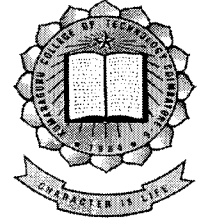
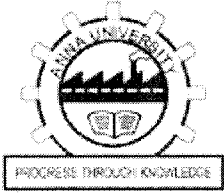


P-3118



**DEVELOPMENT OF WEB PORTAL TO PROMOTE
E-LEARNING**

A PROJECT REPORT



Submitted by

SURESH.M

71206104051

TAMILSELVAN.K.M

71206104052

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE

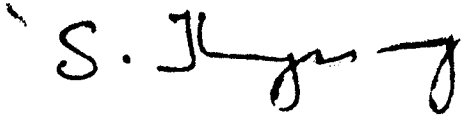
ANNA UNIVERSITY:: CHENNAI 600 025

APRIL 2010

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report “**WEB PORTAL TO PROMOTE E-LEARNING**” is the bonafide work of “**SURESH.M**” and “**TAMILSELVAN.K.M**” who carried out the project under my supervision.



SIGNATURE

Dr. S. Thangasamy

Dean / CSE

Department of Computer Science
Kumaraguru College of Technology,
Chinnavedampatti Post,
Coimbatore – 641606



SIGNATURE

Mr. K.Sivan Arul Selvan

Supervisor

Senior Lecturer

Department of Computer Science
Kumaraguru College of Technology,
Chinnavedampatti Post,
Coimbatore – 641606

The candidates with University Register Nos. **71206104051** and **71206104052** were examined by us in the project viva-voce examination held on**16.04.2010**.....



INTERNAL EXAMINER



EXTERNAL EXAMINER



ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made this possible and whose constant guidance and encouragement crowns all efforts with success.

We are extremely grateful to Dr.S.Ramachandran, Principal, Kumaraguru College of Technology for having given us this opportunity to embark on this project.

We express our sincere and heartfelt thanks to **Dr. S. Thangasamy**, Dean, Department of Computer Science and Engineering, for his kind guidance and support.

We would like to express our sincere thanks to our project coordinator **Mrs. P. Devaki**, Associate Professor, Department of Computer Science and Engineering, for her valuable guidance during the course of the project.

We would also like to thank our class advisor **Mrs. R. Kalaiselvi** Senior Lecturer, for her constant support and guidance.

We would like to thank our guide **Mr. K.Sivan Arul Selvan**, Senior Lecturer, without whose motivation and guidance we would not have been able to embark on a project of this magnitude. We express our sincere thanks for his valuable guidance, benevolent attitude and constant encouragement.

We reciprocate the kindness shown to us by the staff members of our college, people at home and our beloved friends who have contributed in the form of ideas, constructive criticism and encouragements for the successful completion of the project.

ABSTRACT

As the learning through computer is on the rise, the demand for e-learning is gradually increasing. In the proposed a e-learning system, where there is various levels of administrators to manage the portal in an efficient way. Administrator who has the privilege of creating and providing rights to second level people known secretaries. In turn these secretaries can manage the faculties and the e-learners by using proper authentication mechanisms. Each secretary is empowered with specific tasks in managing the portal.

Salient features of our e-learning portal are online test of objective type / multiple choices along with the report regarding the performance, proficiency, weakness of the e-learners. Courseware materials can be uploaded directly by the enrolled faculties. The material may be in any file format. The e-learner, once registered for a specific course can browse through the materials available and he can take up the online test.

The portal was implemented by using the following technologies

GUI design using Java

Validation by Java Script

Server side scripting by JSP

Server used to host the portal is Tomcat

Database used is MySQL

CONTENTS

1. Introduction	1
2. Analysis of problem	3
2.1 Problem Definition	4
2.2 System Analysis	4
2.2.1 Existing System	5
2.2.2 Drawbacks of Existing System	6
2.2.3 Proposed System	6
3. Design and Implementation	7
3.1 Dataflow Diagram	8
3.2 Structural Design	9
3.2.1 Class Diagram	9
3.2.2 Sequence Diagram	10
3.2.3 Collaboration Diagram	12
3.3 Requirement Specification	14
3.3.1 Hardware Requirements	14
3.3.2 Software Requirements	14
3.3.3 Software Used	14
3.4 E-Learning Platform	17

4. Testing	20
4.1 Unit Testing	21
4.2 Integration Testing	21
4.2.1 Bottom up Testing	22
4.2.2 Top Down Testing	22
4.2.3 Sandwich Testing	22
4.3 System Testing	22
4.4 Regression Testing	23
4.5 Alpha Testing	24
4.6 Beta Testing	25
5. Sample Code	26
6. Snap shots	58
7. Conclusion & Future Work	78
7.1 Conclusion	79
7.2 Future Work	79
References	80

CHAPTER-1

INTRODUCTION

As the learning through computer is on the rise, the demand of e-learning is gradually increasing. In our system there is various levels of administrators to manage the portal in an efficient way. . Administrator who has the privilege of creating and providing rights to second level people known secretaries. In turn these secretaries can manage the faculties and the e-learners by using proper authentication mechanisms. Each secretary is empowered with specific tasks in managing the portal.

Salient features of our e-learning portal are online test of objective type / multiple choices along with the report regarding the performance, proficiency, weakness of the e-learners. Courseware materials can be uploaded directly by the enrolled faculties. The material may be in any file format. The e-learner, once registered for a specific course can browse through the materials available and he can take up the online test.

In the online test, the tests are conducted in chapter wise for the learners. One who finish the first chapter with required marks can go to the second chapter. Else he/she should go on with the same chapter until finishing it with required marks.

Selecting the chapter that is displayed to learner that needs more study is accomplished according with learner's performed activity. The activity is represented by the number of taken tests, the average result of the tests and the final result at the discipline.

CHAPTER-2

ANALYSIS OF THE PROBLEM

2.1 Problem Definition:

In existing online education systems, there are several drawbacks are there. The materials and exams will not be according to their courses. They will be available as a common. And the admin is the only one who has the authority to make any changes in the system. The learners can't find out what is their knowledge level before the exams as it is a live one.

In our system, there is an analysis procedure that has as input data representing the performed activities by learners. We will predict the resources that the learner needs to access and study for improving his proficiency regarding the studied subject. Selecting the chapter that is displayed to learner that needs more study is accomplished according with learner's performed activity.

2.2 System Analysis:

JAVA - In the Java programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes — the machine language of the Java Virtual Machine¹ (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.

MYSQL – MySQL is a relational database management

of databases. Many web applications use MySQL as the database component of a LAMP software stack. Its popularity for use with web applications is closely tied to the popularity of PHP, which is often combined with MySQL. Several high-traffic web sites (including Flickr, Facebook, Wikipedia, Google (though not for searches), Nokia and YouTube) use MySQL for data storage and logging of user data.

APACHE TOMCAT- Apache Tomcat (or Jakarta Tomcat or simply Tomcat) is an open source servlet container developed by the Apache Software Foundation (ASF). Tomcat implements the Java Servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, and provides a “pure Java” HTTP web server environment for Java code to run.

This is the top-level entry point of the documentation bundle for the Apache Tomcat Servlet/JSP container. Apache Tomcat version 5.5 implements the Servlet 2.4 and JavaServer Pages 2.0 specifications from the Java Community Process, and includes many additional features that make it a useful platform for developing and deploying web applications and web services.

2.2.1 Existing System:

- In existing online education, materials and exams will be available as a common.
- Authorizations are preferable only for admin so workload will be more for him.
- This was a live and active exam, not a practice test.

CHAPTER-3

DESIGN AND IMPLEMENTATION

3.1 DATA FLOW DIAGRAM

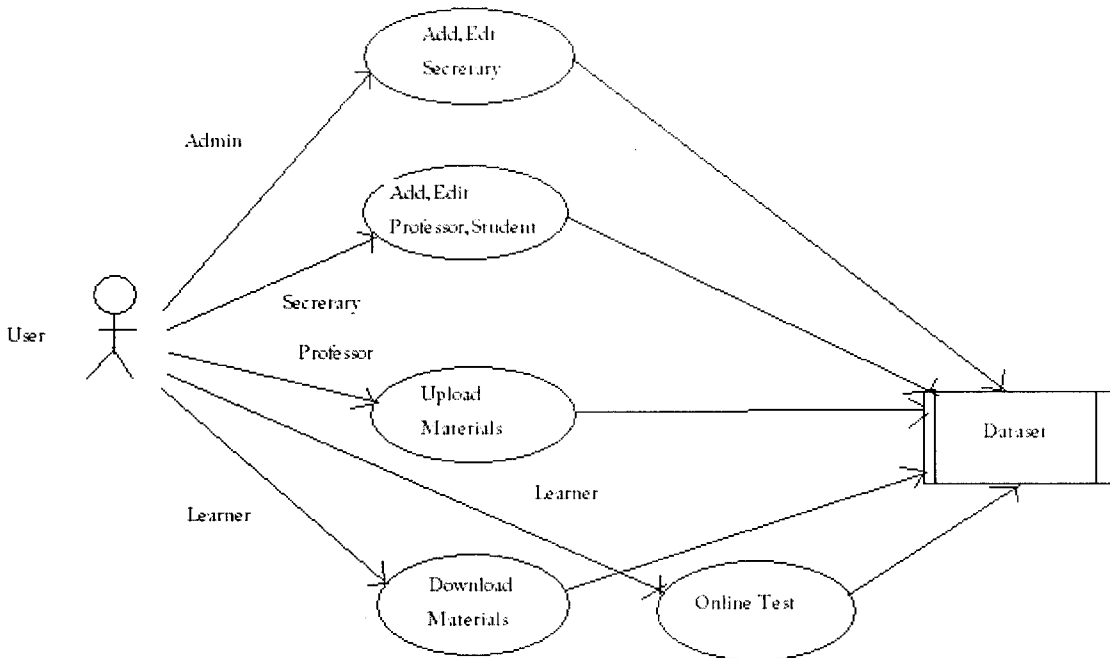


FIGURE 1 - E-LEARNING PROCESS DATA FLOW DIAGRAM

3.2 STRUCTURAL DESIGN

3.2.1 CLASS DIAGRAM

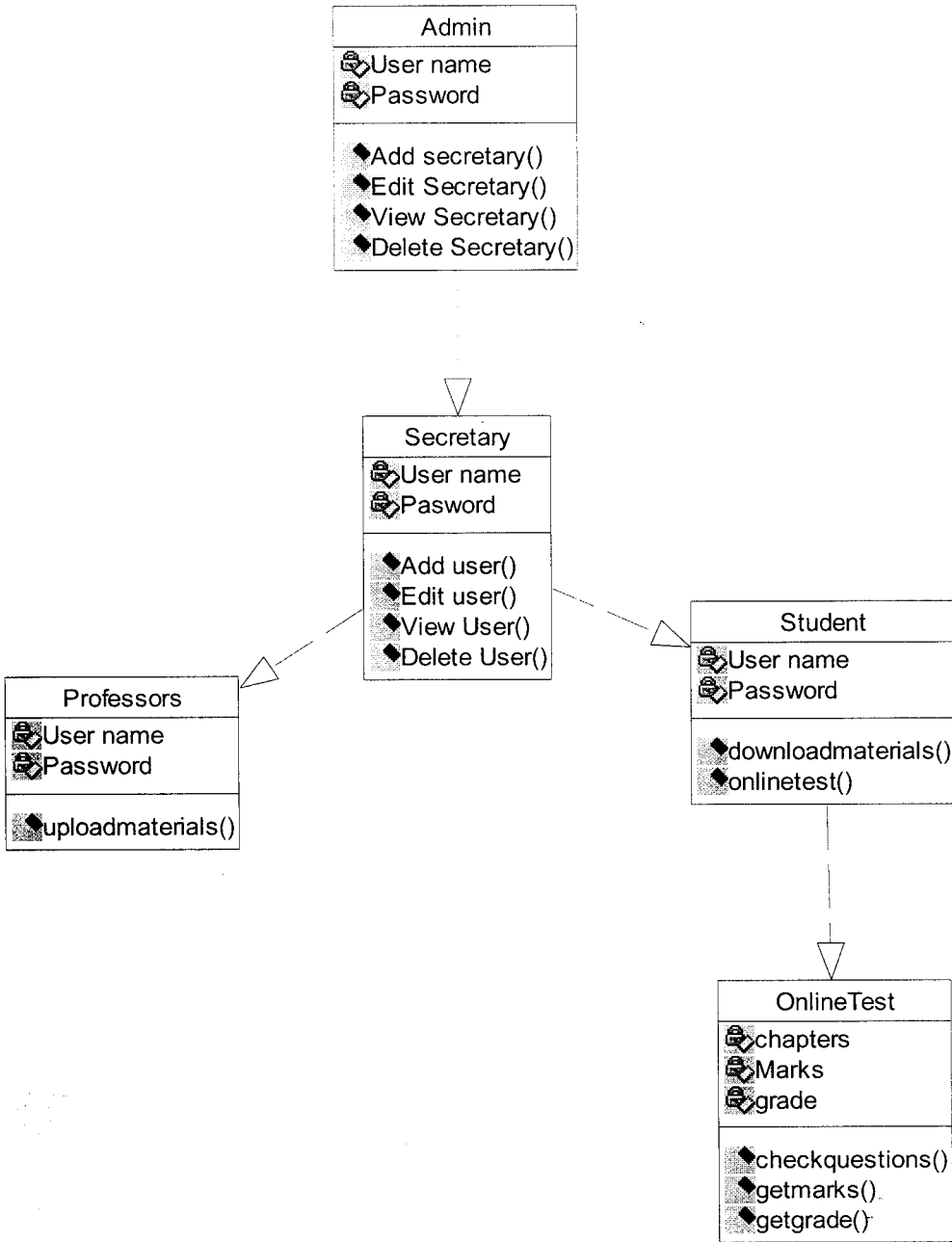


FIGURE 2 – E-LEARNING PROCESS CLASS DIAGRAM

3.2.2 SEQUENCE DIAGRAM

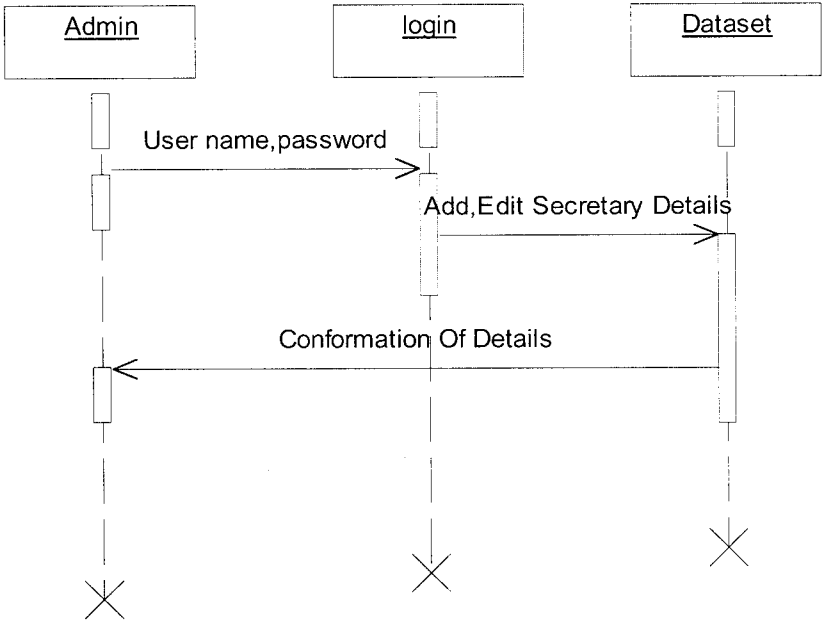


FIGURE 3 - ADMIN PROCESS

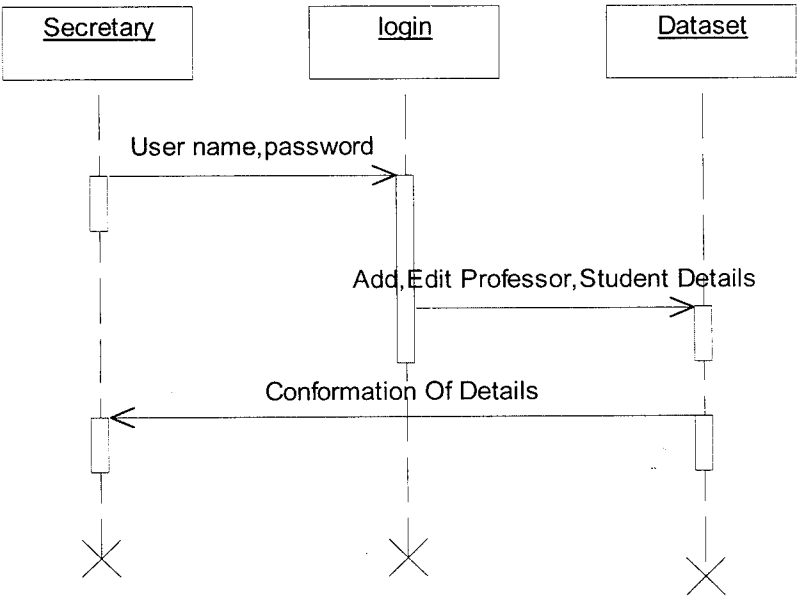


FIGURE 4 - SECRETARY PROCESS

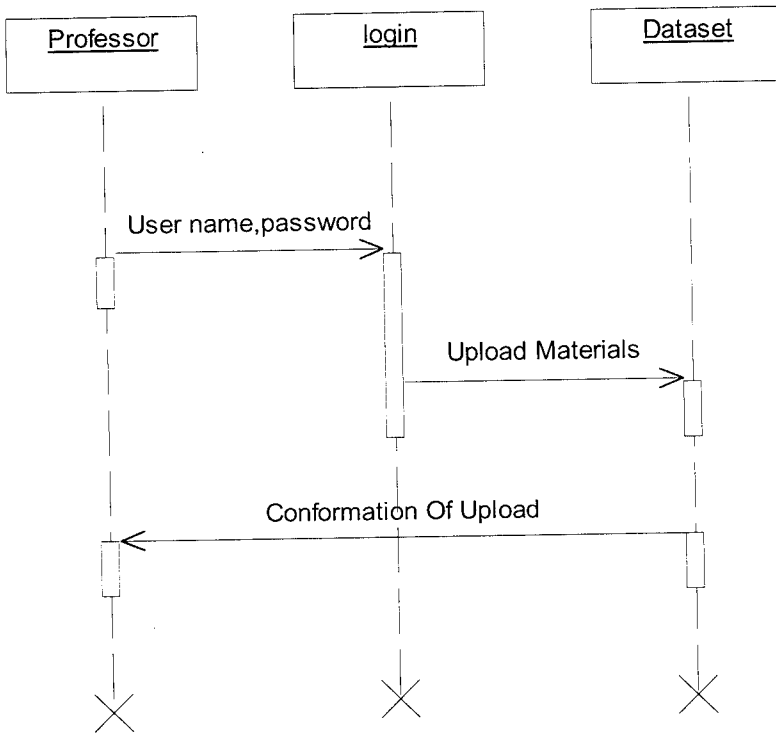


FIGURE 5 - PROFESSOR PROCESS



P-3118

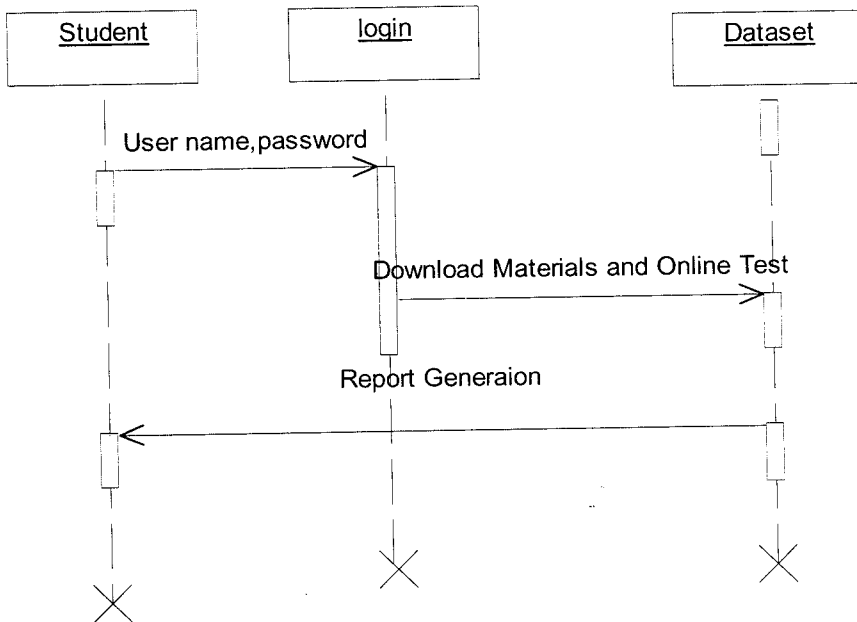


FIGURE 6 - LEARNER PROCESS

3.2.3 COLLABORATION DIAGRAM

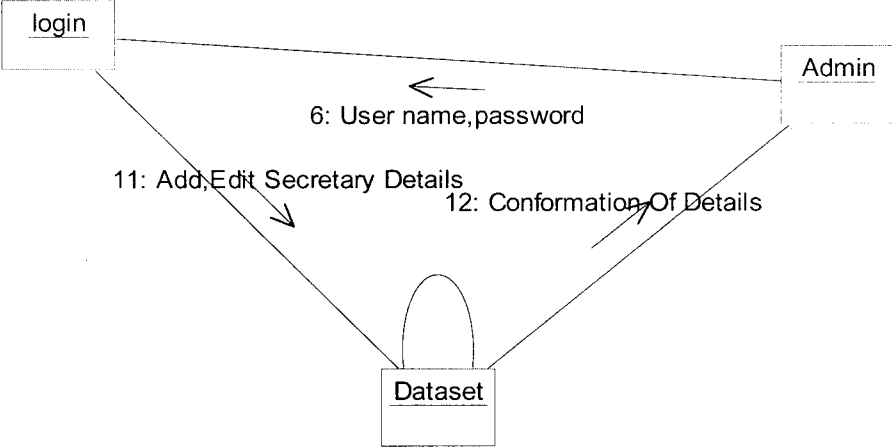


FIGURE 7 - ADMIN PROCESS

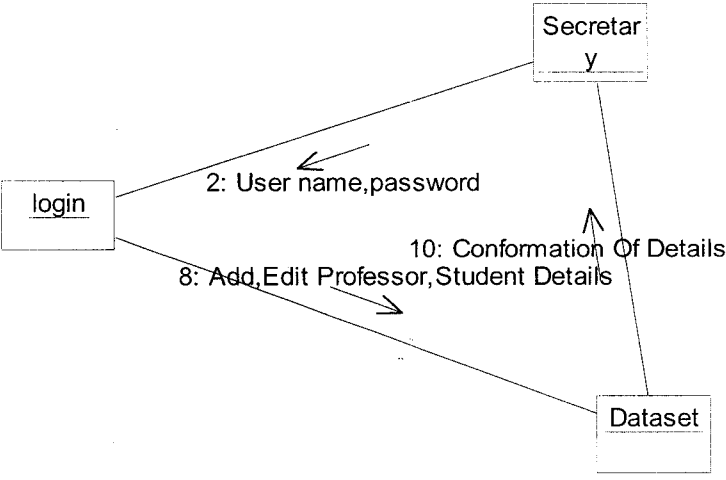


FIGURE 8 - SECRETARY PROCESS

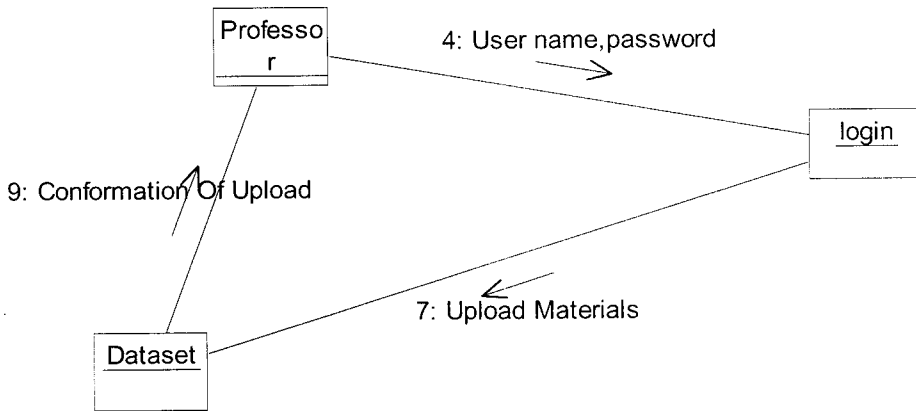


FIGURE 9 - PROFESSOR PROCESS

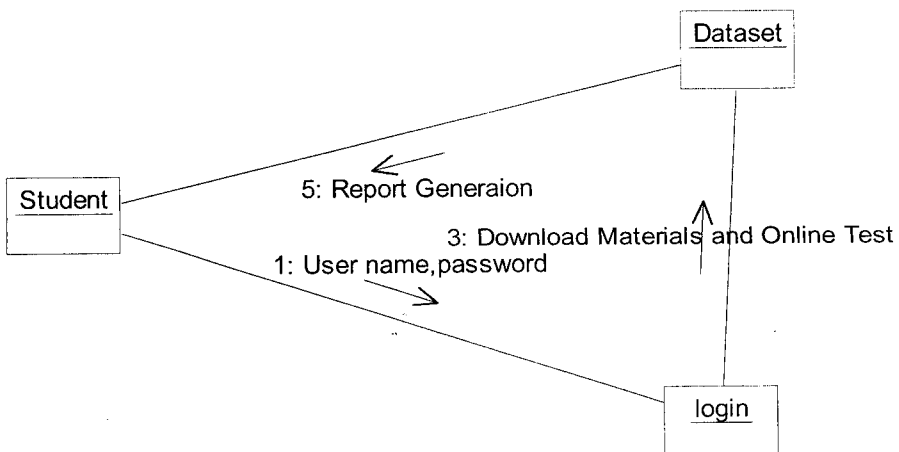


FIGURE 10 - LEARNER PROCESS

3.3.Requirement Specification

3.3.1 Hardware Requirement

- Processor :Intel Pentium or Higher
- RAM :128 MB or Higher
- Hard Disk :40 GB or Higher
- Monitor :Monochrome colour Monitor
- Keyboard :104 keys
- Video :16 bit high colour

3.3.2 Software Requirement

- Language : JAVA
- Web Components : Servlet ,JSP
- Web Server : Tomcat
- Database : MySQL

3.3.3 Software used to design module

- Java
- MySql

3.3.3.1. Java

In the Java programming language, all source code is first written in plain text files ending with the `.java` extension. Those source files are then compiled into `.class` files by the `javac` compiler. A `.class` file does not contain code that is native to your processor; it instead contains *bytecodes* — the machine language of the Java Virtual Machine¹ (Java VM). The `java`

launcher tool then runs your application with an instance of the Java Virtual Machine.

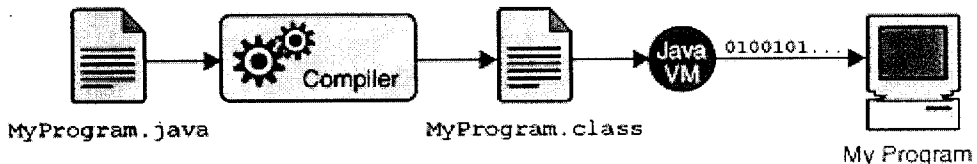


Figure 11 – An overview of the software development process.

Because the Java VM is available on many different operating systems, the same .class files are capable of running on Microsoft Windows, the Solaris™ Operating System (Solaris OS), Linux, or Mac OS. Some virtual machines, such as the Java HotSpot virtual machine, perform additional steps at runtime to give your application a performance boost. This include various tasks such as finding performance bottlenecks and recompiling (to native code) frequently used sections of code.

The Java Platform

A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Microsoft Windows, Linux, Solaris OS, and Mac OS. Most platforms can be described as a combination of the operating system and underlying hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The Java Virtual Machine
- The Java Application Programming Interface (API)

You've already been introduced to the Java Virtual Machine; it's the base for the Java platform and is ported onto various hardware-based platforms.

The API is a large collection of ready-made software components that provide many useful capabilities. It is grouped into libraries of related classes and interfaces; these libraries are known as packages. The next section, highlights some of the functionality provided by the API.

Platform independence

One characteristic, platform independence, means that programs written in the Java language must run similarly on any supported hardware/operating-system platform. One should be able to write a program once, compile it once, and run it anywhere.

This is achieved by most Java compilers by compiling the Java language code halfway (to Java bytecode) – simplified machine instructions specific to the Java platform. The code is then run on a virtual machine (VM), a program written in native code on the host hardware that interprets and executes generic Java bytecode. (In some JVM versions, bytecode can also be compiled to native code, either before or during program execution, resulting in faster execution.) Further, standardized libraries are provided to allow access to features of the host machines (such as graphics, threading and networking) in unified ways. Note that, although there is an explicit compiling stage, at some point, the Java bytecode is interpreted or converted to native machine code by the JIT compiler.

The first implementations of the language used an interpreted virtual machine to achieve portability. These implementations produced programs that ran more slowly than programs compiled to native executables, for instance

More recent JVM implementations produce programs that run significantly faster than before, using multiple techniques.

3.3.3.2 MySQL

- A data base is created in My SQL.
- A table with name login is created.
- The login table has two fields they are username and password.
- Both the fields are described as not null.
- The user name and password are predefined by the administrator.

3.4. E-Learning Platform

The main goal of the platform is to give students the possibility to download course materials, take tests or sustain final examinations and communicate with all involved parties. To accomplish this, four different roles were defined for the platform: sysadmin, secretary, professor and student. The main task of sysadmin users is to manage secretaries. A sysadmin user may add or delete secretaries, or change their password. He may also view the actions performed by all other users of the platform. All actions performed by users are logged. In this way the sysadmin may check the activity that takes place on the application. The logging facility has some benefits. An audit may be performed for the application with the logs as witness. Security breaches may also be discovered. Secretary users manage sections, professors, disciplines and students. On any of these a secretary may perform actions like add, delete or update.

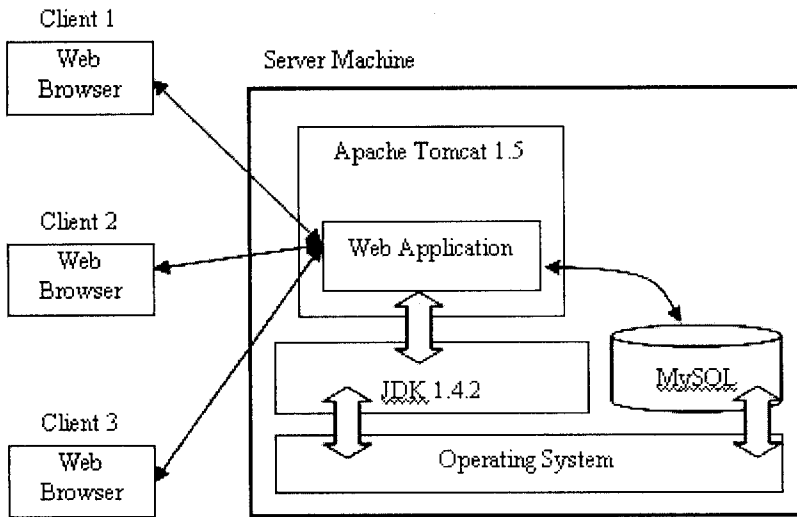


Figure 12 –Software architecture of the platform

These actions will finally set up the application such that professors and students may use it. As conclusion, the secretary manages a list of sections, a list of professors and a list of students. Each discipline is assigned to a section and has as attributes a name, a short name, the year of study and semester when it is studied and the list of professors that teach the discipline which may be maximum three. A student may be enrolled to one or more sections. The main task of a professor is to manage the assigned disciplines while s discipline is made up of chapters. The professor sets up chapters by specifying the name and the course document. Only students enrolled in a section in which a discipline is studied may download the course document and take tests or examinations. Besides setting up the course document for each chapter, the professor manages test and exam questions. For each chapter the professor has to define two pools of questions, one used for testing and one used for exams. He specifies the number of questions that will be randomly extracted to create a test or an exam.

This application offers students the possibility to download course materials, take tests and exams and communicate with other involved parties like professors and secretaries. Students may download only course materials for the disciplines that belong to sections where they are enrolled. They can take tests and exams with constraints that were set up by the secretary through the year structure facility.

CHAPTER-4

TESTING

It is the process used to help identify the correctness, completeness, security, and quality of developed computer software. Testing is a process of technical investigation, performed on behalf of stakeholders, that is intended to reveal quality-related information about the product with respect to the context in which it is intended to operate. This includes, but is not limited to, the process of executing a program or application with the intent of finding errors.

4.1 UNIT TESTING:

The developer carries out unit testing in order to check if the particular module or unit of code is working fine. The Unit Testing comes at the very basic level as it is carried out as and when the unit of the code is developed or a particular functionality is built. Unit testing deals with testing a unit as a whole. This would test the interaction of many functions but confine the test within one unit. The exact scope of a unit is left to interpretation.

4.2 INTEGRATION TESTING:

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before system testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated

4.2.1 Bottom Up Testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher level components. The process is repeated until the component at the top of the hierarchy is tested.

All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower level integrated modules, the next level of modules will be formed and can be used for integration testing. This approach is helpful only when all or most of the modules of the same development level are ready. This method also helps to determine the levels of software developed and makes it easier to report testing progress in the form of a percentage.

4.2.2 Top Down Testing is an approach to integrated testing where the top integrated modules are tested and the branch of the module is tested step by step until the end of the related module.

4.2.3 Sandwich Testing is an approach to combine top down testing with bottom up testing.

4.3 SYSTEM TESTING

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. ^[1]

As a rule, system testing takes, as its input, all of the "integrated" software components that have successfully passed integration testing and also the software system itself integrated with any applicable hardware system(s). The

purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called *assemblages*) or between any of the *assemblages* and the hardware. System testing is a more limiting type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

4.4 REGRESSION TESTING

Regression testing is a style of testing that focuses on retesting after changes are made. In traditional regression testing, we reuse the same tests (the regression tests). In risk-oriented regression testing, we test the same areas as before, but we use different (increasingly complex) tests. Traditional regression tests are often partially automated. These notes focus on traditional regression.

Regression testing attempts to mitigate two risks:

- A change that was intended to fix a bug failed.
- Some change had a side effect, unfixing an old bug or introducing a new bug

Regression testing approaches differ in their focus. Common examples include:

Bug regression: We retest a specific bug that has been allegedly fixed.

Old fix regression testing: We retest several old bugs that were fixed, to see if they are back. (This is the classical notion of regression: the program has regressed to a bad state.)

General functional regression: We retest the product broadly, including areas that worked before, to see whether more recent changes have destabilized

Conversion or port testing: The program is ported to a new platform and a subset of the regression test suite is run to determine whether the port was successful. (Here, the main changes of interest might be in the new platform, rather than the modified old code.)

Configuration testing: The program is run with a new device or on a new version of the operating system or in conjunction with a new application. This is like port testing except that the underlying code hasn't been changed--only the external components that the software under test must interact with.

Localization testing: The program is modified to present its user interface in a different language and/or following a different set of cultural rules. Localization testing may involve several old tests (some of which have been modified to take into account the new language) along with several new (non-regression) tests.

Smoke testing also known as *build verification testing*: A relatively small suite of tests is used to qualify a new build. Normally, the tester is asking whether any components are so obviously or badly broken that the build is not worth testing or some components are broken in obvious ways that suggest a corrupt build or some critical fixes that are the primary intent of the new build didn't work. The typical result of a failed smoke test is rejection of the build (testing of the build stops) not just a new set of bug reports.

4.5 ALPHA TESTING

In this type of testing, the users are invited at the development centre where they use the application and the developers note every particular input or action carried out by the user. Any type of abnormal behaviour of the system is noted and rectified by the developers.

4.6 BETA TESTING

In this type of testing, the software is distributed as a beta version to the users and users test the application at their sites. As the users explore the software, in case if any exception/defect occurs that is reported to the developers. Beta testing comes after alpha testing. Versions of the software, known as beta versions, are released to a limited audience outside of the company. The software is released to groups of people so that further testing can ensure the product has few faults or bugs. Sometimes, beta versions are made available to the open public to increase the feedback field to a maximal number of future users.

CHAPTER-5

SAMPLE CODE

ADDING USERS:

```
package learner.employee.servlet;

import learner.employee.bean.Database;
import learner.employee.servlet.*;

import java.io.*;
import java.util.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.Connection.*;
import java.sql.ResultSet.*;

public class Add extends HttpServlet
{
    HttpSession session;
    PrintWriter out;
    Database db = null;
    ResultSet rs = null;
    String id;
    String username;
    String password;
    String outString;

    public void doGet(HttpServletRequest request, HttpServletResponse response)
    {
        doPost(request, response);
    }

    public void doPost(HttpServletRequest request, HttpServletResponse
response)
    {
        session=request.getSession(true);
        try
        {
            id = (String) session.getAttribute("id");
```

```

System.out.println("Add[id] : "+id);
    db = new Database();
    if(id.startsWith("A") && id.equals("A"))
    {
        username=request.getParameter("addadmin_username");
        password=request.getParameter("addadmin_password");
System.out.println("Add[admin] => username : "+username);
System.out.println("Add[admin] => password : "+password);
        final String query = "insert into secretary (username,password)
values(""+username+"", ""+password+"")";
        int result = db.executeUpdate(query);
        response.setContentType("text/xml");
        response.setHeader("Cache-Control", "no-cache");
        outString = "addadmin_login";
        response.getWriter().write(outString);
    }
    else if(id.startsWith("S") && id.equals("S"))
    {
        String query="";
        int result;
        id = request.getParameter("tableid");
        username=request.getParameter("addsecretary_username");
        password=request.getParameter("addsecretary_password");
        if(id.equals("Professor")) {
            query = "insert into professor(username,password)
values(""+username+"", ""+password+"")";
            result = db.executeUpdate(query);
            outString = "addsecretary_login";
        } else if(id.equals("Student")) {
            query = "insert into student(username,password)
values(""+username+"", ""+password+"")";
            result = db.executeUpdate(query);
            outString = "addsecretary_login";
        }
        response.setContentType("text/xml");
        response.setHeader("Cache-Control", "no-cache");
        response.getWriter().write(outString);
    }
    else if(id.startsWith("P") && id.equals("P"))
    {
System.out.println("professor_add");

```

```

        else if(id.startsWith("Z") && id.equals("Z"))
        {
System.out.println("Student_add");
        }
        else
        {
            System.out.println("Add Error");
        }
    } catch(Exception e) {
        System.out.println("Exception : "+e.toString());
    }
}
}
}

```

DELETING USERS:

```

package learner.employee.servlet;

import learner.employee.bean.Database;
import learner.employee.servlet.*;

import java.io.*;
import java.util.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.Connection.*;
import java.sql.ResultSet.*;

public class Delete extends HttpServlet
{
    HttpSession session;
    PrintWriter out;
    Database db = null;
    ResultSet rs = null;
    String id;
    String did;
    int del_id;
    String query="";
    String outString;

```

```

public void doGet(HttpServletRequest request, HttpServletResponse response)
{
    doPost(request, response);
}

```

```

public void doPost(HttpServletRequest request, HttpServletResponse
response)
{
    session=request.getSession(true);
    try
    {
        id = (String)session.getAttribute("roleid");
        System.out.println("delete[id] : "+id);
        db = new Database();
        if(id.startsWith("A") && id.equals("A"))
        {
            did=request.getParameter("id");
            del_id = Integer.parseInt(did);
            query = "delete from secretary where id = "+del_id;
            int result = db.executeUpdate(query);
            response.setContentType("text/xml");
            response.setHeader("Cache-Control", "no-cache");
            String outString = "deleteadmin_login";
            response.getWriter().write(outString);
        }
        else if(id.startsWith("S") && id.equals("S"))
        {

        }
        else if(id.startsWith("P") && id.equals("P"))
        {

        }
        else if(id.startsWith("Z") && id.equals("Z"))
        {

        }
        else
        {
            System.out.println("Add Error");
        }
    }
}

```

```
System.out.println("Exception : "+e.toString());
```

```
}
```

```
}
```

```
}
```

EDITING USERS:

```
package learner.employee.servlet;
```

```
import learner.employee.bean.Database;
```

```
import learner.employee.servlet.*;
```

```
import java.io.*;
```

```
import javax.servlet.*;
```

```
import javax.servlet.http.*;
```

```
import java.sql.*;
```

```
import java.sql.Connection.*;
```

```
import java.sql.ResultSet.*;
```

```
public class Edit extends HttpServlet
```

```
{
```

```
    HttpSession session;
```

```
    PrintWriter out;
```

```
    Database db=null;
```

```
    ResultSet rs=null;
```

```
    String id;
```

```
    int eid;
```

```
    String username;
```

```
    String password;
```

```
    String outString;
```

```
    String content;
```

```
    String query="";
```

```
public void doGet(HttpServletRequest request, HttpServletResponse response)
```

```
{
```

```
    doPost(request, response);
```

```
}
```

```
public void doPost(HttpServletRequest request, HttpServletResponse response)
```

```

{
    session=request.getSession(true);
    try
    {
        id = (String)session.getAttribute("roleid");
        System.out.println("Edit[id] : "+id);
        content = request.getParameter("content");
        db = new Database();
        if(id.startsWith("A") && id.equals("A"))
        {
            if(content.equals("name"))
            {
                eid = Integer.parseInt(request.getParameter("eid"));
                username = request.getParameter("editadmin_username");
                query = "update secretary set username = '"+username+"' where id
= "+eid;
            } else if(content.equals("pass")) {
                eid = Integer.parseInt(request.getParameter("eid"));
                password = request.getParameter("editadmin_password");
                query = "update secretary set password = '"+password+"' where id
= "+eid;
            }
            int result = db.executeUpdate(query);
            response.setContentType("text/xml");
            response.setHeader("Cache-Control", "no-cache");
            String outString = "editadmin_login";
            response.getWriter().write(outString);
        }
        else if(id.startsWith("S") && id.equals("S"))
        {
            String table="";
            if(content.equals("name"))
            {
                eid = Integer.parseInt(request.getParameter("eid"));
                username = request.getParameter("editsecprof_username");
                table = request.getParameter("table");
                System.out.println("Table = "+table);
                query = "update "+table+" set username = '"+username+"' where id
= "+eid;
            } else if(content.equals("pass")) {
                eid = Integer.parseInt(request.getParameter("eid"));

```

```

        table = request.getParameter("table");
System.out.println("Table = "+table);
        query = "update "+table+" set password = '"+password+"' where id
= "+eid;
    }
    int result = db.executeUpdate(query);
    response.setContentType("text/xml");
    response.setHeader("Cache-Control", "no-cache");
    String outString = "editsecretary_login";
    response.getWriter().write(outString);
}
else
{
    System.out.println("Edit Error");
}
} catch(Exception e) {
    System.out.println("Exception : "+e.toString());
}
}
}
}

```

VIEWING USERS:

```

package learner.employee.servlet;

import learner.employee.bean.Database;
import learner.employee.servlet.*;

import java.io.*;
import java.util.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.Connection.*;
import java.sql.ResultSet.*;

```

```

public class View extends HttpServlet

```

```

{
    HttpSession session;
    PrintWriter out;
    Database db = null;

```

```
ResultSet rs = null;
String id;
String username;
String password;
String outString;
ArrayList usernamelist;
ArrayList passwordlist;
```

```
public void doGet(HttpServletRequest request, HttpServletResponse response)
{
    doPost(request, response);
}
```

```
public void doPost(HttpServletRequest request, HttpServletResponse
response)
```

```
{
    session=request.getSession(true);
    try
    {
        id = (String)session.getAttribute("roleid");
        System.out.println("view[id] : "+id);
        usernamelist = new ArrayList();
        passwordlist = new ArrayList();
        db = new Database();
        if(id.startsWith("A") && id.equals("A"))
        {
            String query = "select * from secretary";
            rs = db.executeQuery(query);
            usernamelist.clear();
            passwordlist.clear();
            while(rs.next())
            {
                username = rs.getString(2);
                usernamelist.add(username);
                password = rs.getString(3);
                passwordlist.add(password);
            }
            session.setAttribute("usernamelist",usernamelist);
            session.setAttribute("passwordlist",passwordlist);
            response.setContentType("text/xml");
        }
    }
}
```



```

    outString = "viewadmin_login";
    response.getWriter().write(outString);
}
else if(id.startsWith("S") && id.equals("S"))
{
    String radio = request.getParameter("radio");
    String query="";
    if(radio.equals("Professor")) {
        query = "select * from professor";
        rs = db.executeQuery(query);
        outString = "viewsecretary_professor_login";
    } else if(radio.equals("Student")) {
        query = "select * from student";
        rs = db.executeQuery(query);
        outString = "viewsecretary_student_login";
    }
    usernamelist.clear();
    passwordlist.clear();
    while(rs.next())
    {
        username = rs.getString(2);
        usernamelist.add(username);
        password = rs.getString(3);
        passwordlist.add(password);
    }
        session.setAttribute("usernamelist",usernamelist);
        session.setAttribute("passwordlist",passwordlist);
    response.setContentType("text/xml");
    response.setHeader("Cache-Control", "no-cache");
    response.getWriter().write(outString);
}
else if(id.startsWith("P") && id.equals("P"))
{

}
else if(id.startsWith("Z") && id.equals("Z"))
{

}
}
else
{

```

```

    }
} catch(Exception e) {
    System.out.println("Exception : "+e.toString());
}
}
}
}

```

ADMIN LOGIN:

```

package learner.employee.servlet;

import learner.employee.bean.Database;
import learner.employee.servlet.LoginAuthentication;

import java.io.*;
import java.util.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.Connection.*;
import java.sql.ResultSet.*;

public class AdminLogin extends HttpServlet
{
    Database db = null;
    ResultSet rs = null;
    String aid;
    String username;
    String password;
    String outString;
    RequestDispatcher rd;
    HttpSession session;
    PrintWriter out;

    public void doGet(HttpServletRequest request,HttpServletResponse
response) throws IOException, ServletException
    {
        doPost(request, response);
    }
}

```

```

public void doPost(HttpServletRequest request,HttpServletResponse
response) throws IOException, ServletException
{
    session=request.getSession(true);
    try {
        aid = (String)session.getAttribute("roleid");
        session.setAttribute("aid",aid);
        System.out.println("AdminLogin : "+aid);
        username = (String)session.getAttribute("username");
        password = (String)session.getAttribute("password");

        db = new Database();
        final String query = "select * from admin where username =
""+username+"" and password = ""+password+""";
        rs = db.executeQuery(query);
        response.setContentType("text/xml");
        response.setHeader("Cache-Control", "no-cache");
        if(rs.next()) {
            outString = "success1";
            response.getWriter().write(outString);
        }else {
            outString = "failure";
            response.getWriter().write(outString);
        }
    } catch(Exception e) {
        System.out.println("Exception : "+e);
    }
}
}
}
}

```

STUDENT LOGIN:

```

package learner.employee.servlet;

import learner.employee.bean.Database;
import learner.employee.servlet.LoginAuthentication;

import java.io.*;
import java.util.*;
import java.sql.*;
import javax.servlet.*;

```

```
import javax.servlet.http.*;
import java.sql.Connection.*;
import java.sql.ResultSet.*;
```

```
public class StudentLogin extends HttpServlet
{
```

```
    Database db = null;
    ResultSet rs = null;
    String stdid;
    String username;
    String password;
    String outString;
    RequestDispatcher rd;
    HttpSession session;
    PrintWriter out;
```

```
    public void doGet(HttpServletRequest request,HttpServletResponse
response) throws IOException, ServletException
    {
        doPost(request, response);
    }
```

```
    public void doPost(HttpServletRequest request,HttpServletResponse
response) throws IOException, ServletException
    {
```

```
        session=request.getSession(true);
        try {
            stdid = (String)session.getAttribute("roleid");
            System.out.println("StudentLogin : "+stdid);
            username = (String)session.getAttribute("username");
            password = (String)session.getAttribute("password");
```

```
            db = new Database();
            final String query = "select * from student where username =
"+"username+" and password = "+"password+"";
            rs = db.executeQuery(query);
            response.setContentType("text/xml");
            response.setHeader("Cache-Control", "no-cache");
            if(rs.next()) {
                outString = "success4";
                response.getWriter().write(outString);
```

```

        outString = "failure";
        response.getWriter().write(outString);
    }
} catch(Exception e) {
    System.out.println("Exception : "+e);
}
}
}
}

```

SECRETARY LOGIN:

```
package learner.employee.servlet;
```

```
import learner.employee.bean.Database;
import learner.employee.servlet.LoginAuthentication;
```

```
import java.io.*;
import java.util.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.Connection.*;
import java.sql.ResultSet.*;
```

```
public class SecretaryLogin extends HttpServlet
{
```

```
    Database db = null;
    ResultSet rs = null;
    String sid;
    String username;
    String password;
    String outString;
    RequestDispatcher rd;
    HttpSession session;
    PrintWriter out;
```

```
    public void doGet(HttpServletRequest request,HttpServletResponse
response) throws IOException, ServletException
```

```
    {
        doPost(request, response);
    }
}

```

```

public void doPost(HttpServletRequest request,HttpServletResponse
response) throws IOException, ServletException
{
    session=request.getSession(true);
    try {
        sid = (String)session.getAttribute("roleid");
        System.out.println("SecretaryLogin : "+sid);
        username = (String)session.getAttribute("username");
        password = (String)session.getAttribute("password");

        db = new Database();
        final String query = "select * from secretary where username =
""+username+"" and password = ""+password+""";
        rs = db.executeQuery(query);
        response.setContentType("text/xml");
        response.setHeader("Cache-Control", "no-cache");
        if(rs.next()) {
            outString = "success3";
            response.getWriter().write(outString);
        }else {
            outString = "failure";
            response.getWriter().write(outString);
        }
    } catch(Exception e) {
        System.out.println("Exception : "+e);
    }
}
}
}

```

PROFESSOR LOGIN:

```

package learner.employee.servlet;

import learner.employee.bean.Database;
import learner.employee.servlet.LoginAuthentication;

import java.io.*;
import java.util.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;

```

```
import java.sql.ResultSet.*;
```

```
public class ProfessorLogin extends HttpServlet  
{
```

```
    Database db = null;  
    ResultSet rs = null;  
    String pid;  
    String username;  
    String password;  
    String outString;  
    RequestDispatcher rd;  
    HttpSession session;  
    PrintWriter out;
```

```
    public void doGet(HttpServletRequest request,HttpServletResponse  
response) throws IOException, ServletException  
    {  
        doPost(request, response);  
    }
```

```
    public void doPost(HttpServletRequest request,HttpServletResponse  
response) throws IOException, ServletException  
    {  
        session=request.getSession(true);  
        try {  
            pid = (String)session.getAttribute("roleid");  
            session.setAttribute("pid",pid);  
            System.out.println("ProfessorLogin : "+pid);  
            username = (String)session.getAttribute("username");  
            password = (String)session.getAttribute("password");  
  
            db = new Database();  
            final String query = "select * from professor where username =  
"+username+" and password = "+password+"";  
            rs = db.executeQuery(query);  
            response.setContentType("text/xml");  
            response.setHeader("Cache-Control", "no-cache");  
            if(rs.next()) {  
                outString = "success2";  
                response.getWriter().write(outString);  
            }else {
```

```

        response.getWriter().write(outString);
    }
} catch(Exception e) {
    System.out.println("Exception : "+e);
}
}
}
}

```

LOGIN AUTHENTICATION:

```
package learner.employee.servlet;
```

```
import java.io.*;
import java.util.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
```

```
public class LoginAuthentication extends HttpServlet
```

```
{
    String roleid;
    String username;
    String password;
    HttpSession session;
    RequestDispatcher rd;
```

```
    public void doGet(HttpServletRequest request,HttpServletResponse
response) throws IOException, ServletException
```

```
{
    doPost(request, response);
}

```

```
    public void doPost(HttpServletRequest request,HttpServletResponse
response) throws IOException, ServletException
```

```
{
    try
    {
        session = request.getSession();
        roleid = request.getParameter("roleid");
        username=request.getParameter("username");
        password = request.getParameter("password");
    }
}
}
}

```



```

session.setAttribute("roleid",roleid);
session.setAttribute("username",username);
session.setAttribute("password",password);

//session.setAttribute("id",request.getParameter("roleid"));
//id=(String)session.getAttribute("id");

//session.setAttribute("username",request.getParameter("username"));
//username = (String)session.getAttribute("username");

//session.setAttribute("password",request.getParameter("password"));
//password = (String)session.getAttribute("password");

if(roleid.startsWith("A"))
{
    rd=request.getRequestDispatcher("AdminLogin");
}
else if(roleid.startsWith("S"))
{
    rd=request.getRequestDispatcher("SecretaryLogin");
}
else if(roleid.startsWith("P"))
{
    rd=request.getRequestDispatcher("ProfessorLogin");
}
else if(roleid.startsWith("Z"))
{
    rd=request.getRequestDispatcher("StudentLogin");
}
rd.forward(request,response);
}
catch (Exception e2)
{
    System.out.println("Exception : "+e2.toString());
}
}
}

```

UPLOADING FILE:

```

package learner.employee.servlet;

import learner.employee.bean.Database;
import learner.employee.servlet.*;

import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;

import org.apache.commons.fileupload.DiskFileUpload;
import org.apache.commons.fileupload.FileItem;
import org.apache.commons.fileupload.FileItemFactory;
import org.apache.commons.fileupload.FileUpload;
import org.apache.commons.fileupload.FileUploadException;

public class UploadFile extends HttpServlet
{
    HttpSession session;
    String id,title="";
    String outString,department;
    PrintWriter out;
    Connection con;
    Statement st;
    ResultSet rs;

    public void doPost(HttpServletRequest request,HttpServletResponse
response)throws IOException, ServletException
    {
        session=request.getSession(true);
        out = response.getWriter();
        department = (String)session.getAttribute("Department");
        try
        {
            //title=request.getParameter("title");
            // System.out.println("title"+title);
            Class.forName("com.mysql.jdbc.Driver");
            con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/student","root","pa
ssword");

```

```

FileUpload fup=new FileUpload();
45boolean isMultipart = FileUpload.isMultipartContent(request);
DiskFileUpload upload = new DiskFileUpload();
List items = upload.parseRequest(request);
Iterator iter = items.iterator();

```

```

while (iter.hasNext())

```

```

{
    FileItem item = (FileItem) iter.next();
    if (item.isFormField())
    {
        System.out.println("its a field");
    }
    else
    {
        File cfile1=new File(item.getName());
        //System.out.println("FileName : "+cfile1.getName());
        File cfile2=new File(item.getContentType());
        //System.out.println("FileType : "+item.getContentType());
        title=cfile1.getName();
        if(department.equals("CSE"))
        {

```

```

            File tosave=new

```

```

File(getServletContext().getRealPath("jsp/professor/CSE/"),cfile1.getName());

```

```

            item.write(tosave);

```

```

                String

```

```

str="jsp/professor/CSE/"+cfile1.getName();

```

```

                rs = st.executeQuery("select * from material

```

```

");

```

```

                int j = st.executeUpdate("insert into material

```

```

values(""+department+"", ""+str+"", ""+title+"");

```

```

            }

```

```

            else if(department.equals("IT"))

```

```

            {

```

```

                File tosave=new

```

```

File(getServletContext().getRealPath("jsp/professor/IT/"),cfile1.getName());

```

```

                String

```

```

str="jsp/professor/IT/"+cfile1.getName();

```

```

        rs = st.executeQuery("select * from material
");
        int j = st.executeUpdate("insert into material
values(""+department+"", ""+str+"", ""+title+"")");
    }
    else if(department.equals("ECE"))
    {
        File tosave=new
File(getServletContext().getRealPath("jsp/professor/ECE/"),cfile1.getName());
        String
str="jsp/professor/ECE/"+cfile1.getName();
        item.write(tosave);
        rs = st.executeQuery("select * from material
");
        int j = st.executeUpdate("insert into material
values(""+department+"", ""+str+"", ""+title+"")");
    }

    //System.out.println("FileSource : "+item.getName());
    //System.out.println("FileDestination : "+tosave);
}
out.println("Successfully Uploaded!..."); //for return Response
}
}
catch(Exception e)
{
    System.out.println("Exception : "+e.toString());
}
}
}
}
}

```

QUESTION ACTION:

```
package learner.employee.servlet;
```

```
import java.io.*;
import java.util.*;
import javax.servlet.*;
```

```

public class QuestionActionServlet extends HttpServlet
{
    HttpSession session;
    String result="";
    double pdfTALy, pdfTALn;
    double pdfDy, pdfDn;

    double
pdfCP1y, pdfCP1n, pdfCP2y, pdfCP2n, pdfCP3y, pdfCP3n, pdfCP4y, pdfCP4n;
    double
pdfnotCP1y, pdfnotCP1n, pdfnotCP2y, pdfnotCP2n, pdfnotCP3y, pdfnotCP3n, pdf
notCP4y, pdfnotCP4n;
    double
pdfavgCP1y, pdfavgCP1n, pdfavgCP2y, pdfavgCP2n, pdfavgCP3y, pdfavgCP3n, p
dfavgCP4y, pdfavgCP4n;
    double
pdffinalCP1y, pdffinalCP1n, pdffinalCP2y, pdffinalCP2n, pdffinalCP3y, pdffinalC
P3n, pdffinalCP4y, pdffinalCP4n;
    double
pdfYescp1, pdfNocp1, pdfYescp2, pdfNocp2, pdfYescp3, pdfNocp3, pdfYescp4, pd
fNocp4;

    public void doGet(HttpServletRequest request, HttpServletResponse response)
    {
        doPost(request, response);
    }

    public void doPost(HttpServletRequest request, HttpServletResponse
response)
    {
        try
        {
            session=request.getSession(true);
            Detection d2 = new Detection();
            Prediction p2 = new Prediction();
            Calculation c2 = new Calculation();

            String notch = (String)session.getAttribute("notch");
            String notch1 = (String)session.getAttribute("notch1");
            String notch2 = (String)session.getAttribute("notch2");

```

```
String notch4 = (String)session.getAttribute("notch4");
String ch1marks = (String)session.getAttribute("ch1marks");
String ch2marks = (String)session.getAttribute("ch2marks");
String ch3marks = (String)session.getAttribute("ch3marks");
String ch4marks = (String)session.getAttribute("ch4marks");
```

```
String ch1avgmarks = setAvg(ch1marks);
String ch2avgmarks = setAvg(ch2marks);
String ch3avgmarks = setAvg(ch3marks);
String ch4avgmarks = setAvg(ch4marks);
```

```
String ch1finalmark = findFinal(notch1,ch1avgmarks);
String ch2finalmark = findFinal(notch2,ch2avgmarks);
String ch3finalmark = findFinal(notch3,ch3avgmarks);
String ch4finalmark = findFinal(notch4,ch4avgmarks);
```

```
session.setAttribute("ch1avgmarks",ch1avgmarks);
```

```
session.setAttribute("ch2avgmarks",ch2avgmarks);
```

```
session.setAttribute("ch3avgmarks",ch3avgmarks);
```

```
session.setAttribute("ch4avgmarks",ch4avgmarks);
```

```
session.setAttribute("ch1finalmark",ch1finalmark);
```

```
session.setAttribute("ch2finalmark",ch2finalmark);
```

```
session.setAttribute("ch3finalmark",ch3finalmark);
```

```
session.setAttribute("ch4finalmark",ch4finalmark);
```

```
//session.setAttribute("ch1avgmarks",ch1avgmarks);
```

```
pdfCP1y = c2.setPdfchapterId10();
pdfCP1n = c2.setPdfchapterId11();
pdfCP2y = c2.setPdfchapterId20();
pdfCP2n = c2.setPdfchapterId21();
pdfCP3y = c2.setPdfchapterId30();
pdfCP3n = c2.setPdfchapterId31();
pdfCP4y = c2.setPdfchapterId40();
pdfCP4n = c2.setPdfchapterId41();
```

```
/******No Of
```

```
Test******/
```

```
if(notch1.equals("1")) {
    pdfnotCP1y = c2.setPdfNoOfTests10();
    pdfnotCP1n = c2.setPdfNoOfTests11();
} else if(notch1.equals("2")) {
    pdfnotCP1y = c2.setPdfNoOfTests20();
    pdfnotCP1n = c2.setPdfNoOfTests21();
} else if(notch1.equals("3")) {
    pdfnotCP1y = c2.setPdfNoOfTests30();
    pdfnotCP1n = c2.setPdfchapterId31();
} else if(notch1.equals("4")) {
    pdfnotCP1y = c2.setPdfFinalResult40();
    pdfnotCP1n = c2.setPdfNoOfTests41();
} else if(notch1.equals("5")) {
    pdfnotCP1y = c2.setPdfNoOfTests50();
    pdfnotCP1n = c2.setPdfFinalResult51();
}
```

```
if(notch2.equals("1")) {
    pdfnotCP2y = c2.setPdfNoOfTests10();
    pdfnotCP2n = c2.setPdfNoOfTests11();
} else if(notch2.equals("2")) {
    pdfnotCP2y = c2.setPdfNoOfTests20();
    pdfnotCP2n = c2.setPdfNoOfTests21();
} else if(notch2.equals("3")) {
    pdfnotCP2y = c2.setPdfNoOfTests30();
    pdfnotCP2n = c2.setPdfNoOfTests31();
} else if(notch2.equals("4")) {
```

```
pdfnotCP2n = c2.setPdfNoOfTests41();
} else if(notch2.equals("5")) {
pdfnotCP2y = c2.setPdfNoOfTests50();
pdfnotCP2n = c2.setPdfNoOfTests51();
}
```

```
if(notch3.equals("1")) {
pdfnotCP3y = c2.setPdfNoOfTests10();
pdfnotCP3n = c2.setPdfNoOfTests11();
} else if(notch3.equals("2")) {
pdfnotCP3y = c2.setPdfNoOfTests20();
pdfnotCP3n = c2.setPdfNoOfTests21();
} else if(notch3.equals("3")) {
pdfnotCP3y= c2.setPdfNoOfTests30();
pdfnotCP3n = c2.setPdfNoOfTests31();
} else if(notch3.equals("4")) {
pdfnotCP3y = c2.setPdfFinalResult40();
pdfnotCP3n = c2.setPdfNoOfTests41();
} else if(notch3.equals("5")) {
pdfnotCP3y = c2.setPdfNoOfTests50();
pdfnotCP3n = c2.setPdfNoOfTests51();
}
```

```
if(notch4.equals("1")) {
pdfnotCP4y = c2.setPdfNoOfTests10();
pdfnotCP4n = c2.setPdfNoOfTests11();
} else if(notch4.equals("2")) {
pdfnotCP4y = c2.setPdfNoOfTests20();
pdfnotCP4n = c2.setPdfNoOfTests21();
} else if(notch4.equals("3")) {
pdfnotCP4y= c2.setPdfNoOfTests30();
pdfnotCP4n = c2.setPdfNoOfTests31();
} else if(notch4.equals("4")) {
pdfnotCP4y = c2.setPdfFinalResult40();
pdfnotCP4n = c2.setPdfNoOfTests41();
} else if(notch4.equals("5")) {
pdfnotCP4y = c2.setPdfNoOfTests50();
pdfnotCP4n = c2.setPdfNoOfTests51();
}
```



```
/**AVG***/
***/
```

```
if(ch1avgmarks.equals("1")) {
    pdfavgCP1y = c2.setPdfAvgTest10();
    pdfavgCP1n = c2.setPdfAvgTest11();
} else if(ch1avgmarks.equals("2")) {
    pdfavgCP1y = c2.setPdfAvgTest20();
    pdfavgCP1n = c2.setPdfAvgTest31();
} else if(ch1avgmarks.equals("3")) {
    pdfavgCP1y= c2.setPdfAvgTest30();
    pdfavgCP1n = c2.setPdfAvgTest31();
} else if(ch1avgmarks.equals("4")) {
    pdfavgCP1y = c2.setPdfAvgTest40();
    pdfavgCP1n = c2.setPdfAvgTest41();
} else if(ch1avgmarks.equals("5")) {
    pdfavgCP1y = c2.setPdfAvgTest50();
    pdfavgCP1n = c2.setPdfAvgTest51();
}
```

```
if(ch2avgmarks.equals("1")) {
    pdfavgCP2y = c2.setPdfAvgTest10();
    pdfavgCP2n = c2.setPdfAvgTest11();
} else if(ch2avgmarks.equals("2")) {
    pdfavgCP2y = c2.setPdfAvgTest20();
    pdfavgCP2n = c2.setPdfAvgTest31();
} else if(ch2avgmarks.equals("3")) {
    pdfavgCP2y= c2.setPdfAvgTest30();
    pdfavgCP2n = c2.setPdfAvgTest31();
} else if(ch2avgmarks.equals("4")) {
    pdfavgCP2y = c2.setPdfAvgTest40();
    pdfavgCP2n = c2.setPdfAvgTest41();
} else if(ch2avgmarks.equals("5")) {
    pdfavgCP2y = c2.setPdfAvgTest50();
    pdfavgCP2n = c2.setPdfAvgTest51();
}
```

```
if(ch3avgmarks.equals("1")) {
```

```
pdfavgCP3n = c2.setPdfAvgTest11();
} else if(ch3avgmarks.equals("2")) {
pdfavgCP3y = c2.setPdfAvgTest20();
pdfavgCP3n = c2.setPdfAvgTest31();
} else if(ch3avgmarks.equals("3")) {
pdfavgCP3y= c2.setPdfAvgTest30();
pdfavgCP3n = c2.setPdfAvgTest31();
} else if(ch3avgmarks.equals("4")) {
pdfavgCP3y = c2.setPdfAvgTest40();
pdfavgCP3n = c2.setPdfAvgTest41();
} else if(ch3avgmarks.equals("5")) {
pdfavgCP3y = c2.setPdfAvgTest50();
pdfavgCP3n = c2.setPdfAvgTest51();
}
```

```
if(ch4avgmarks.equals("1")) {
pdfavgCP4y = c2.setPdfAvgTest10();
pdfavgCP4n = c2.setPdfAvgTest11();
} else if(ch4avgmarks.equals("2")) {
pdfavgCP4y = c2.setPdfAvgTest20();
pdfavgCP4n = c2.setPdfAvgTest31();
} else if(ch4avgmarks.equals("3")) {
pdfavgCP4y= c2.setPdfAvgTest30();
pdfavgCP4n = c2.setPdfAvgTest31();
} else if(ch4avgmarks.equals("4")) {
pdfavgCP4y = c2.setPdfAvgTest40();
pdfavgCP4n = c2.setPdfAvgTest41();
} else if(ch4avgmarks.equals("5")) {
pdfavgCP4y = c2.setPdfAvgTest50();
pdfavgCP4n = c2.setPdfAvgTest51();
}
```

```
/******final
Result*****/
```

```
if(ch1finalmark.equals("1")) {
pdffinalCP1y = c2.setPdfFinalResult10();
pdffinalCP1n = c2.setPdfFinalResult11();
} else if(ch1finalmark.equals("2")) {
pdffinalCP1y = c2.setPdfFinalResult20();
pdffinalCP1n = c2.setPdfFinalResult21();
```

```
    pdfFinalCP1y = c2.setPdfFinalResult30();
    pdfFinalCP1n = c2.setPdfFinalResult31();
} else if(ch1finalmark.equals("4")) {
    pdfFinalCP1y = c2.setPdfFinalResult40();
    pdfFinalCP1n = c2.setPdfFinalResult41();
} else if(ch1finalmark.equals("5")) {
    pdfFinalCP1y = c2.setPdfFinalResult50();
    pdfFinalCP1n = c2.setPdfFinalResult51();
}
```

```
if(ch2finalmark.equals("1")) {
    pdfFinalCP2y = c2.setPdfFinalResult10();
    pdfFinalCP2n = c2.setPdfFinalResult11();
} else if(ch2finalmark.equals("2")) {
    pdfFinalCP2y = c2.setPdfFinalResult20();
    pdfFinalCP2n = c2.setPdfFinalResult21();
} else if(ch2finalmark.equals("3")) {
    pdfFinalCP2y = c2.setPdfFinalResult30();
    pdfFinalCP2n = c2.setPdfFinalResult31();
} else if(ch2finalmark.equals("4")) {
    pdfFinalCP2y = c2.setPdfFinalResult40();
    pdfFinalCP2n = c2.setPdfFinalResult41();
} else if(ch2finalmark.equals("5")) {
    pdfFinalCP2y = c2.setPdfFinalResult50();
    pdfFinalCP2n = c2.setPdfFinalResult51();
}
```

```
if(ch3finalmark.equals("1")) {
    pdfFinalCP3y = c2.setPdfFinalResult10();
    pdfFinalCP3n = c2.setPdfFinalResult11();
} else if(ch3finalmark.equals("2")) {
    pdfFinalCP3y = c2.setPdfFinalResult20();
    pdfFinalCP3n = c2.setPdfFinalResult21();
} else if(ch3finalmark.equals("3")) {
    pdfFinalCP3y = c2.setPdfFinalResult30();
    pdfFinalCP3n = c2.setPdfFinalResult31();
} else if(ch3finalmark.equals("4")) {
    pdfFinalCP3y = c2.setPdfFinalResult40();
    pdfFinalCP3n = c2.setPdfFinalResult41();
} else if(ch3finalmark.equals("5")) {
    pdfFinalCP3y = c2.setPdfFinalResult50();
}
```

```
}
```

```
if(ch4finalmark.equals("1")) {  
    pdffinalCP4y = c2.setPdfFinalResult10();  
    pdffinalCP4n = c2.setPdfFinalResult11();  
} else if(ch4finalmark.equals("2")) {  
    pdffinalCP4y = c2.setPdfFinalResult20();  
    pdffinalCP4n = c2.setPdfFinalResult21();  
} else if(ch4finalmark.equals("3")) {  
    pdffinalCP4y = c2.setPdfFinalResult30();  
    pdffinalCP4n = c2.setPdfFinalResult31();  
} else if(ch4finalmark.equals("4")) {  
    pdffinalCP4y = c2.setPdfFinalResult40();  
    pdffinalCP4n = c2.setPdfFinalResult41();  
} else if(ch4finalmark.equals("5")) {  
    pdffinalCP4y = c2.setPdfFinalResult50();  
    pdffinalCP4n = c2.setPdfFinalResult51();  
}
```

```
double pdfYescp1 = pdfCP1y*pdfnotCP1y*pdfavgCP1y*pdffinalCP1y;  
double pdfYescp2 = pdfCP2y*pdfnotCP2y*pdfavgCP2y*pdffinalCP2y;  
double pdfYescp3 = pdfCP3y*pdfnotCP3y*pdfavgCP3y*pdffinalCP3y;  
double pdfYescp4 = pdfCP4y*pdfnotCP4y*pdfavgCP4y*pdffinalCP4y;
```

```
double pdfNocp1 = pdfCP1n*pdfnotCP1n*pdfavgCP1n*pdffinalCP1n;  
double pdfNocp2 = pdfCP2n*pdfnotCP2n*pdfavgCP2n*pdffinalCP2n;  
double pdfNocp3 = pdfCP3n*pdfnotCP3n*pdfavgCP3n*pdffinalCP3n;  
double pdfNocp4 = pdfCP4n*pdfnotCP4n*pdfavgCP4n*pdffinalCP4n;
```

```
double totChapYes = (pdfYescp1+pdfYescp2+pdfYescp3+pdfYescp4)/4;  
double totChapNo = (pdfNocp1+pdfNocp2+pdfNocp3+pdfNocp4)/4;
```

```
session.setAttribute("Yes",totChapYes);  
Double yes=(Double)session.getAttribute("Yes");  
double YES = Double.valueOf(yes);
```

```
session.setAttribute("No",totChapNo);
```

```
double NO = Double.valueOf(no);
```

```
    if(totChapYes>totChapNo)
```

```
    {  
        result = "Improve Learner Proficiency";
```

```
    }  
    else
```

```
    {  
        result = "No Need Improving Learner Proficiency";
```

```
    }  
    session.setAttribute("result",result);    //for ajax response
```

```
getServletConfig().getServletContext().getRequestDispatcher("/success.jsp").fo  
rward(request, response);
```

```
    }  
    catch(Exception e2)  
    {  
        System.out.println("Exception "+e2.toString());
```

```
    }  
}
```

```
public String setAvg(String str){
```

```
String avg="";
```

```
int avgmark=Integer.parseInt(str);
```

```
if(avgmark>=9){
```

```
    avg="1";
```

```
}else if(avgmark<9 && avgmark>=7 ){
```

```
    avg="2";
```

```
}else if(avgmark<7 && avgmark>=5){
```

```
    avg="3";
```

```
}else if(avgmark<5 && avgmark>3){
```

```
    avg="4";
```

```
}else if(avgmark<=3 ){
```

```
    avg="5";
```

```
}
```

```
return avg;
```

```
}
```

```
public String findFinal(String not,String avg){
```

```
String temp="";
```

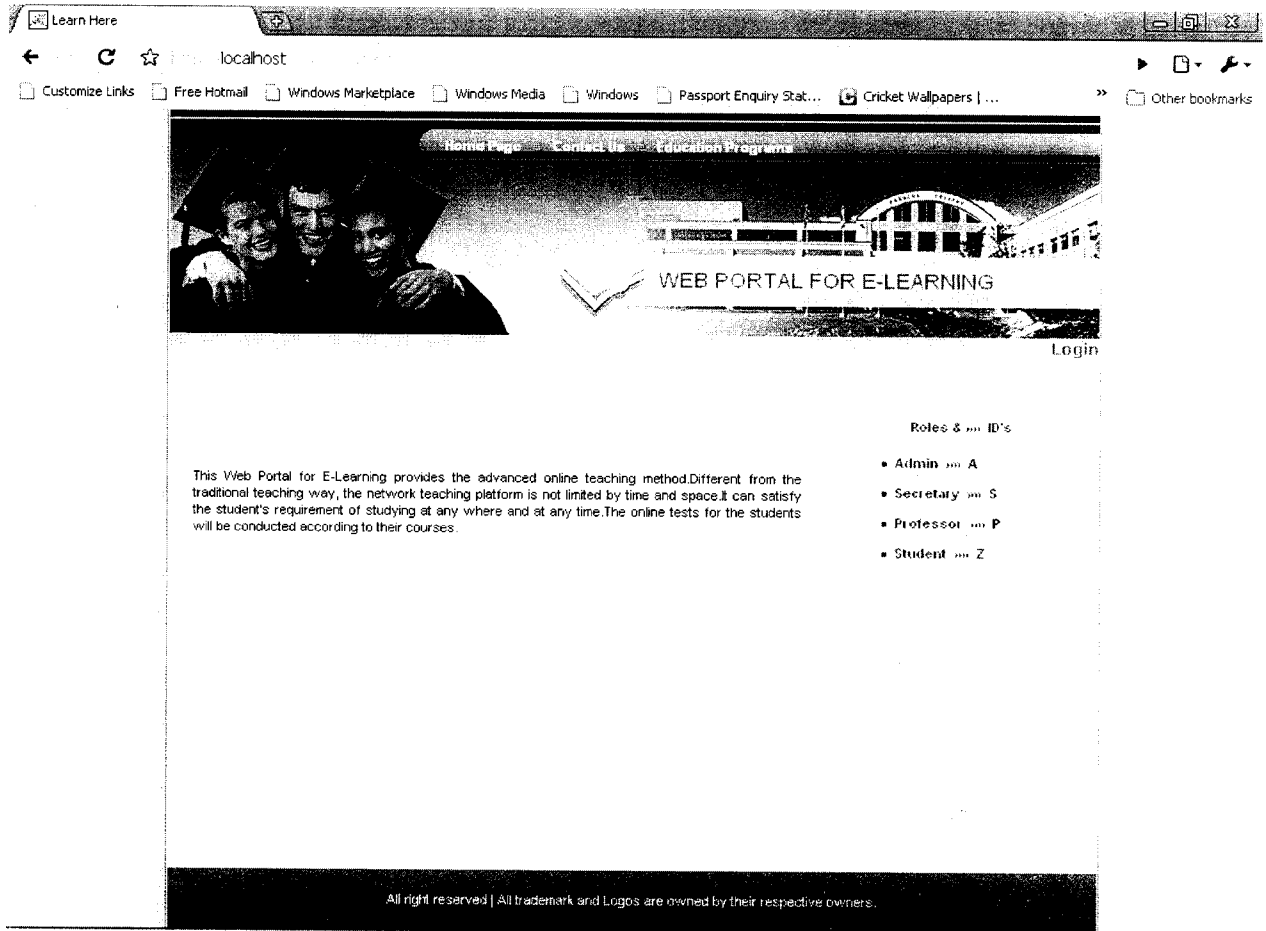
```
int not1=Integer.parseInt(not);
if(avgmark==1){
    if(not1==1){
        temp="1";
    }else if(not1==2){
        temp="1";
    }else if(not1==3){
        temp="2";
    }else if(not1==4){
        temp="3";
    }else if(not1==5){
        temp="4";
    }
}else if(avgmark==2){
    if(not1==1){
        temp="1";
    }else if(not1==2){
        temp="2";
    }else if(not1==3){
        temp="3";
    }else if(not1==4){
        temp="4";
    }else if(not1==5){
        temp="4";
    }
} else if(avgmark==3){
    if(not1==1){
        temp="2";
    }else if(not1==2){
        temp="4";
    }else if(not1==3){
        temp="4";
    }else if(not1==4){
        temp="5";
    }else if(not1==5){
        temp="5";
    }
} else if(avgmark==4){
    if(not1==1){
        temp="3";
    }else if(not1==2){
```

```
}else if(not l==3){
    temp="5";
}else if(not l==4){
    temp="5";
}else if(not l==5){
    temp="5";
}
} else if(avgmark==5){
    if(not l==1){
        temp="4";
    }else if(not l==2){
        temp="5";
    }else if(not l==3){
        temp="5";
    }else if(not l==4){
        temp="5";
    }else if(not l==5){
        temp="5";
    }
}
return temp;
}
```

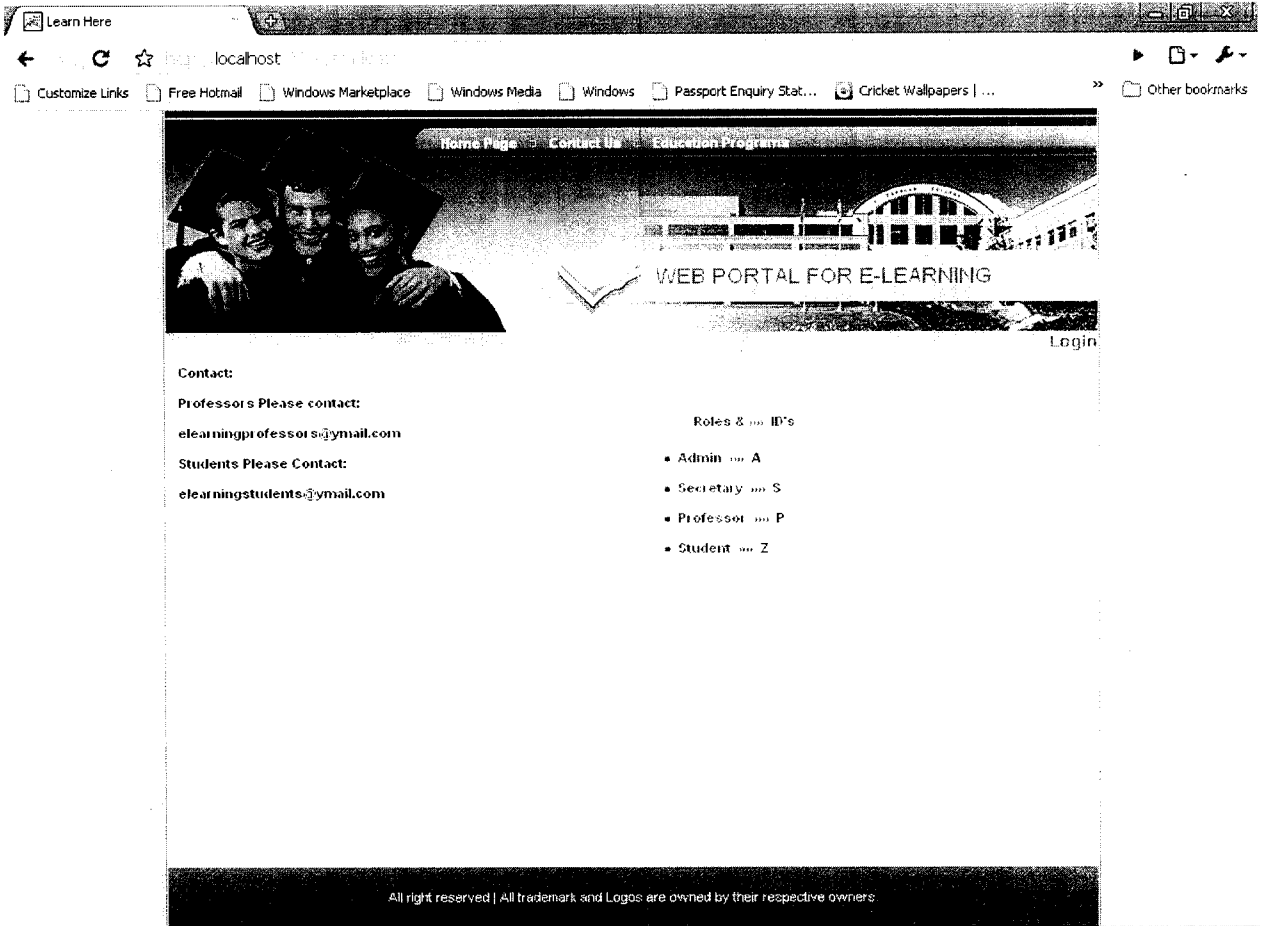
```
}
```


CHAPTER-6

SNAP SHOTS



HOME PAGE



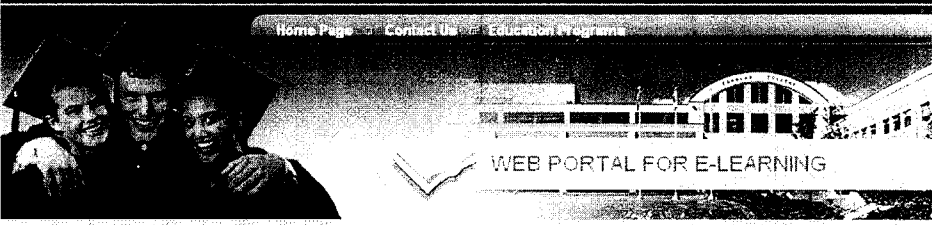
CONTACT PAGE

Learn Here

localhost:8080/learn.html

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks

Home Page Contact Us Education Program



WEB PORTAL FOR E-LEARNING

Logout

Enter your ID: A

Username: a

Password: |

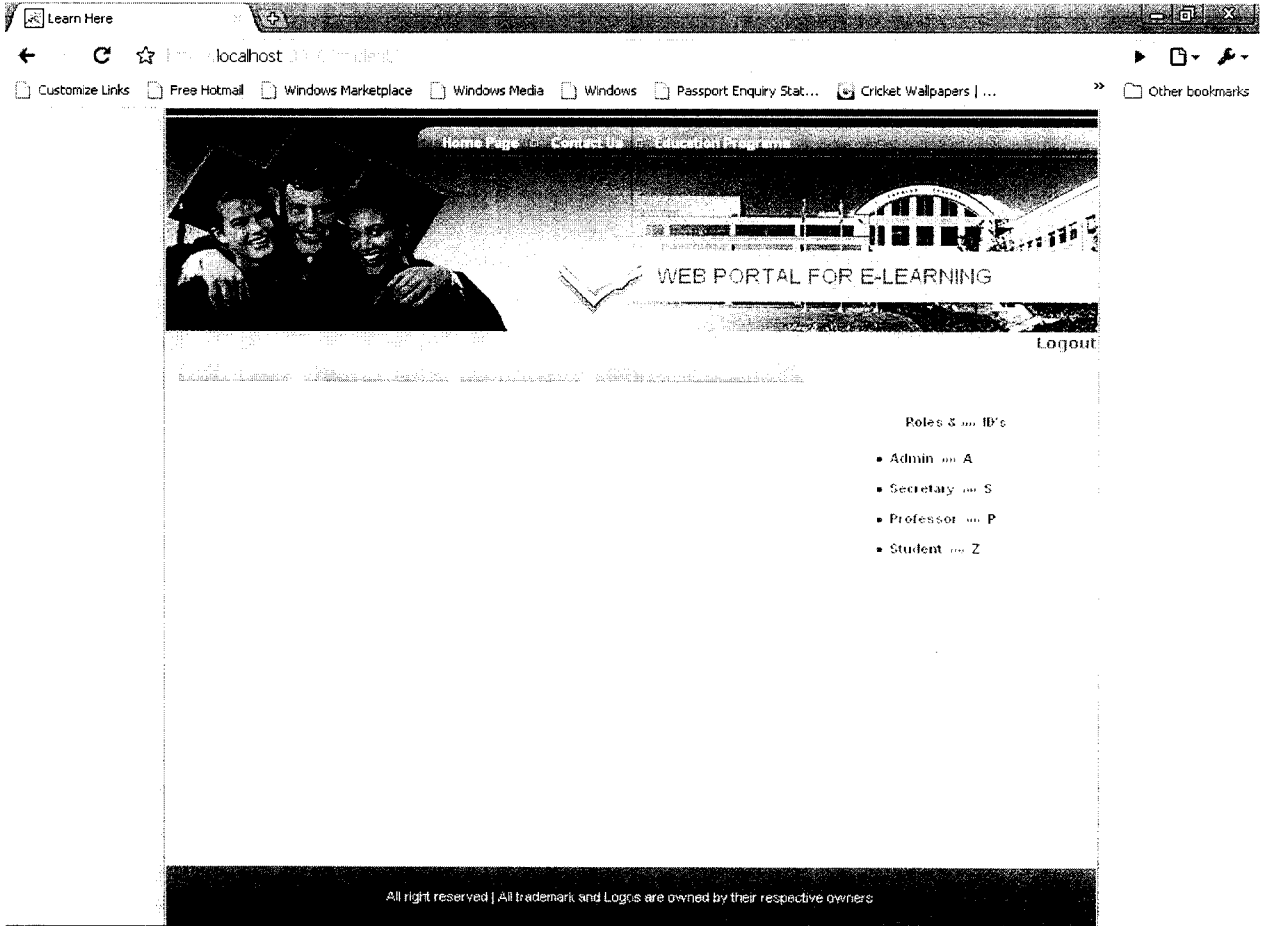
Login Clear

Roles & ID's

- Admin ID's A
- Secretary ID's S
- Professor ID's P
- Student ID's Z

All right reserved | All trademark and Logos are owned by their respective owners.

ADMIN LOGIN



ADMIN HOME

Learn Here

localhost

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks

Home Page Contact Us Education Programs

Logout

ID S

Username bala

Password ****

Add Clear

Roles & ID's

- Admin ... A
- Secretary ... S
- Professor ... P
- Student ... Z

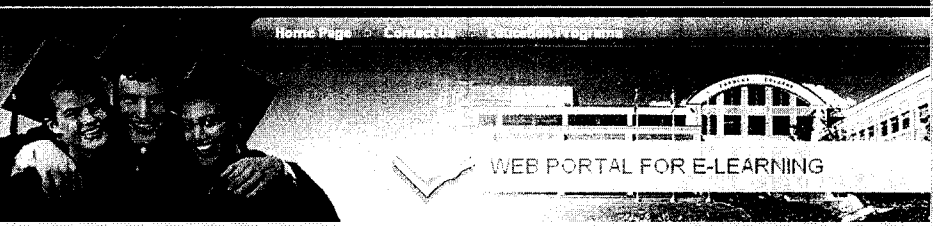
All right reserved | All trademark and Logos are owned by their respective owners.

ADDING SECRETARY

Learn Here

localhost:8007/Student/

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks



Normal Page

UserName	Password	Roles & ID's
b	b	Admin A
bb	bb	Secretary S
bala	bala	Professor P
		Student Z

All right reserved | All trademark and Logos are owned by their respective owners.

VIEWING SECRETARY DETAILS

Learn Here

localhost

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks

Home Page Contact Us Education Programs

Username **Password**

b	b
bb	bb
bala	bala

Roles & IP's

- Admin ... A
- Secretary ... S
- Professor ... P
- Student ... Z

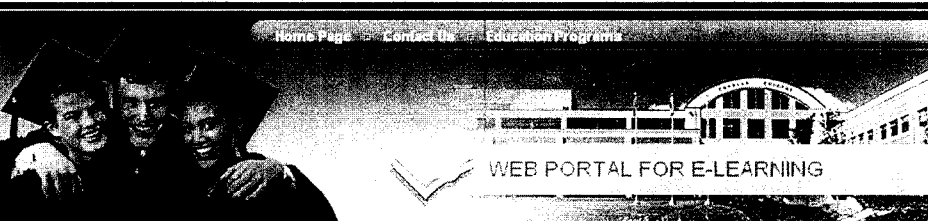
All right reserved | All trademark and Logos are owned by their respective owners.

EDITING SECRETARY DETAILS

Learn Here localhost

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks

Home Page Contact Us Education Programs



Logout

Username	Password		Roles & ID's
b	b	<input type="checkbox"/>	• Admin → A
bb	bb	<input type="checkbox"/>	• Secretary → S
bala	bala	<input type="checkbox"/>	• Professor → P
			• Student → Z

All right reserved | All trademark and Logos are owned by their respective owners.

DELETING SECRETARY DETAILS

Learn Here localhost

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks



Home | About | Contact Us | Education Programs

WEB PORTAL FOR E-LEARNING

Logout

Secretary Page

Select your ID: Professor

Username pro Professor
Student

Password ***

Add Clear

Roles & ID's

- Admin ID: A
- Secretary ID: S
- Professor ID: P
- Student ID: Z

All right reserved | All trademark, and Logos are owned by their respective owners.

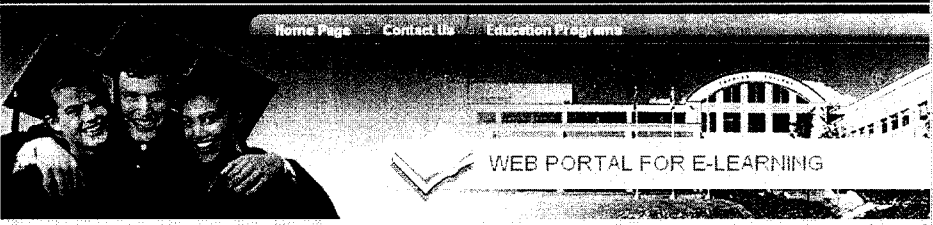
ADDING PROFESSORS AND STUDENT

Learn Here

localhost

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks

home Page Contact Us Education Programs



WEB PORTAL FOR E-LEARNING

Logout

Secretary Page

Professor
 Student

UserName	Password	Roles & ID's
c	c	• Admin → A
cc	cc	• Secretary → S
pro	pro	• Professor → P
		• Student → Z

All right reserved | All trademark and Logos are owned by their respective owners.

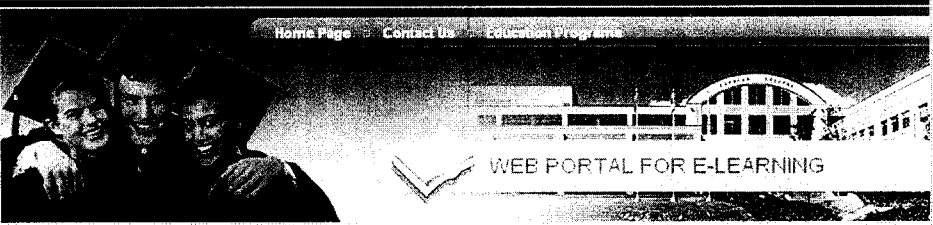
VIEWING PROFESSOR OR STUDENT DETAILS

Learn Here

localhost

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks

Home Page Contact Us Education Programs



Logout

Secretary Page

Professor Student

UserName	Password
c	c
cc	cc
pro	pro

Roles & ID's

- Admin ... A
- Secretary ... S
- Professor ... P
- Student ... Z

All right reserved | All trademark and Logos are owned by their respective owners

EDITING PROFESSOR OR STUDENT DETAILS

Learn Here

http://localhost:3000/secretary/

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks

Home Page Contact Us Education Programs

Logout

Secretary Page

Professor Student

UserName	Password	
c	c	<input checked="" type="checkbox"/>
cc	cc	<input type="checkbox"/>
pro	pro	<input type="checkbox"/>

Roles & ID's

- Admin ... A
- Secretary ... S
- Professor ... P
- Student ... Z

All right reserved | All trademark and Logos are owned by their respective owners

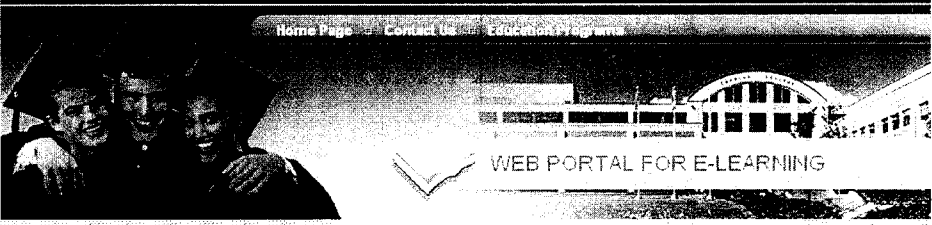
DELETING PROFESSOR OR STUDENT DETAILS

Learn Here

localhost:8080/Account/...

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks

home Page Contact Us User Registration



WEB PORTAL FOR E-LEARNING

Logout

Welcome To Professor Page

CSE IT ECE

CSE

Roles & ID's

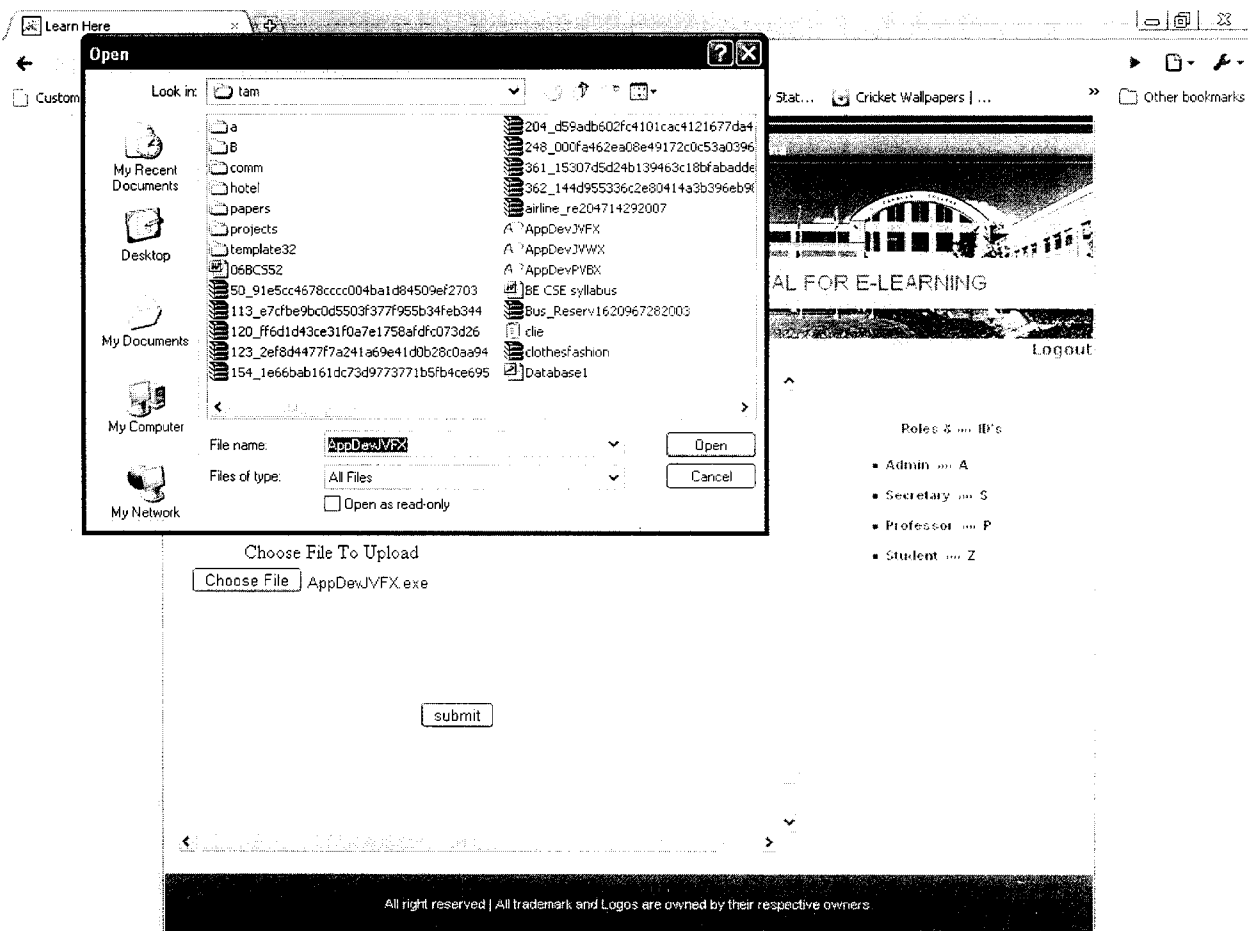
- Admin ID: A
- Secretary ID: S
- Professor ID: P
- Student ID: Z

Choose File To Upload

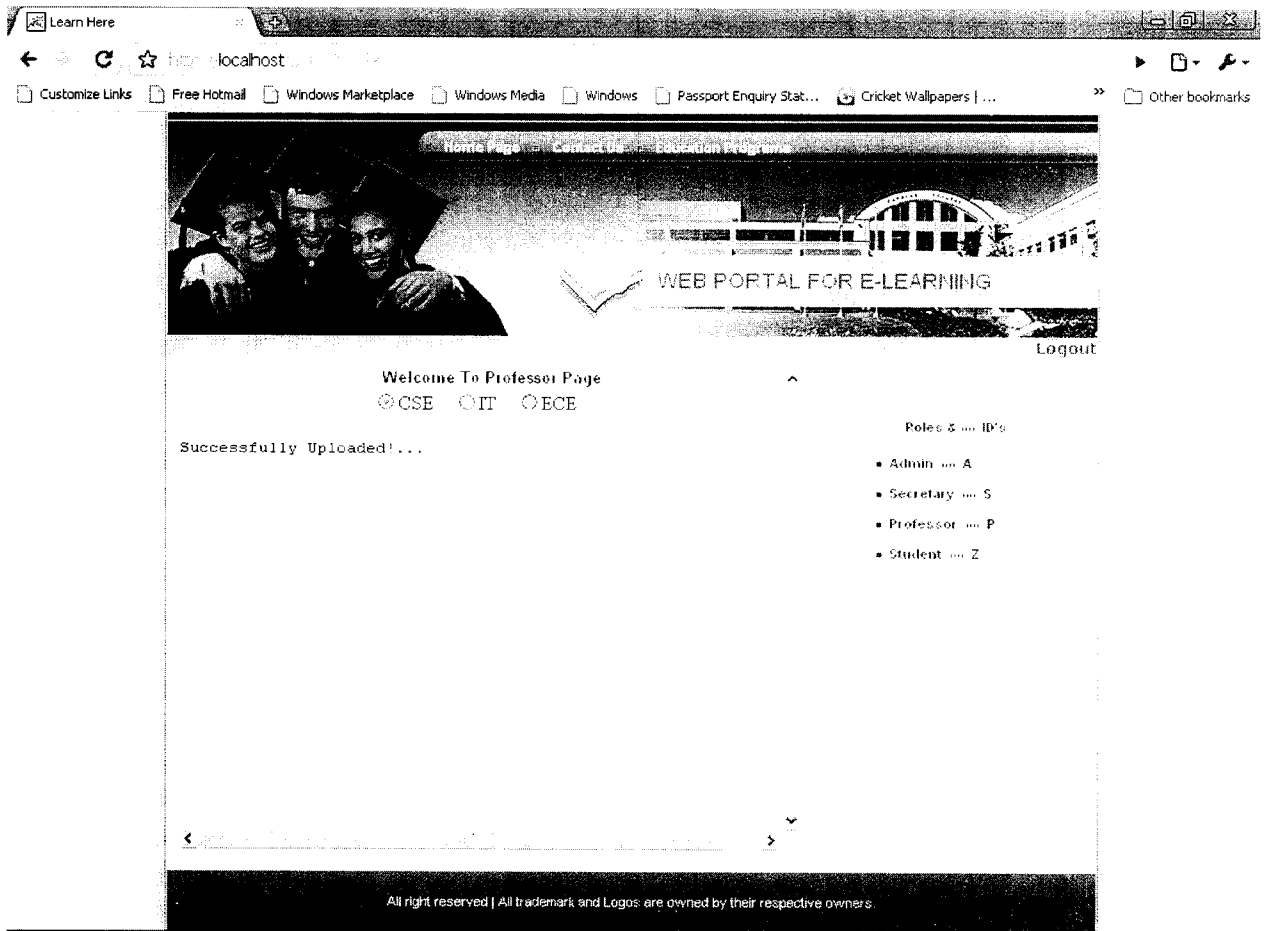
No file chosen

All right reserved | All trademark and Logos are owned by their respective owners.

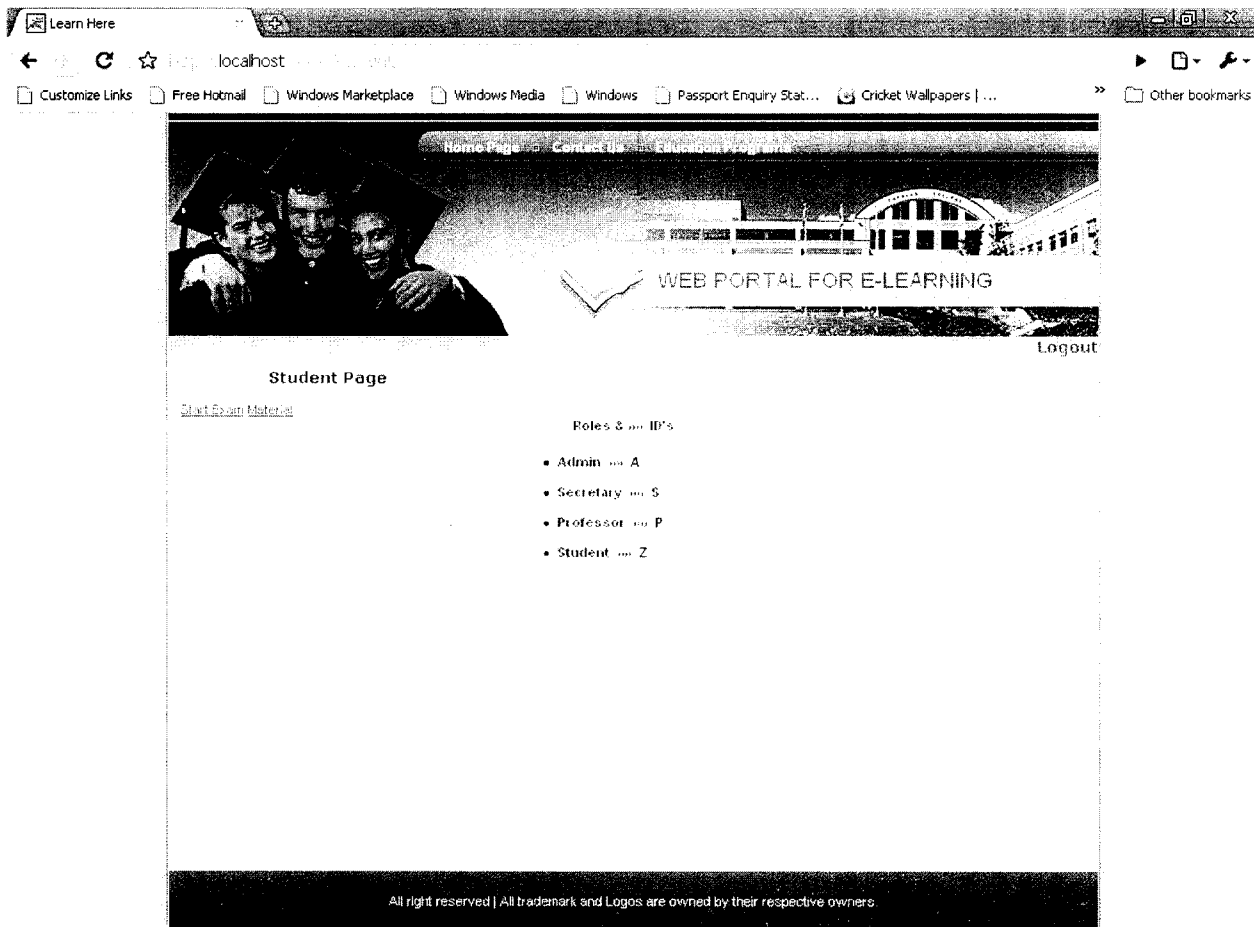
UPLOADING STUDY MATERIALS



CHOOSING FILE TO UPLOAD



AFTER UPLOAD IS SUCCEEDED

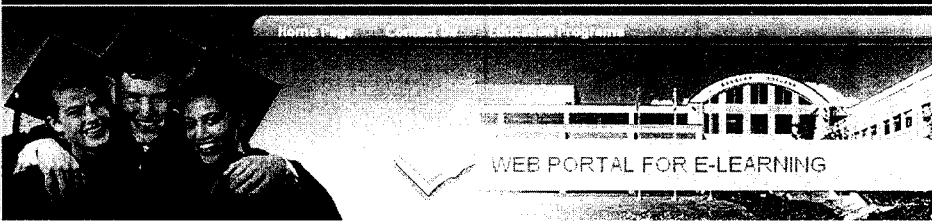


STUDENT HOME

Learn Here

localhost

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers |... Other bookmarks



Logout

Student Page

Materials

- [Java Applet Basics.mht](#) CSE
- [Systolic pressure is peak pressure in the arteries IT](#)
- [Java Applet Basics.mht](#) CSE
- [saple.html](#) CSE
- [AppDevJVFEX.exe](#) CSE

Links

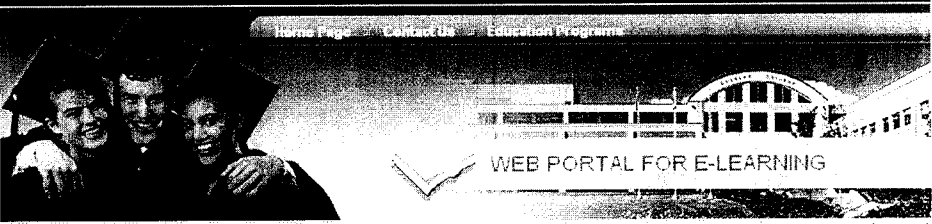
Roles & ID's

- Admin ID: A
- Secretary ID: S
- Professor ID: P
- Student ID: Z

MATERIALS LIST TO DOWNLOAD

Learn Here localhost

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks



Home Page | About Us | Education Programs

WEB PORTAL FOR E-LEARNING

Logout

Student Page

Start Euan

chapter1

Roles & ID's

Which colour is used to indicate instance methods in the standard "javadoc" format documentation

- blue
- red
- purple
- orange

Next

- Admin ... A
- Secretary ... S
- Professor ... P
- Student ... Z

ONLINE TEST

Learn Here

localhost

Customize Links Free Hotmail Windows Marketplace Windows Media Windows Passport Enquiry Stat... Cricket Wallpapers | ... Other bookmarks

Home Page Contact Us Education Bloggers

WEB PORTAL FOR E-LEARNING

Logout

Student Page

Start Exam

Chapter	No Of Test	Avg Mark	Final result
1	1	1	1
2	1	3	2
3	1	2	1
4	2	2	2

Roles & ID's

- Admin ... A
- Secretary ... S
- Professor ... P
- Student ... Z

No Need Improving Learner Proficiency

REPORT GENERATED AFTER THE TEST

CHAPTER-7

CONCLUSION AND FUTURE WORK

7.1 CONCLUSION

In this project we have developed an e-learning portal which will help the e-learning community to identify their strength and weaknesses along with the e-content which will help them to enrich their knowledge. Similarly the faculties having expertise in any subject can serve the community by uploading e-content of any format. Due to this it percolates to all the e-learners. An analysis process helped us to identify in discovering the student's needs regarding the chapters that needs more learning.

The student who follows the recommendations will get better results in shorter time due to optimization of their learning process.

7.2 FUTURE WORKS:

As future works the following directions may be taken one regards in selecting the question from the question bank in a random manner. More over timing constraint can be incorporated to take the test. An attempt might be taken to randomize the choices also.

Interactivity may be added between the faculty and the e-learner to clarify doubts through comments / posts. It will be bright if we add interactivity using Audio and Video components.

Mechanisms should be incorporated to prevent Intellectual Property Rights violations.

REFERENCES:

[1] <http://www.efrontlearning.net/> **eFront** is a complete e-learning software with a good looking Ajaxed interface. It has the ability to assign projects creating surveys.

[2] <http://moodle.org/> **Moodle** is one of the most popular **open source e-learning system**. It is built with PHP & uses MySQL to store data.

[3] <http://www.claroline.net/> **Claroline** is an **open source e-learning and e-working platform** that allows teachers to create effective online courses and to manage learning and collaborative activities on the web.

[4] Referred the book “JAVA complete Reference” written by Herbert Schildt.

[5] Referred the web page “<http://dev.mysql.com/doc/refman/5.0/en/tutorial.html>” for the MySQL

[6] <http://www.sakaiproject.org/> **Sakai** is a powerful yet flexible solution that supports not only teaching and learning but also research. Using the application, you can create courses, manage assignments, share documents, and prepare exams. As a result, grade information's can be calculated, stored and distributed.