NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

Student Management System

Graduation Project

COM-400

Student: Muhammed Emin ER (20031517)

Supervisor: Assist. Prof. Dr. Hüseyin SEVAY

Nicosia - 2008

ACKNOWLEDGMENTS

I would like to thank my supervisor Assist. Prof. Dr. Hüseyin Sevay for his invaluable advice and belief in my work and myself over the course of this Graduation Project.

I'd also like to express my gratitude to Near East University for the scholarship that made the work possible.

Finally, I thank my family for their constant encouragement and support during the preparation of this project.

ABSTRACT

Saving data is important thing for using it later in a work. There are a lot of saving data procedures. Too many years people use pens and papers for saving them. Nowadays with new technologies, most popular way of collecting data is saving to inside of the computer. You can access data more faster and processing from computer. This new technology covers our needs like, fast processing, able to searching very fast and so on.

In my work, I purpose to collect the informations of the students all data to computer environment. For covering this purpose, the system should not run with dependences of any specific environment. It should be elastic and develop able, for these reasons, it should be coding by object oriented programming. The system should run on every environment and fast, remote control and easy to use. And so, for doing this work, it should not use more sources(it should be cheap). For all this things, coding and using the other programs should be "Open Source" system will increase our work more elastic and develop able. So, we can access our purpose and not spend more sources.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

NEU	Near East University
SMS	Student Management System
SIMS	Student Information and Management System
IDE	integrated development environment
GUI	Graphical User Interface

INTRODUCTION

Student Management System (SMS) is educational establishments to manage students data. Student Management System provides capabilities for entering student test and other scores, building and showing student schedules, uploading documents and managing many other student-related data needs in a university. Also known as student management system(SMS), student information system(SIS) or student information management system(SIMS).

Chapter One describes of this project investigate the development of SMS for the Near East University.

Chapter Two presents the architecture of Java, JSP, Linux and other technologies.

Chapter Three presents of the application of Student Management System, it's run time and work results.

Chapter 1

Student Management System

1.1 What is SMS?

SMS, Student Management System is a software for educational establishments to manage students data. Student Management System provides capabilities for entering student test and other scores, building and showing student schedules, uploading documents and managing many other student-related data needs in a university. Also known as student management system(SMS), student information system(SIS) or student information management system(SIMS).

The mission of the Student Information System SIS Project is to create an integrated information technology environment for students, faculty, instructor, staff and administration.

This system is implemented in a small organizations to cover student records alone. The aim is cover the most of student running organizations with university responsibilities. The system can be scaled to different levels of activity and can be configured by their homes or anywhere has internet by explorer program.

The system is designed in large solutions that have students data, functions in finance, human resources, estates management and more may be catered for by the software. Also the system can be develop able to the way that you want.

The system uses open source software as MySQL for collecting data, apache tomcat for serving the web site, Java for programming and more. It makes the system flexible and configurable. Whole the system can be changeable. Now, everything capable up to the system administrators.

1.2 Why SMS is necessary?

Saving data is very important thing for using it later in a work. There are a lot of saving data procedures. Too many years people use pens and papers for saving them. Nowadays with new technologies, most popular way of collectingdata is saving to inside of the computers. Accessing data is done more faster and processing from computer. This new technology covers our needs like, fast processing, able to searching very fast and so on.

In the universities, informations stored in papers cause needing to control it, save them in archive rooms for many years. But computer technology represents collecting data in the computers is more usable, easy to save and can be processed in seconds.

In Near East University, student registration is done on the paper and there is a program to control them. But it is not useful. Because, it has dependences and just for instructors. Students can not control their information in that system. The program runs on Windows Operating System but not the other systems. Users need this program is installed in their computers and can not access out of their computers, For students, there is a web-site for showing to students their finance information and grades. But all those systems do not run all together.

The aim of this project is to control everything in one interface. It will cause managing student information will be more easy. Of course the system of this project is designed to able to run with the other systems, like Registration System. In the project, it is assumed to run in backside.

1.3 Features of SMS

The common functions of students, instructors, staffs and administrator records are to support the maintenance of personal, study and work information relating to:

Handling student information and documents,
Handling the admissions process,
Enrolling new student and storing teaching option choices,
Handling examinations, grades, graduation,
Collection of information,
Less paper driven with SIS system,
Update technology infrastructure for more effective and flexible delivery of new systems.
Checking grades remote.
And so many features there are.

1.4 Where to use?

The system controls the functions of students, means, it was designed for using in education associations. SMS for every universities. This project was designed for NEU and it's education system was based. That is why more efficient to use in the universities. Also it is more close to universities but it can be implemented to other kind of schools in any education system.

Nowadays, SMS is used the most of the universities all over the world. This project is assumed to run in Near East University. In a word, information of courses and departments are based NEU education system. Examples are taken from them.

Chapter 2

SYSTEM NEEDS AND TECHNIQUES

2.1 What does the system need?

Development tools are defined all of open sources. There are a lot of open source software all over the world. Open source softwares are selected that the most used, common and has good documents for the project.

The main used softwares are:

1. Operating System
2. Database Software

3. Programming Language
1. Server Software
2. IDE for compiling Source Code

2.1.1 Operating System

Operating system should be open source and most popular is Linux. There are a lot of Linux distributions all over the world.

Pardus distribution is chosen as the operating system of the project that run on it. Because Pardus is build on the architecture of Linux.

The advantages of Linux are :
1) There is a lot of development tools on it.
2) It has GPL license, means free to use.
3) Portable
4) Easy to use network tools.
5) Open source.
6) Fast to run.
7) Friendly with the users.
8) Support.
9) Good documentation.
and there are a lot of advantages.

Also, Pardus has this advantages all.

2.1.1.1 Linux

Linux is the name usually given to Unix-like computer operating system that uses the Linux kernel. Linux is the most famous examples of free software open source development. It's source codes can be freely modified, used and redistributed by anyone.

The name "Linux" comes from the Linux kernel, started in 1991 by Linus Torvalds. The system's utilities and libraries usually come from the GNU operating system, announced in 1983 by Richard Stallman. The GNU contribution is the basis for the alternative name "GNU/Linux".

The GNU Project, started in 1984, had the goal of creating a "complete Unix-compatible software system" made entirely of free software. In 1985, Richard Stallman created the Free Software Foundation and developed the GNU General Public License (GNU GPL). Many of the programs required in an OS (such as libraries, compilers, text editors, a Unix shell, and a windowing system) were completed by the early 1990s, although low level elements such as device drivers, daemons, and the kernel stalled and incomplete. Linus Torvalds has said that if the GNU kernel had been available at the time (1991), he would not have decided to write his own.

Linux is a modular Unix-like operating system. It derives mush of its basic design from principles established in Unix during the 1970s and 1980s.Linux uses the Linux kernel, which handles process control, networking, peripheral and file system access. Device drivers are integrated directly with the kernel.

Much of Linux's higher-level functionality is provided by separate projects which interface with the kernel. The GNU user land is an important part of most Linux systems, providing the shell and Unix tools which carry out many basic operating system tasks. On top of the kernel, these tools form a Linux system with a graphical user interface that can be used, usually running in the X Window System.

Linux can be controlled by one or more of a text-based command line interface (CLI), graphical user interface (GNU)(usually the default for desktop), or through controls on the device itself.

On desktop machines, KDE, GNOME and Xfce are the most popular user interfaces. Most popular user interfaces run on top of the X Window System (X), which provides network transparency, enabling a graphical application running on one machine to be displayed and controlled from another.

A Linux system usually provides a command line interface of some sort through a shell, which is the traditional way of interacting with a Unix system. A Linux distribution specialized for servers may use the CLI as its only interface. A "headless system" run without even a monitor can be controlled by the command line via a protocol such as SSH or telnet.

Linux is not the only such operating system, although it is the best-known and most widely used. Some free and open source software licenses are based on the principle of copyleft, a kind of reciprocity: any work derived from a copyleft piece of software must also be copyleft itself. The most common free software license, the GNU GPL, is a form of copyleft, and is used for the Linux kernel and many of the components from the GNU project.

Most Linux distributions support dozens of programming languages. The most common collection of utilities for building both Linux applications and operating system programs is found within the GNU tool chain, which includes the GNU Compiler Collection (GCC) and the GNU build system. Amongst others, GCC provides compilers for Ada, C, C++, Java and Fortran. The Linux kernel itself is written to be compiled with GCC.

Most distributions are also support for Perl, Ruby, Python and other dynamic languages. A number of Java Virtual Machines and development kits run on Linux, including the original Sun Microsystems JVM, and IBM's J2SE RE, as well as many open-source projects like Kaffe.

2.1.1.2 Pardus

Pardus is an open source Linux distribution developed in Turkey, as a product of the Pardus Project. It was named from Anatolian Leopard's Latin name which is "Panthera pardus tulliana".

Pardus was started and is developed by Turkish National Research Institute of Electronics and Cryptology (UEKAE), which is under the Scientific and Technological Research Council of Turkey(TUBITAK).

The first version, Pardus I.O, was released on 2005-12-26. The Pardus Live CD was the first product of the Pardus Project. The latest stable version (Pardus 2007.3) of the Live CD includes the Linux 2.6. 18 kernel, the OpenOffice.org office suite, Internet tools (browser, e-mail, instant messaging, etc), multimedia and graphics tools (video player, music player, etc.), games, and many other applications. COMAR is the configuration manager developed in house, and Tasma is the custom KDE and system configuration tool.

Pisi (Packages Installed Successfully as Intended) is the package management system of Pardus. It is the primary tool for installing, upgrading and removing software packages. PISI stores and handles the dependencies for the various packages, libraries, and COMAR tasks. Also more package can be supported, like "tar.gz". For RPM packages installation, convert rpm to tar.gz packages then is installed with rpm2targz software.

2.1.1.3 Other Linux Distributions

There are a lot of Linux distributions all over the world. Mostly similar distributions and softwares. There is no big differences them but their package manager, installation and some similar things are different. This makes Linux distributions powerful and different users for different versions.

Most popular Linux distributions are:
Red Hat, Ubuntu, OpenSuse, Pardus, PCLinuxOS, Gentoo Linux, CentOS, Debian, Fedora, Knoppix, Mandriva Linux, Slackware and much more.

2.1.2 DatabaseSoftware

Most popular databases are Oracle, MySQL and MsSQL. These all databases have good documentation and support. But MsSQL and Oracle are not open source. In these databases, just MySQL is open source. It has a developer zone, wide support and well designed documentation in the web-site ofMySQL. It runs on the konsole and not use more computer sources.

2.1.2.1 MySQL

MySQL is 3: multi threaded, multi-user SQL database management system (DBMS). The program runs as a server providing multi-user access to a number of databases. MySQL was owned by Sun Microsystems which holds the copyright to most of the code base. The project's source code is available under terms of the GNU General Public License, as well as under a variety of proprietary agreements,

MySQL is popular for web applications and runs on Linux, BSD, Mac and Windows platforms. It has open source bug tracking tools like Bugzilla. It's popularity for use with web applications is closely tied to the popularity of PHP and Ruby on Rails, which are often combined with MySQL.

2.1.2.2 MySQL Installation

Open "Pardus> Package Manager" (Shown in Figure 1.1)

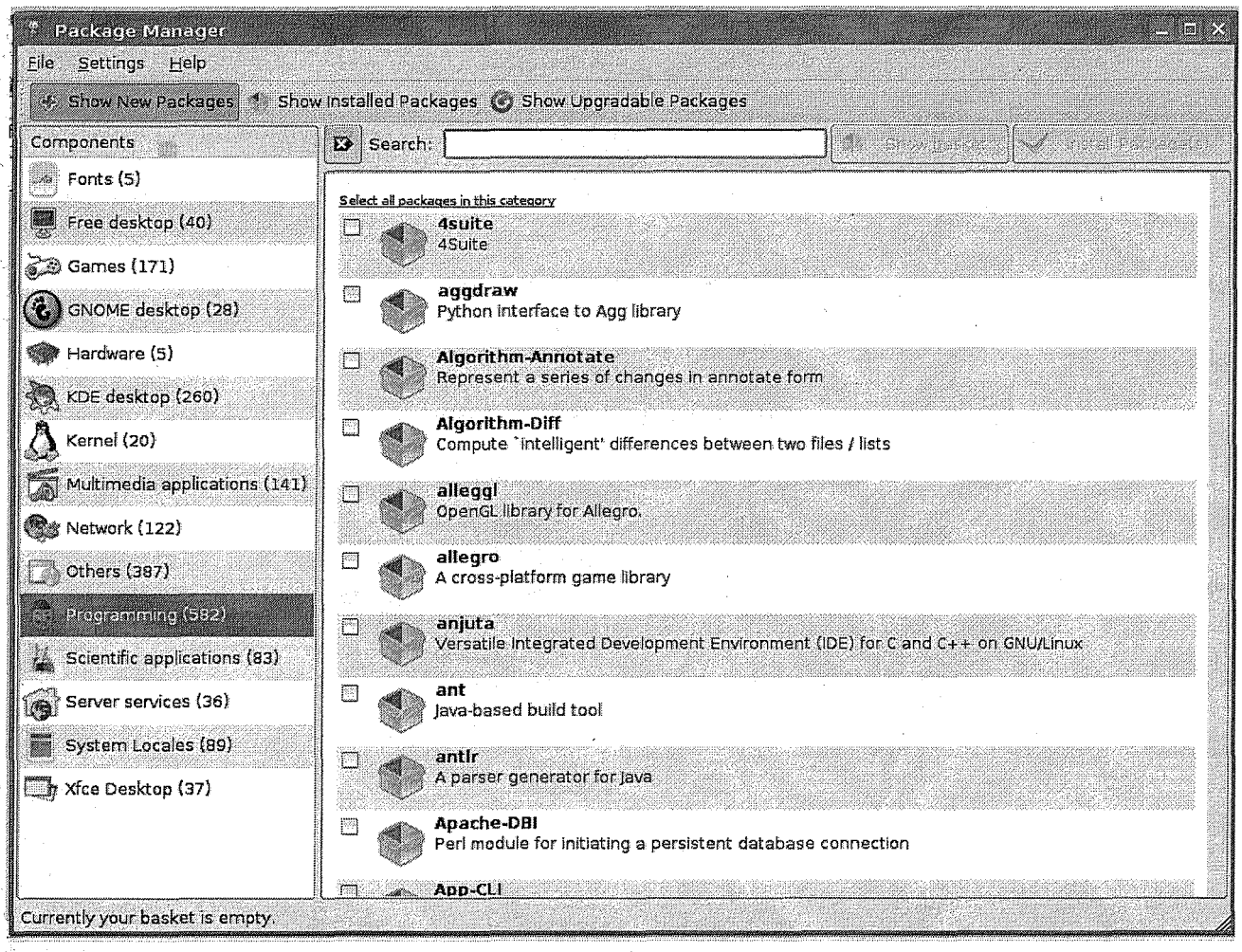

Figure 1.1: Package Manager

Write "MySQL" to search text and find it from the panel then mark it. Then click "Install Package(s)" (Shown in Figure 1.2).

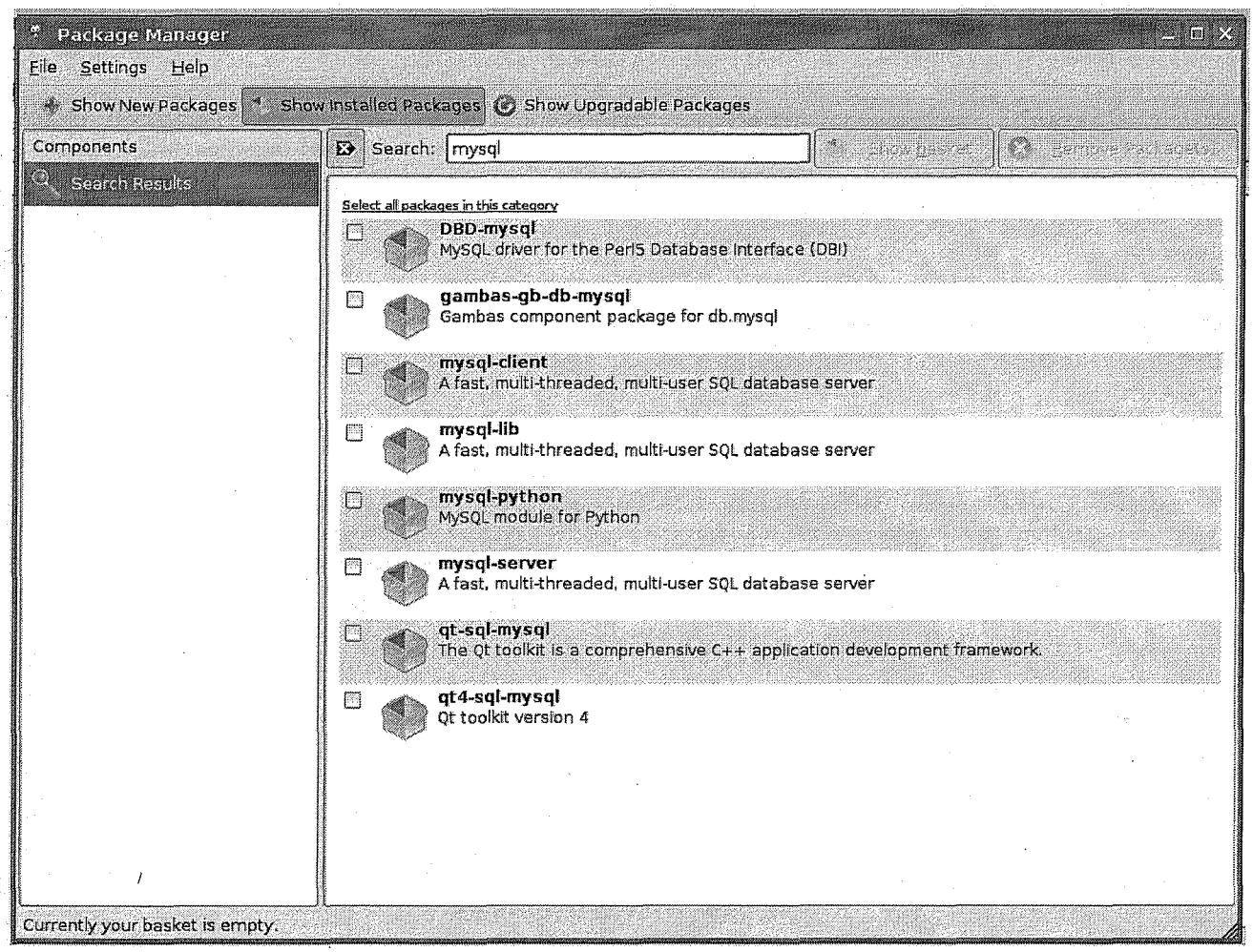

Figure 1.2: Installation of MySQL

Installing with the source code should be downloaded from web site ofMySQL.

Go to www.mysql.com > Developer Zone> Download> MySQL Community Server> 5.0 > Linux (non RPM packages) downloads> Linux (x86) > Pick a mirror and download "mysql-5. l.24-rc-linux-i686-glibc23.tar.gz" file (Shown in Figure 1.3).

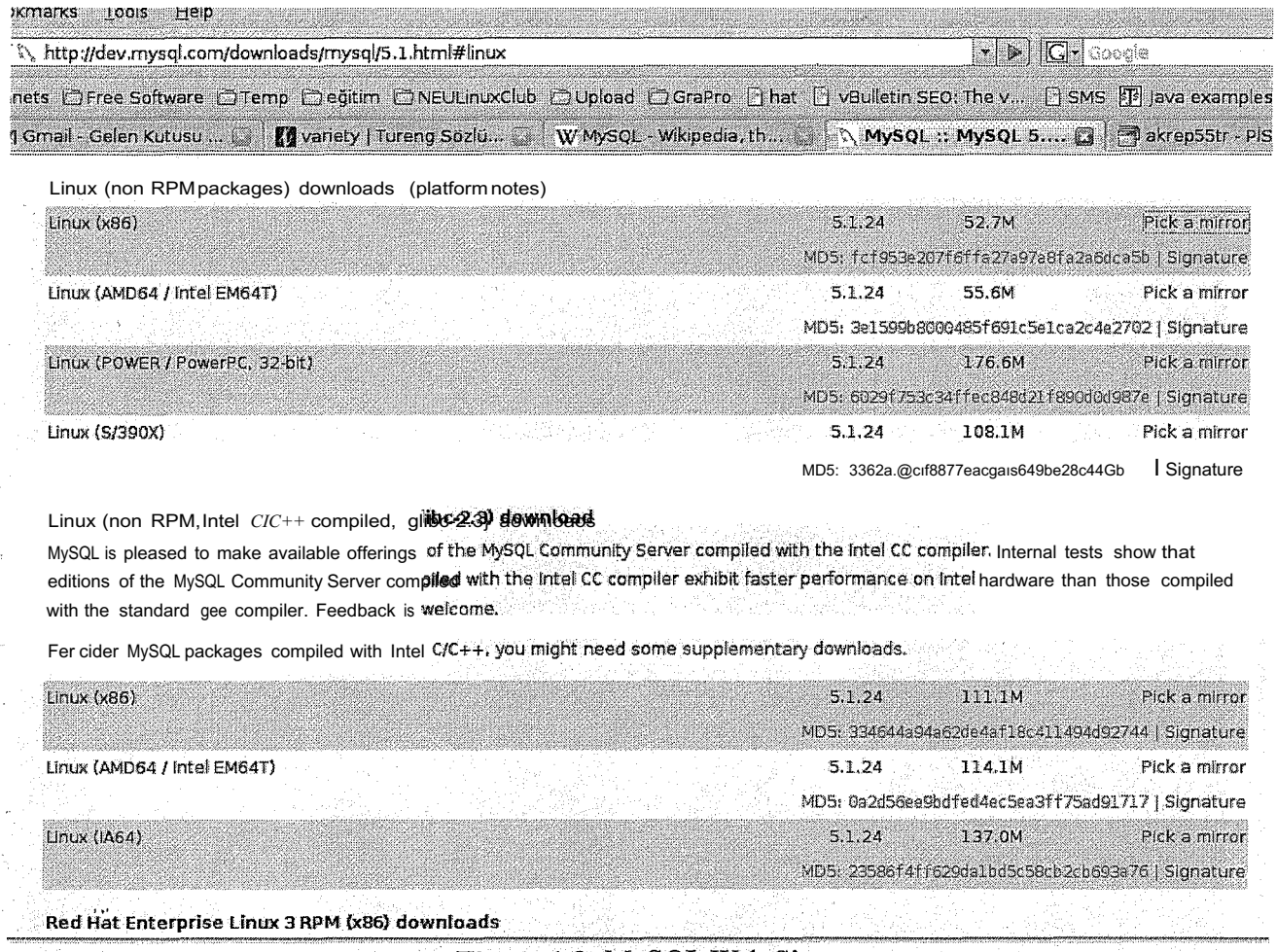

Figure 1.3: MySQL Web Site

Then downloadthe source in a directory and do those in the order.

1. Open console, became "super user" with typing "su" and enter root password.
2. tar -xvf mysql-5.l.24-rc-Iinux-i686-glibc23.tar.gz
3. cd ./mysql-5.l.24-rc-linux-i686-glibc23
4 .

2.1.2.3 Start and Stop the Service

There are two different ways of start and stop MySQL server, using "service manager" and console.

Start the mysql-server on console,

1. Open console, became "super user" with typing "su" and enter root password (Shown Figure 1.4).
2. service mysql-server start(Shown Figure 1.4).

Start the mysql-server on console,

1. Open console, became "super user" with typing "su" and enter root password(Shown Figure 1.4).
2. service mysql-server stop(Shown Figure 1.4).

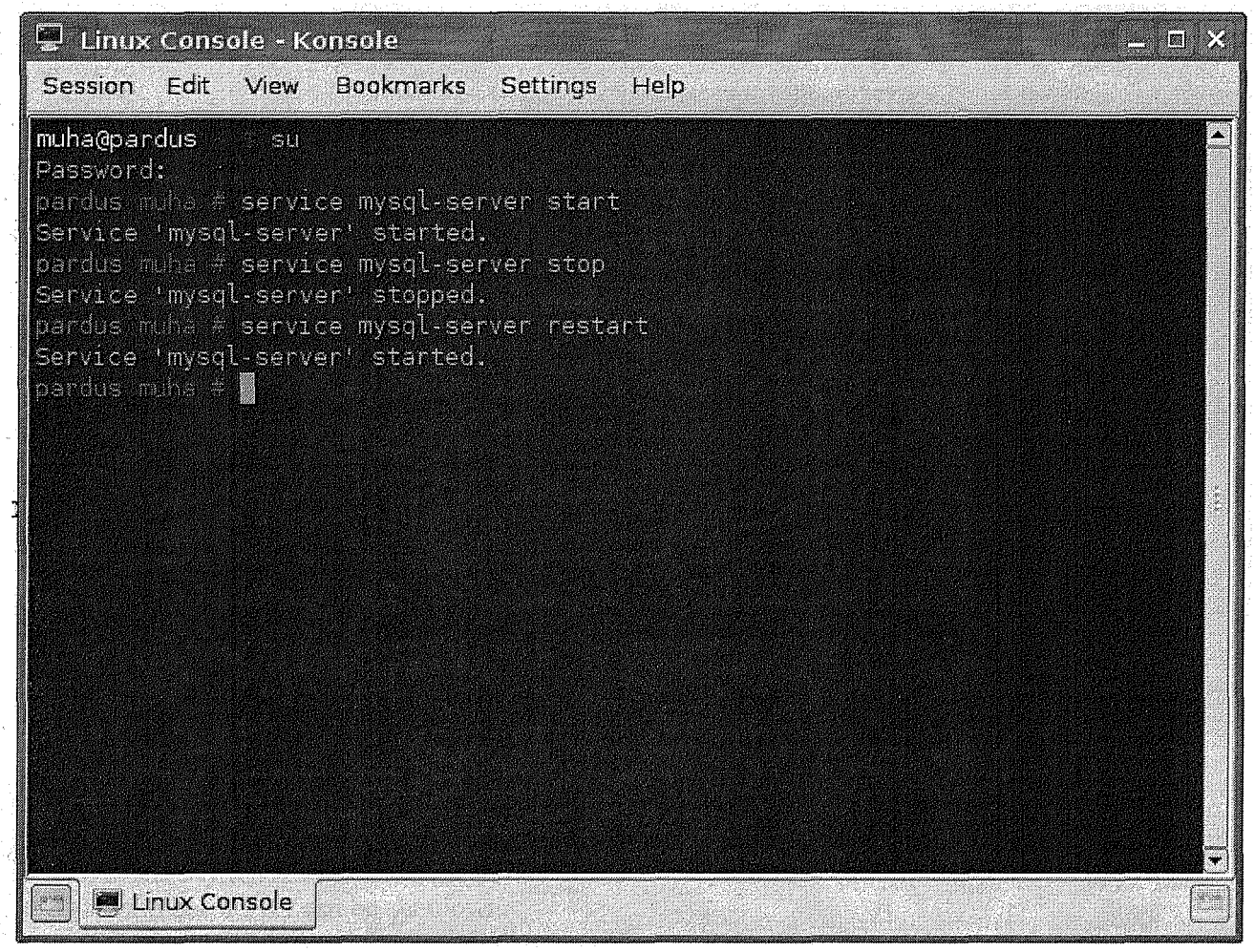

Figure 1.4: MySQL Server

Start, stop or restart mysql-server,

1. Pardus> Tasma (Pardus Configuration Center)> System> Service Manager
2. Select "MySQLI)atabase Server
3. Then click Start, Stop or Restart which one is needed (Shown in Figure 1.5).

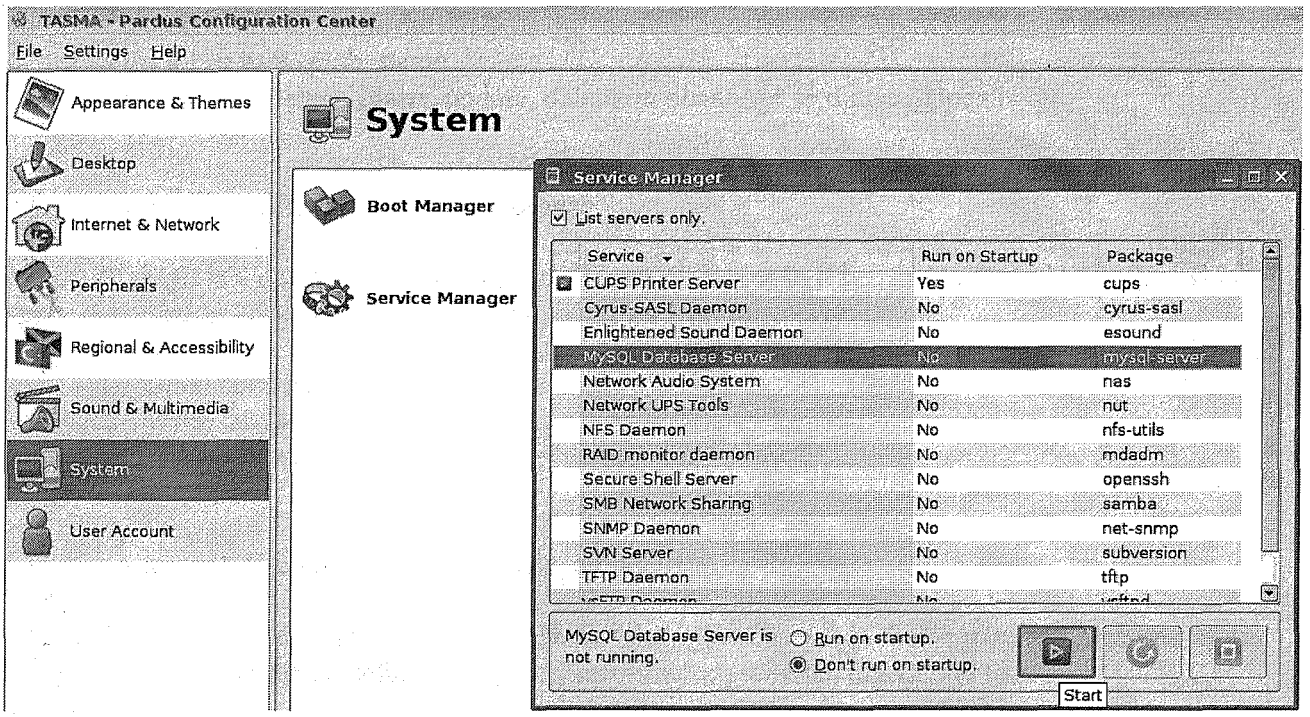

Figure 1.5- Service Manager

2.1.2.4 Setting **.ROOT** Password

MySQLhas a useful program is named as Mysqladmin for controlling MySQL actions and admissions. Mysqladmin can be used for several works, setting password, changing password, shutdown the server for updates and determining of the status of the server while it's running.

With Mysqladmin, the operator can perform many administrative functions without having to enter into the MySQL Command Line (CLI). For example, database can be created and dropped via Mysqladmin and server can be shutdown.

There is no password for the initial of "root" user. For setting it,

$ mysqladmin -u root password 'new password'

can be done, if there were a password before,

$ mysqladmin -u root -p 'new password'

will be set the password on the console.

Choose a password carefully . Password based on dictionary words. Which are guessable are not acceptable for most environments. For better data protection, choose a password that is at least six characters and includes non-alphanumeric characters.

For shutting down the server, use this command,

$ mysqladmin -p shutdown

2.1.2.5 Create MySQL User

After installation, there is a default user who is named as "root" and password is empty in MySQL. But there will be needed more users for multi tasks .

For example, a user created for accessing product information for a Web site can be given read access to the product name and price columns in the inventory database's product table. User can access with user name and password and could only read the data and not modify it.

Connect to the dbname database as the the root user from the MySQL CLI, as follows:
$ mysql -u root -p db_name
mysql> grant select, insert, update, delete on dbİıame.* to dbuser@localhost;

2.1.2.6 Create and Showing Database

MySQL is a database server program. There are tables in it but tables should be collected in a space is named as "database". Creating database, go with this order:

1. Open console and became super user.
2. Write "mysql" to console (Now, it is MySQL program).
3. Write "Create database database_name;"

$ mysql
mysql> Create database database , name;

For showing databases, write MySQL to "show databases;" In the beginnig ofMySQL Installation, there are three default databases, information_ schema, mysql and. test.

2.1.2.7 Create Table

1. Open MySQL
2. Write "show databases;"
3. Connect to database in the list writing "connect database_name;"
4. Write "CREATE TABLE table_name(i int, c varchar(5));"

Created a table is named as "table_name" that has "i" integer, "c" variable character.

Showing tabİes, write to table "show tables;"

2.1.2.8 Make a Query

Now, there are database and tables. After connecting database, we can use SQL language that we need,

1. Write to MySQL "Select * from table_ name;"
2. or inserting a row, write "insert into table_name values(3,'a');"

2.1.3 Programming Language

The Project's programming language should have these criteria:
1) Should be fast on network applications.
2) May be develop able.
3) Portable
4) Should be compatible new technologies and supported to object oriented programming.
5) Should not be any dependences.
6) Should run on internet explorer.

Java is a new technology in programming and can be run in mobile phones, computers and so many devices. It has support web programming, computer programming and programming for mobile applications. It is used very common on this devices. Most web-applications coding by Java technologies. Some examples are: online chat, games, security works and so many applets. JSP technology is used in this project. It is also open source project.

2.1.3.1 Java 'Iechnology

Java refers to a numberof computer ~ofüv~teproductsand specifications from sun Microsystems that together provide a systemforaeveloping appfication software and deployingit in a cross-platform environment. Writing in the.JavaprogranıniingJanguage is the primary way to produce code that will be deployed as Java byte code. The.success.of Javaand itswrite once, run anywhere concept.

The platform is not specific to ally örie processor Or operating system, but.there is an execution engine (called a virtual machine) and a compiler with a set of standard libraries that are implemented for different hardware and operating systems so that Java programs can run identically on all of them.

There are three different editions of the platforms are available. There are Java ME, Java SE and Java EE.

Java ME (Micro Edition), specifies several different sets of libraries (known as profiles) for devices which are sufficiently limited that supplying the full set of Java libraries would take up unacceptably large amounts of storage.

Java SE (Standard Edition), For generalpurpose use on desktop PCs, servers and similar devices

Java EE (Enterprise Edition), Java SE plus various API useful for multi-tier client-server enterprise applications.

The current version of the Java Platform is specified as either 1.6.0 or 6 (both refer to the same version). Version 6 is the product version, while 1.6.0 is the developer version.

The Java Platform consists of several programs, each of which provides a distinct portion of its overall capabilities. For example, the Java compiler, which converts Java source code into Java bytecode (an intermediate language for the Java Virtual Machine (JVM)), is provided as part of the Java Development Kit (JDK). The Java Runtime Environment (JRE), complementing the JVM with a just in-time (JIT) compiler, converts intermediate bytecode into native machine code on the fly. Also supplied are extensive libraries (pre-compiled into Java bytecode) containing reusable code, as well as numerous ways to deploy Java applications, including embedding them in a web page as an applet.

There are several other components, some available only in certain editions. The essential components in the platform are the Java language compiler, the libraries, and the runtime environment in which Java intermediate bytecode "executes" according to the rules laid out in the virtual machine specification (Shown in Figure 1.6).

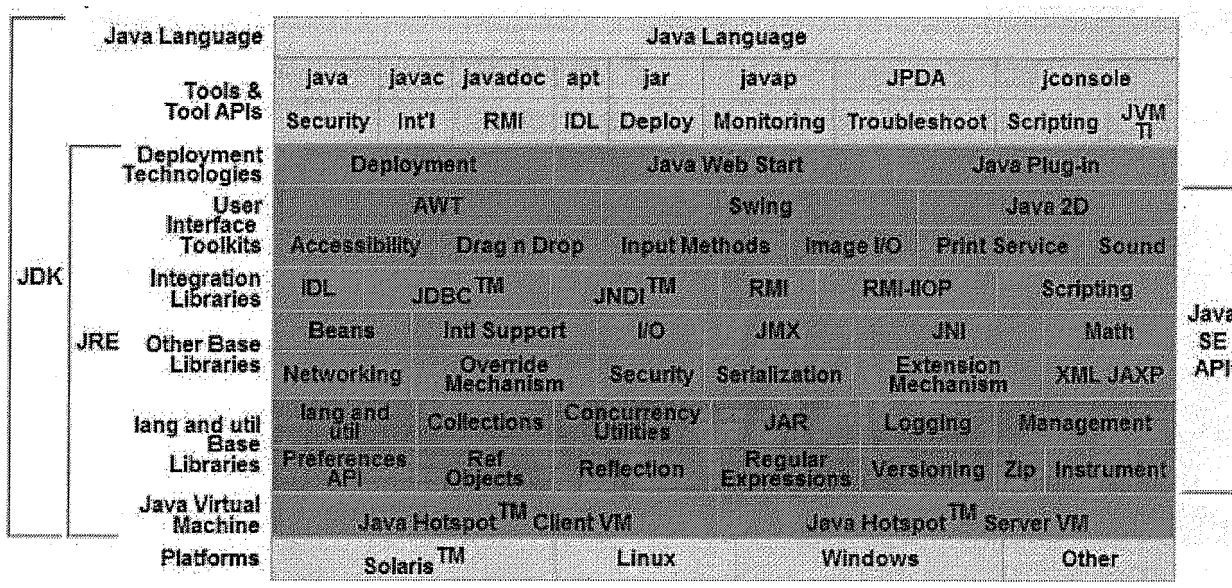

Figure 1.6 - Java Platform

Java Virtual Machine is the heart of the Java Platform is the conçept of a "virtual machine" that executes Java byte code programs. This byte code is the same no matter what hardware or operating system the program is running under. There is aJIT compiler within the Java Virtual Machine, or JVM. The JIT compiler translates the Java byte code into native processor instructions at run-time and caches the native code in memory during execution.

Java class libraries, in most modem operating systems, a large body ofreusable code is provided to simplify the ptôgtammer's job. This code is typically provided a~ a set of dynamically loadable libraries that applications can call at runtime. Because the Java Platform is not dependent on any specific operating system, applications cannot rely on any of the pre-existing OS libraries. Instead, the Java Platform provides a comprehensive set of its own·standard class libraries containing much of the same reusable functions commonly found in modem operating systems.

The Java Developmeiit'Kit (JDK) is a Sun product aimed at Java developers. It has been by far the most widely used Java SDK. It contains a Java compiler and a number of other important development tools as well as a full copy of the Java Runtime Environment.

2.1.3.1.1 Java Tools Installation

For compiling Java files, the operating system needs JDK (Java Development Kit) and JRE (Java Runtime Environment). Those two things support for Java needs.

There are also two different methods of installation.

1. Open Web Browser and go to "http://java.sun.com"
2. Go to "Downloads" for menu.
3. Go to "Java SE" (Shown in Figure 1.7)

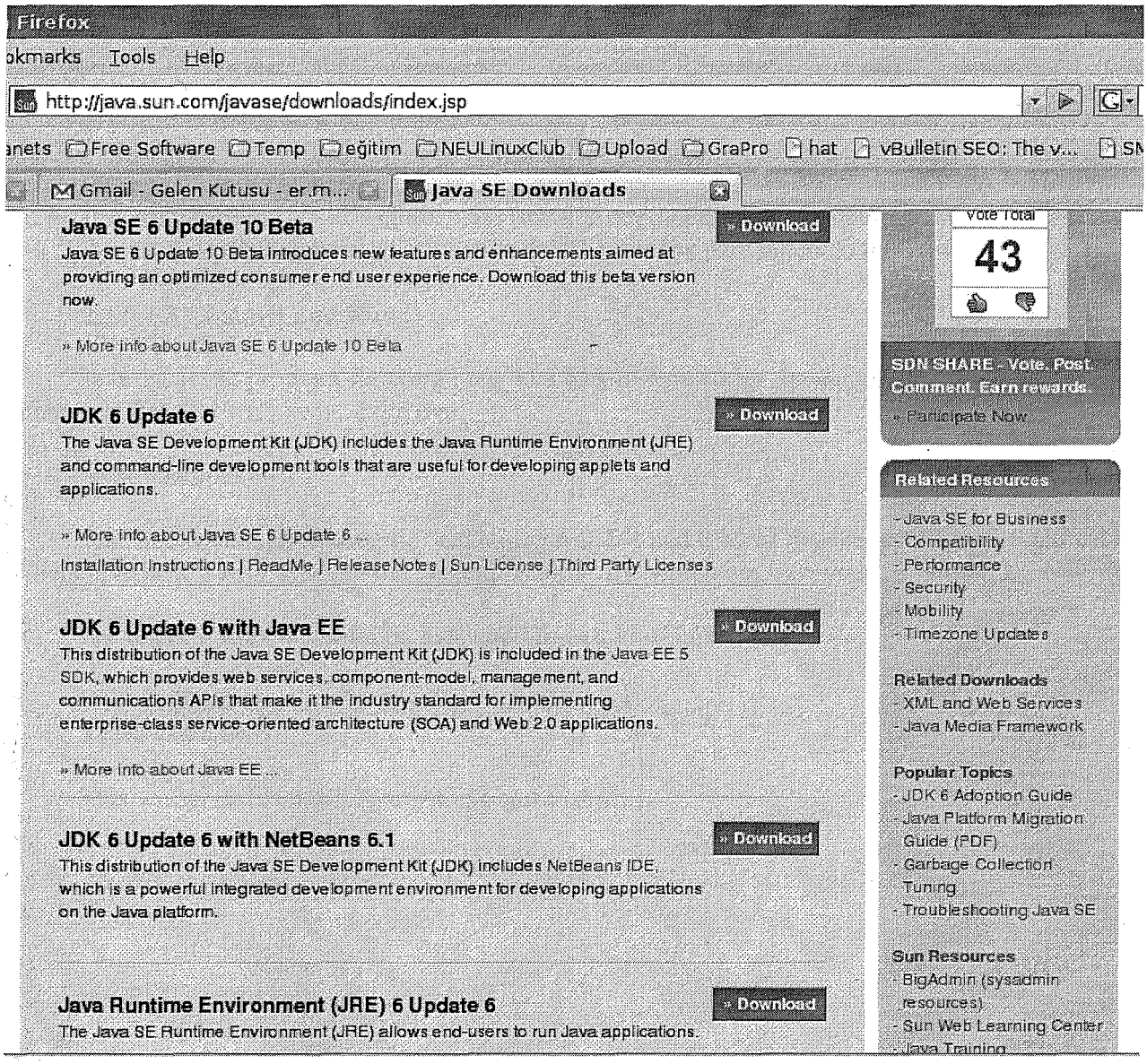

Figure 1.7 -Java Web Site

4. There are JDK and JRE which are ready to install.

Easy way to installation is:
Pardus > Package Manager
Write JDKand JRE then mark them and click "install package(s)".

2.1.3.1.2 Write a Java program

1. Open a text editor.
2. Write this codes

```
public class example{
    public static void main(String argc[]){
            System.out.println("Student Management System");
```

```
        }
    }
```

3. And save file as example java

File name should be same as main class name.
If it is a program should include main function and main function should be static and void.
System.out.println() is a static function that is write the parameters to the screen.

2.1.3.1.3 Compile and Run

Javausejava.c.progr~ntforcompiliıgjava.filestöbytecodes.Javacptograföcompiles parameters that sentwithjavac. Like that,

javac example.java

This line is enough to compile "example.java" file and it build byte codes of the file and named as "example.java". And it brigns out "example.class" file which has byte code of "exampleJava"

For running files, it should be byte code format like "example.class".

Write konsole this line fo; running "example.class":
$ java example.

This will give this result (Shown in Figure 1.8).

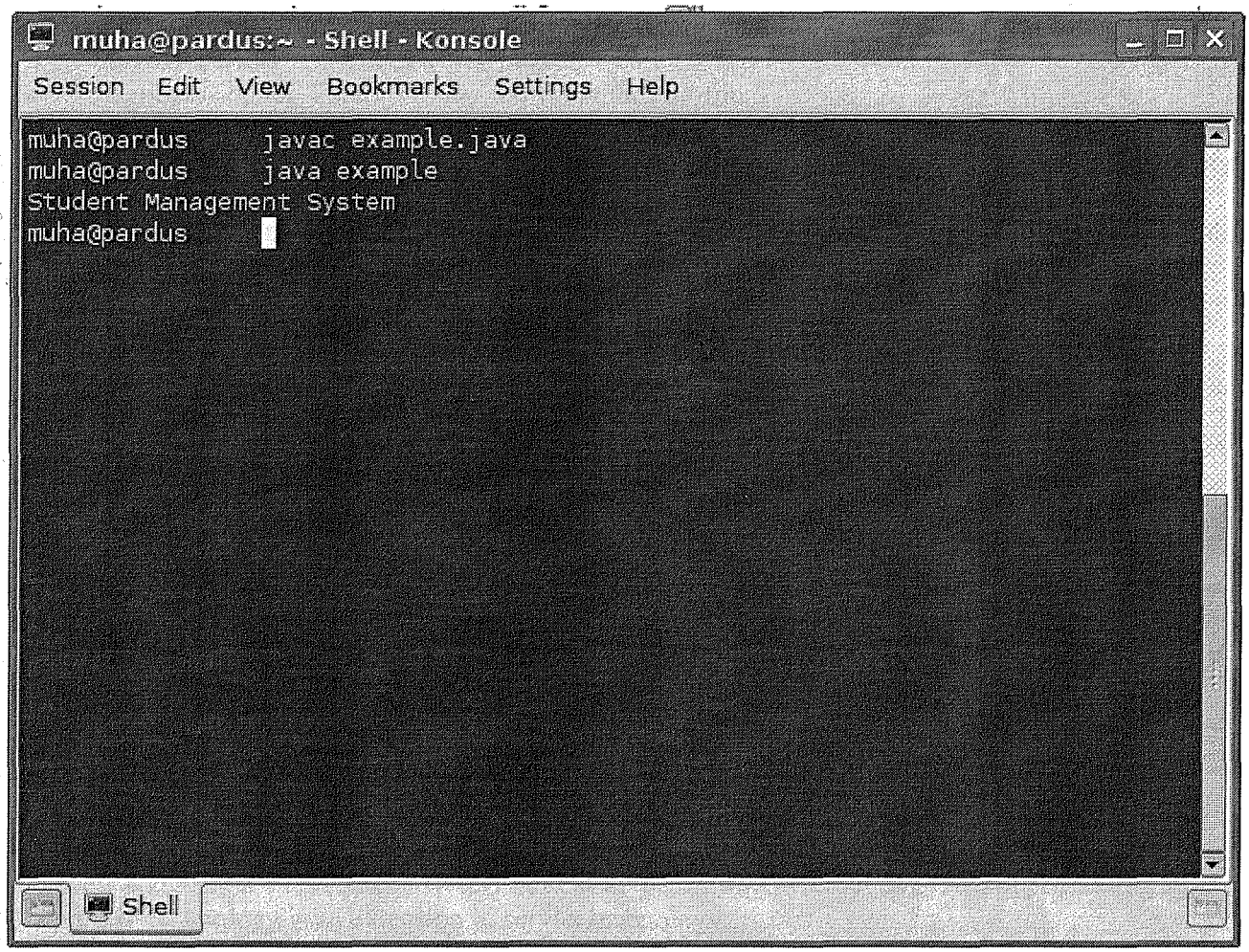

Figure 2.8: Compile Java Codes

2.1.3.2 JSP

Java Server Pages (JSP) is a Java technology that allows software developers to dynamically generate HTML, XML or other types. The technology allows Java code actions in JSP files.

JSPs are compiled into Java Servlets by a JSP compiler. A JSP compiler may generate a servlet in Java code that is then compiled by the Java compiler, or it may generate byte code for the servlet directly.

A Java Server Page may be broken down into the following pieces:

- static data such as HTML
- JSP directives such as the include directive
- JSP scripting elements and variables
- JSP actions
- custom tags with correct library

JSP directives control how the JSP compiler generates the servlet. The following directives are available:

include: for including the other files into the that file.
page: for using features of a web page.
taglib: Using tag libraries (class, JARs) in the JSP file.

JSP standard tag library: The Java Server Pages Standard Tag Library (JSTL), is a component of the Java EE Web application development platform. It extends the JSP specification by adding a tag library of JSP tags for common tasks, such as XML data processing, conditional execution, loops and internationalization.

The Java Platform has two different but complementary technologies for producing dynamic web content in the presentation tier, namely Java Servlet and Java Server Pages (JSP).

Java Servlet, the first of these technologies to appear, was initially described as extensions to a web server for producing dynamic web content. JSP, on the other hand, is a newer technology but is equally capable of generating the same d)ffliirriic content. However, the way in which a servlet and a JSP page produce their content ig fundamentally different; servlets embed content into logic, whereas JSP pages embed logic into content.

2.1.3.2.1.JSP files

A JSP page is simply a regular text file that contains markup (usually HTML) suitable for display inside a browser. Within this markup are special JSP elements.

In JSP terms, any markup that is not a JSP element is known as template text, and this really can be .any form of text-based content such as HTML, WML, XML, or even plain text. The mixture of JSP elements and template text cannot simply be sent to the browser without any form of processing by the server. JSP technology is an extension of servlet technology.

Figure 2.9 shows a JSP page being-translated and compiled into a servlet in response to a request. This servlet is known as the JSP implementation servlet.

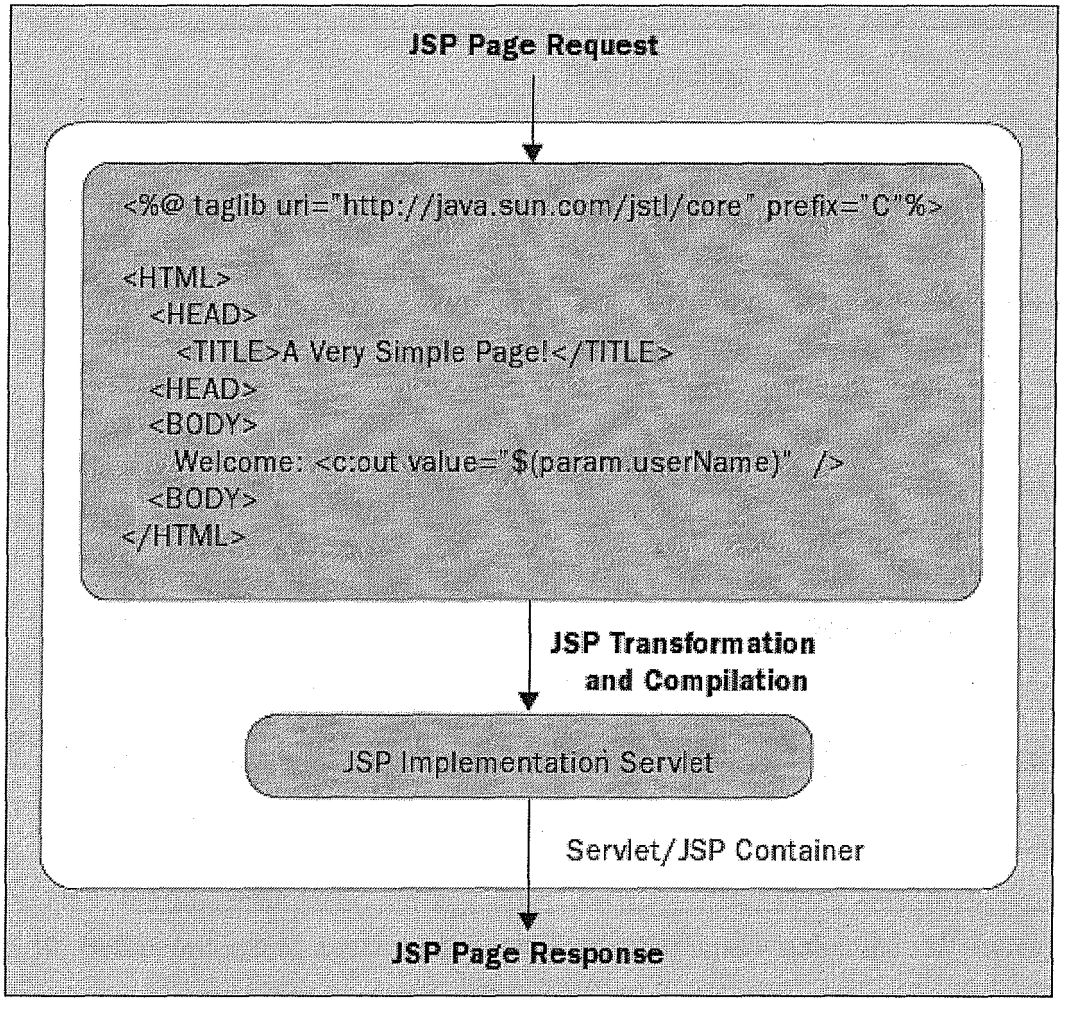

Figure 2.9: The JSP container translates and compiles the JSP source

JSP pages do not directly return content to the client browser themselves. Instead, they rely on some initial server-side processing that converts the JSP page into the JSP page implementation class (Shown Figure 2.10), which handless all requests made of the JSP.

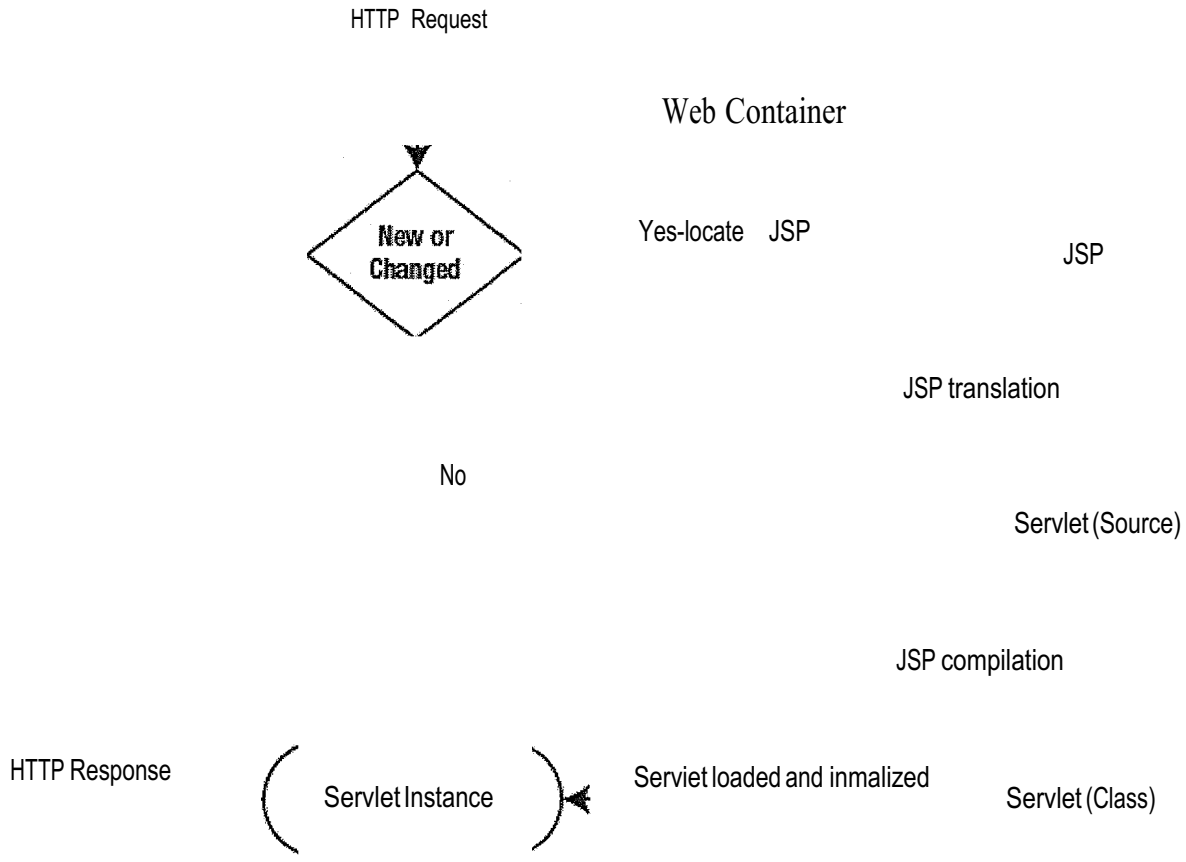

Figure 2.10: Before processing a request, the container determines whether the JSP source is new or has changed.

As shown in Figure 2.10, the JSP servlet container decides whether the JSP page has been translated before. The first stage in the life cycle of a JSP page is known as the translation phase. When a request is first made for a JSP page, the JSP engine will examine the JSP file to check that it is correctly formed and that the JSP syntax is correct. If the syntax check is successful, the JSP engine will translate the JSP page into. its page implementation class, which takes the form of a standard Java Servlet. After the page's implementation servlet has been created, it will be compiled into a class file by the JSP engine and will be ready for use.

Each time a container receives a request, it first checks whether the JSP file has changed since it was last translated. If It has, it's retranslated so that the response is always generated by the most up to date implementation of the JSP file.

After the translation phase has been completed, the JSP engine will need to load the generated class file and create an instance of the servlet in order to continue processing the initial request. Therefore, the JSP engine works very closely with the servlet container and the JSP page implementation servlet and will typically load a single instance of the servlet into memory. This single instance will be used to service all requests for the JSP page. In a real-world web application, those requests will most likely happen concurrently, so your JSP page must be multithreaded.

After the web container has loaded and initialized the implementation servlet, the initial request can be serviced. To service the request, the web container calls the j spService() method of the

implementation servlet. As we mentioned, each request to the JSP page results in a separately threaded call to the jspService() method.

The jspService() method provides all the functionality for handling a request and returning a response to the client. All the scriptlets and expressions end up inside this method, in the order in which they were declared inside the JSP page. Notice that JSP declarations and directives aren't included inside this method because they apply to the entire page, not just to a single request, and therefore exist outside the method. The jspService() method may not be overridden in the JSP page.

The last phase in the life cycle is the finalization phase. As with the previous two phases, there is a corresponding method in the implementation servlet for this phase. The method is named jspDestroy(). Like the destroy() method found in a standard servlet, this method is called by the servlet container when the page implementation servlet is about to be destroyed. This destruction could be for various reasons, such as the server being low on memory and wanting to free up some resources, but the most common reason is that the servlet container is shutting down or being restarted.

After this method has been called, the servlet can no longer serve any requests. Like the destroy() method, jspDestroy() is an excellent place to release or close application-level resources when the servlet is shutting down.To do this, simply provide an implementation of this method via a JSP method declaration. For example, to release the application resource you opened inside the jsplnit() method, you would use the following:

```
<%!
  public void jspDestroy() {
    try {
      appVar.release();
    } catch (Exception e){}
    appVar = null;
  }
%>
```

For the security, by providing an initial single point of access for potential requests, a servlet-based controller component is an excellent place to provide some form of authentication. If the user making the request can pass the authentication mechanism (perhaps a user name or password test), the controller can continue with the request as usual or alternatively forward it to an appropriate page (perhaps a login page!) where the error can be dealt with.

Because the controller component is responsible for handling every request, security checks have to exist in only a single place, and of course any changes to the security mechanism have to be made only once. By implementing your security constraints in a single place, it's far easier to take advantage of declarative security mechanisms. Each page to provide similar security checks by itself, which provides a significant security hole if the developer forgets to provide it.

2.1.3.2.2 Use JSP codes

JSP files is like HTML files. It has all features of HTML.

```
<html>
<head></head>
<body>
```

```
</body>
</html>
```

Also tags are used in JSP,

```
<html>
<head></head>
<body>
<jsp:include page="page2.jsp" />
</body>
</html>
```

For forwarding web page,

```
<jsp:forward page="subpage.jsp" />
```

There are a lot of tag libraries and its features for controlling web applications. This technology permits using common variables in a page to other pages with "session".

2.1.3.2.3.Use Java codes in JSP

For using Java code, it should be start with"<%" and end this with"%>".

```
<%
out.println("Student Management System");
%>
```

Using HTML codes and Java codes in JSP may like this,

```
<html>
<head></head>
<body>
<%
out.println("Student Management System");
%>
</body>
</html>
```

There are a lot of techniques at using Java codes in JSP.

For example getting requests,

```
<%
String r = (String)req-uest.getParameter("param");
out.println(r);
%>
```

2.1.4 Server Software

There are two common server software for serving JSP files, Apache Tomcat and JBoss. Tomcat is used in the project because it is more usable with the IDE.

The Java Servlet API allows a software developer to add dynamic content to a Web server using the Java platform. The generated content is commonly HTML, but may be other data such as XML. 'Servlets are the Java counterpart to non-Java dynamic Web content technologies such as PHP, CGI and

ASP.NET. Servlets can maintain state across many server transactions by using HTTP cookies, session variables or URL rewriting.

The ServletAPI, contained in the Java package hierarchy javax.servlet, defines the expected interactions of a Web container and a servlet. A Web container is essentially the component of a Web server that interacts with the servlets. The Web container is responsible for managing the life cycle of servlets, mapping a URL to a particular servlet and ensuring that the URL requester has the correct access rights.

A Servlet is an object that receives a request and generates a response based on that request. The basic servlet package defines Java objects to represent servlet requests and responses, as well as objects to reflect the servlet's configuration parameters and execution environment. The package javax.servlet.http defines HTTP-specific subclasses of the generic servlet elements, including session management objects that track multiple requests and responses between the Web server and a client. Servlets may be packaged in a WAR file as a Web application.

A Servlet container is a specialized web server that supports Servlet execution. It combines the basic functionality of a web server with certain Java/Servlet specific optimizations and extensions - such as an integrated Java runtime environment, and the ability to automatically translate specific URLs into Servlet requests. Individual Servlets are registered with a Servlet container, providing the container with information about what functionality they provide, and what URL or other resource locater they will use to Identify themselves. The Servlet container is then able to initialize the Servlet as necessary and deliver requests to the Servlet as they arrive. Many containers have the ability to dynamically add and remove Servlets from the system, allowing new Servlets to quickly be deployed or removed without affecting other Servlets running from the same container. Servlet containers are also referred to as web containers or web engines.

2.1.4.1 Apache Tomcat

Apache Tomcat is a Servlet container developed at the Apache Software Foundation (ASF). Tomcat implements the Java Servlet and the Java Server Pages (JSP) specifications from Sun Microsystems, and provides a "pure Java" HTTP web server environment for Java code to run. Tomcat should not be confused with.the Apache web server, which is a C implementation of a HTTP web server; these two HTTP web servers are not bundled together. Apache Tomcat includes tools for configuration and management, but can also be configured by editing configuration files that are normally XML-formatted.

An overview of the different versions can be found on the Apache website. Tomcat 3.X, Tomcat 4.X, Tomcat 5.X and Tomcat 6.X are the versions of Tomcat which are still on the website of Apache.

2.1.4.1.1 Hierarchy of Tomcat

There are seven directory and six files in the apache tomcat directory. The directories are: bin, conf, lib, logs, temp, webapps and work. The files are: License, License Tomcat.txt, notice, release-notes, running.txt and uninstall.sh.

Bin: includes binary files of tomcat such as start, stop, restart and shutdown.
Conf: includes configuration files in it.
Lib: has libraries in it for used by all web sites, such as mysql.jar, standat.jar.

Logs: collects log files when tomcat broken or some web sites done illegal works.
Temp: This is a temporary file system that use it for.
Webapps: Collects web sites in it writen in JSP or HTML formats. We will put our JSP files under it in a our directory. There is a ROOT directory which gives us direct access.
Work: Tomcat uses it for working on it.
License and License Tomcat.txt: are license files of Tomcat.
Notice: are notes of what the operating system needs ..
Release-notes: are notes of that version.
Running.txt: is a file which shows us how to use Tomcat such as start and stop functions.

2.1.4.1.2 Start and Stop

1. Open console and became super user.
2. Go into apache tomcat using "cd" command like "cd apache-tomcat-6.0.16".
3. Write "sh ./bin/startup.sh" for starting Tomcat.
4. Write "sh ./bin/shutdown.sh" for stopping Tomcat,

2.1.4.1.3 Run JSPFiles

For running JSP files, puts them under "webapps" directory.

1. Open webapps.
2. Create a directory and give its a name "example".
3. Create a file is named index.jsp and write this codes in it.

```
<html>
<head>
</head>
<body>
<%
out.println("This is a JSP file");
%>
</body>
</html>
```

4. Then start Tomcat.
5. Go to http://localhost:8080/example by web browser.
   Now, you will see theindex.jsp file in the web browser. Tomcat uses 8080 port for its application server. Forlocalhost, writing to address line 8080 is not needed but for local using it is needed.

2.1.4.2 JBoss

JBoss Application Server (or JBoss AS) is a free software / open source Java EE-based application server. Because it is Java-based, JBoss AS is cross-platform, useable on any on operating system that Java supports.

JBoss runs between the application code and the operating system to provide services. JBoss is implemented in Java allows to run on many different operating systems, giving the flexibility to develop and deploy applications wherever you like.

2.1.5 IDE for compiling Source Code

For building JSP files, only needed a text editor. Just store file with .JSP (fileName.JSP) format. When coding, sometimes needed helper for correcting errors by itself. IDEs are used for correcting errors, filling the codes and the code blocks. Most popular open source Java Ides are NetBeans and Eclipse.

NetBeans and Eclipse run on Windows, Linux, Mac OS X and Solaris because there are written with Java, can run every platform. There are open-sources, good prepared documentations and free. They are built for whose projects are focused on building an open development platform. In this project, Eclipse and NetBeans documentation were used. For the coding Eclipse is used because it is more faster then NetBeans.

2.1.5.1.NetBeans

NetBeans refers to both a platform for the development of Java desktop applications, and an integrated development environment (IDE) developed using the NetBeans Platform.

The NetBeans Platform allows applications to be developed from a set of modular software components called modules. A module is a Java archive file that contains Java classes written to interact with the NetBeans Open APis and a manifest file that identifies it as a module. Applications built on modules can be extended by adding new modules. Since modules can be developed independently, applications based on the NetBeans platform can be easily and powerfully extended by third party developers.

The NetBeans Platform is a reusable framework for simplifying the development of other desktop applications. When an application based on the NetBeans Platform is run, the platform's Main class is executed. Available modules are located, placed in an in-memory registry, and the modules' startup tasks are executed. Generally, a module's code is loaded into memory only as it is needed.

Applications can install modules dynamically. Any application can include the Update Center module to allow users of the application to download digitally-signed upgrades and new features directly into the running application. Reinstalling an upgrade or a new release does not force users to download the entire application again.

The NetBeans IDE is an open source integrated development environment written in Java using the NetBeans Platform. NetBeans IDE supports development of all Java application types (J2SE, web, EJB and mobile applications) out of the box. Among other features are an Ant-based project system, version control and refactoring. It's modularity is all the functions of the IDE are provided by modules. Each module provides a well defined function, such as support for the Java language, editing, or support for the CVS versioning system. NetBeans contains all the modules needed for Java development in a single download, allowing the user to start working immediately. Modules also allow NetBeans to be extended. New features, such as support for other programming languages, can be added by installing additional modules. For instance, Sun Studio, Sun Java Studio Enterprise, and Sun Java Studio Creator from Sun Microsystems are all based on the NetBeans IDE.

NetBeans has GUI design tools for using Swing GUis. It is much easier GUI components positioning.

NetBeans has more package, Mobile Pack for mobile applications, *CIC++* pack for *CIC++* applications, Enterprise Pack for visual design tools for UML modeling, XML schema and web services, Visual Web Pack for visual web pack considered to be a much better and Ruby Pack.

2.1.5.2.Eclipse

Eclipse is an open source community whose projects are focused on building an open development platform comprised of extensible frameworks, tools and run times for building, deploying and managing software across the life cycle.

Eclipse is an integrated development environment (IDE) written primarily in Java. In its default form it is meant for Java developers, consisting of the Java Development Tools (JDT). Users can extend its capabilities by installing plug-ins written for the Eclipse software framework, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules.

Eclipse began as an IBM Canada project. In 2003, the Eclipse Foundation was created. It is now GPL license. Most of the Linux distributions has eclipse ide in their package system. Also, it can be installed to operating system. It has more packages for compiling different language like *CIC++,* Python and more.

2.1.5.2.1.Start Eclipse

Pardus has Eclipse IDE in its development package. It can be installed using with Package Manager.

Pardus > Package Manager. Then Select Eclipse packages under Programming.

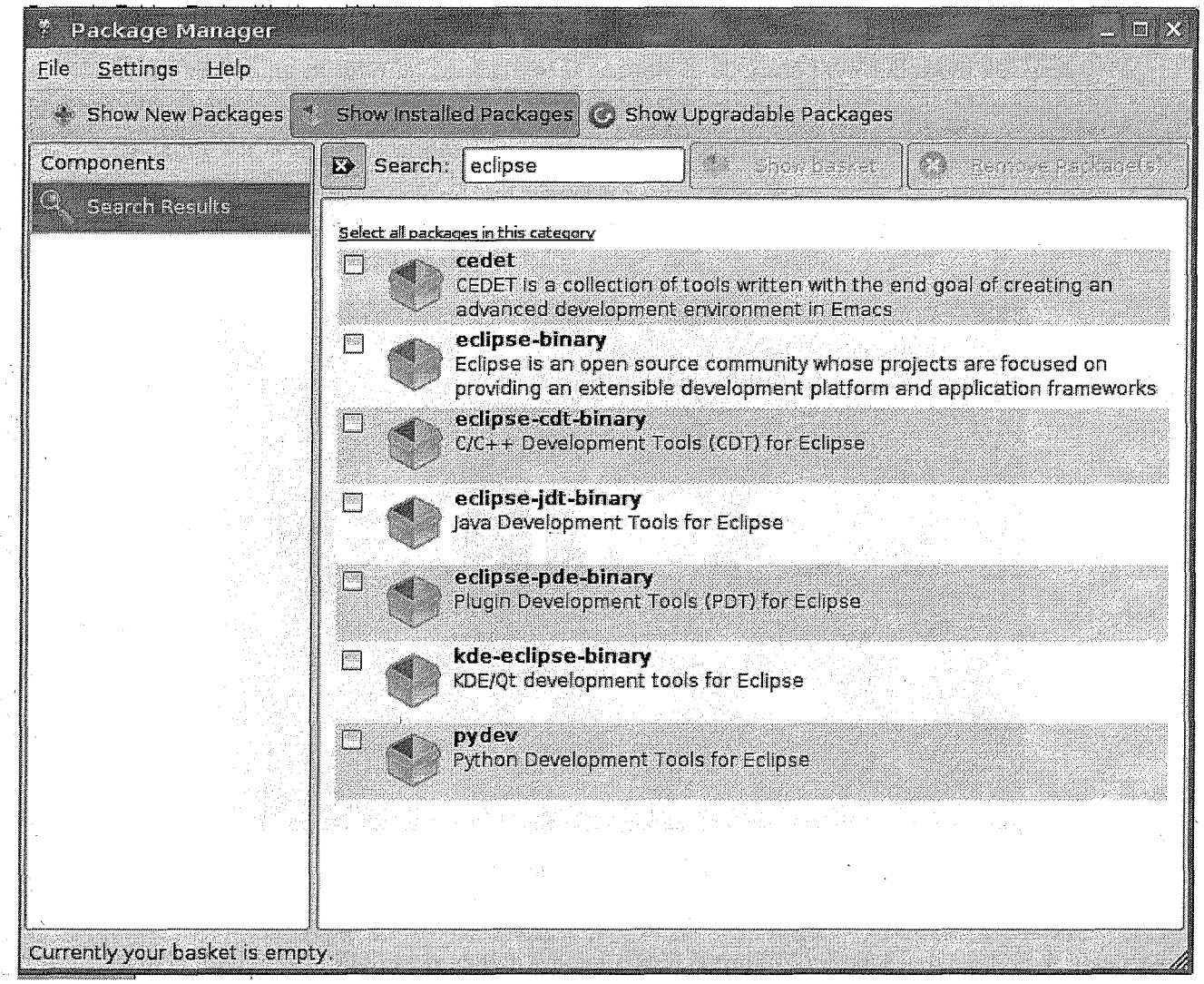

Figure 2.11: Installation of Eclipse

After installation of Eclipse, it is placed

Pardus > Programs > Development > Eclipse 3.3

2.1.5.2.2.Create a Java project

Before coding, need to create a project that is dependent to project type. For Java project can be created in it with this order.

File> New> Project> Java> Java Project

or

Click right button of mouse on Project Explorer> New Project> Java> Java Project.

These are both can be used.

There are a lot of type of project like *CIC++*, Web application, Java Project, Ruby and more.

2.1.5.2.3.Update Eclipse

Inthe beginning of installation, Eclipse is capable to compile Java codes but riôtweb applications. It needs to update to compile more languages.

Updating for Web Applications

Help > Software Update > Find and Install> Select "Search for New Features" and next> Europa Discovery Site, then Click "Finish" > Select "Mirror" which is more close to your country (I select [Turkey]Tubitak-Ulakbim(http))

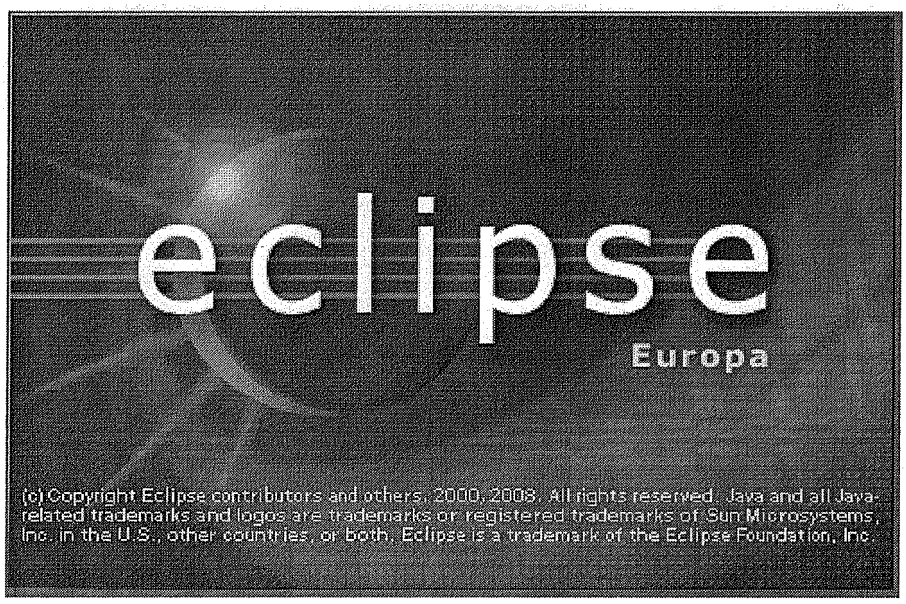

Figure 2.12: Run Eclipse

Here is the Logo of Eclipse is shown when Eclipse is run (Shown in Figure 2. 12). It also give us information of Eclipse and it's foundation. It is copyright between 2000 and 2008.

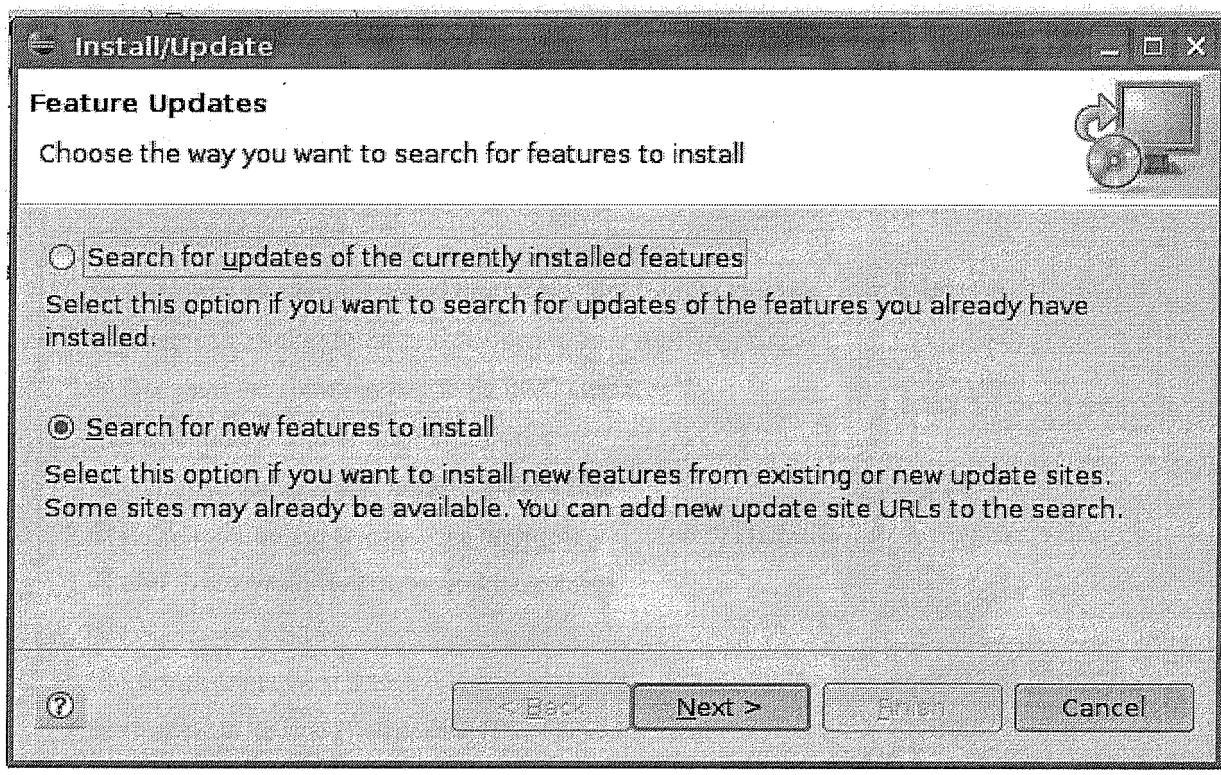

Figure 2.13: Install/Update Form

After running Eclipse, Help > Software Update, this page will come and asks add new features or update installed features. At the beginning Eclipse has no Web Application feature, so should "Search for new features to install" be selected (Shown in Figure 2.13)

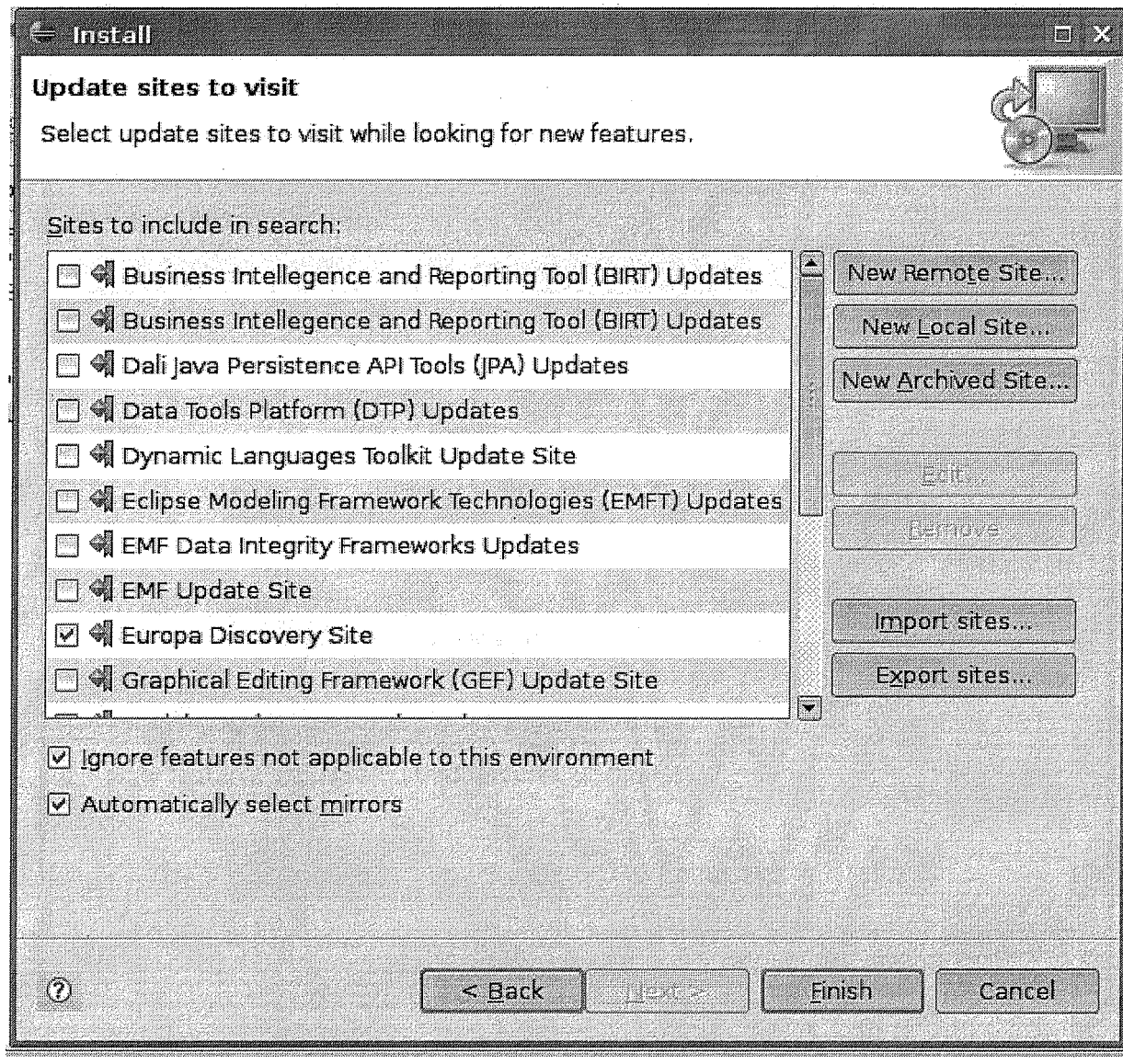

Figure 2.14: Select Server of Installation

For updating Eclipse, should chosen Europa Discovery Site in the sites because it is more nearest then the other. And mark Automatic select mirrors for leave it to select which one is best (Shown in Figure 2.14).

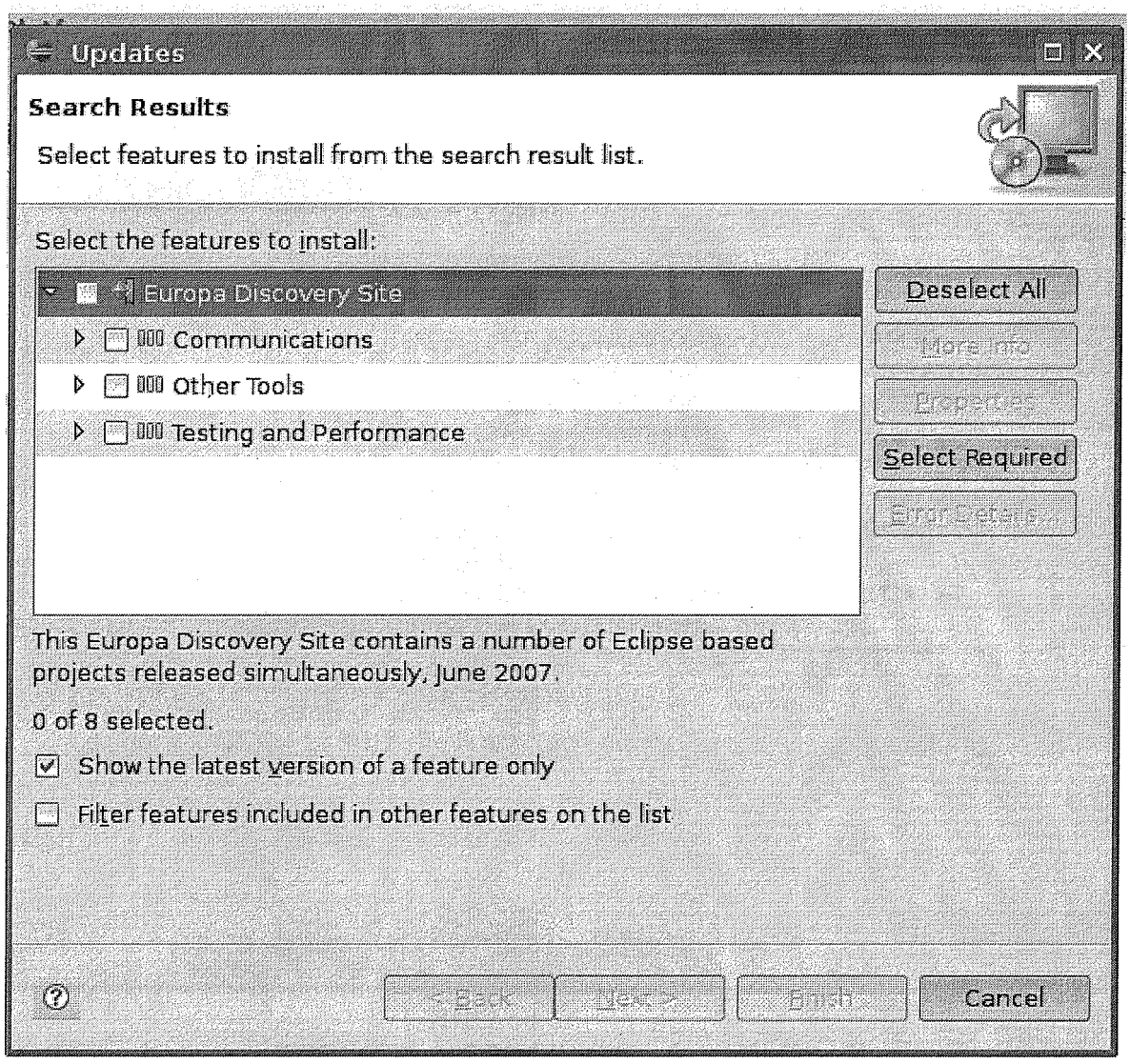

Figure 2.15: List of Available Updates

In the list, web application should be marked for Dynamic Web Projects. After selecting it, should click first, Select Required, then Finish.

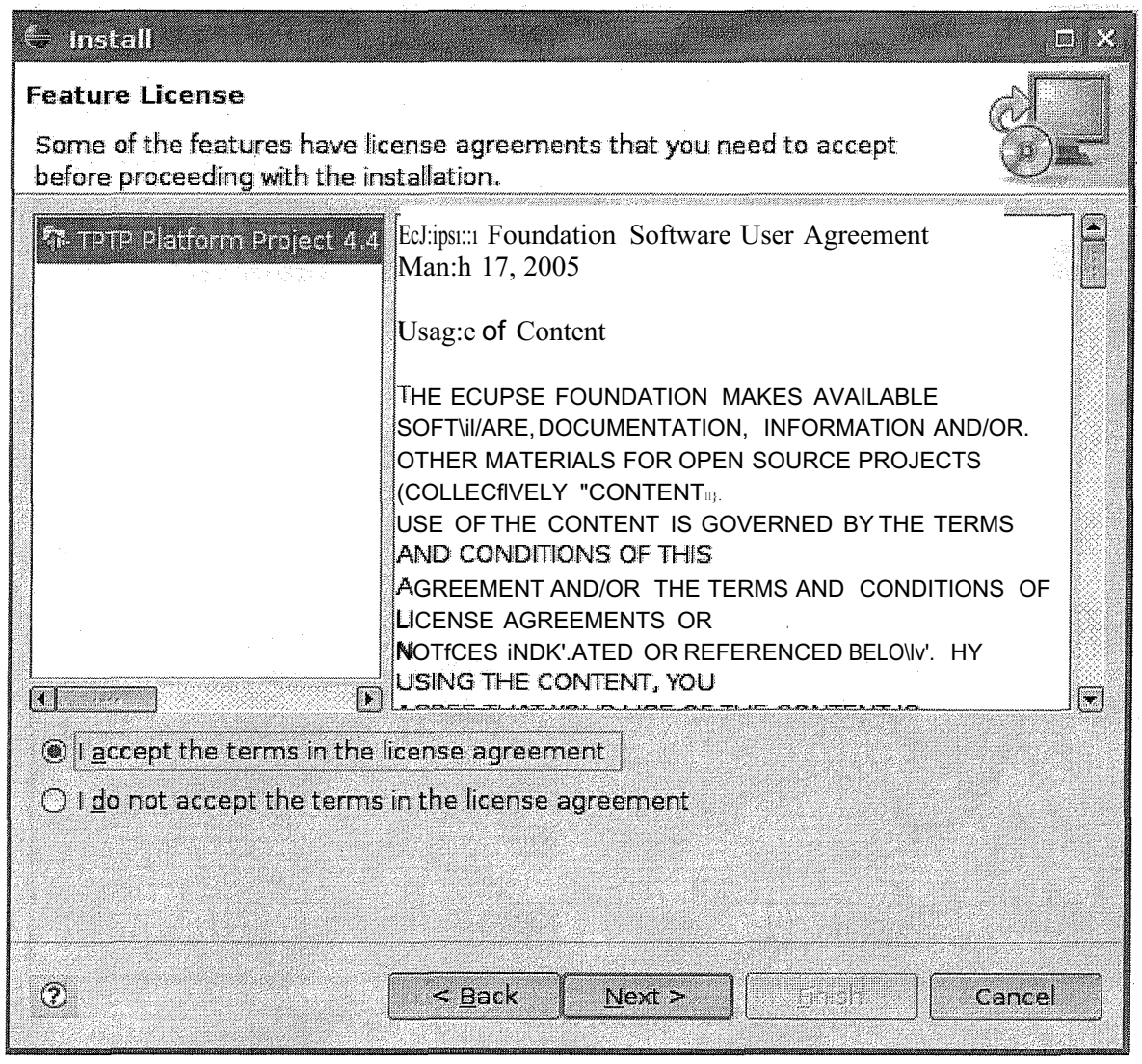

Figure 2.26: License agreements

For updating Eclipse,;xour selected packages has license that you should accept all of them. If not accept, they will not be'installed.

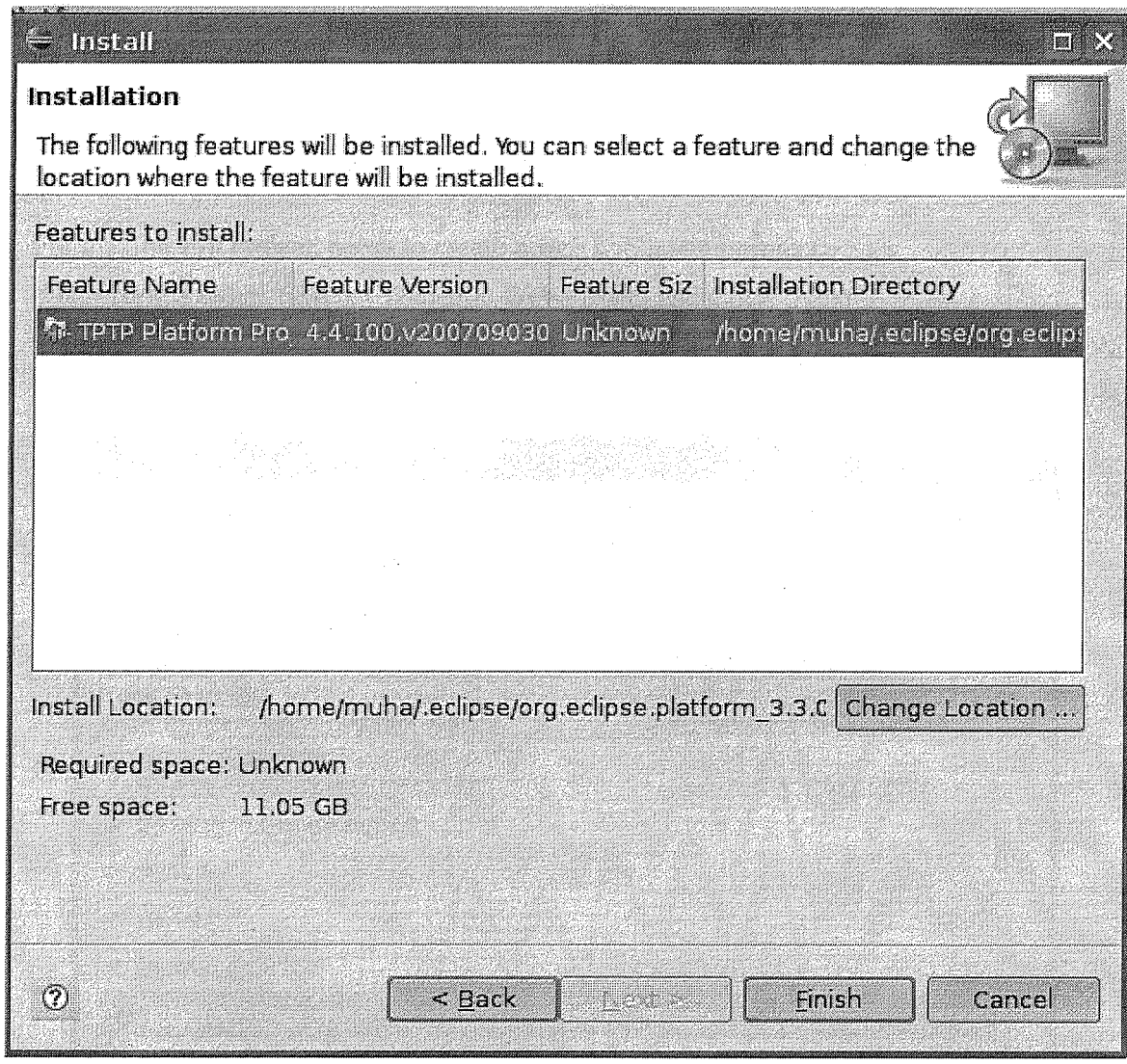

Figure 2.27: Show which package will be installed

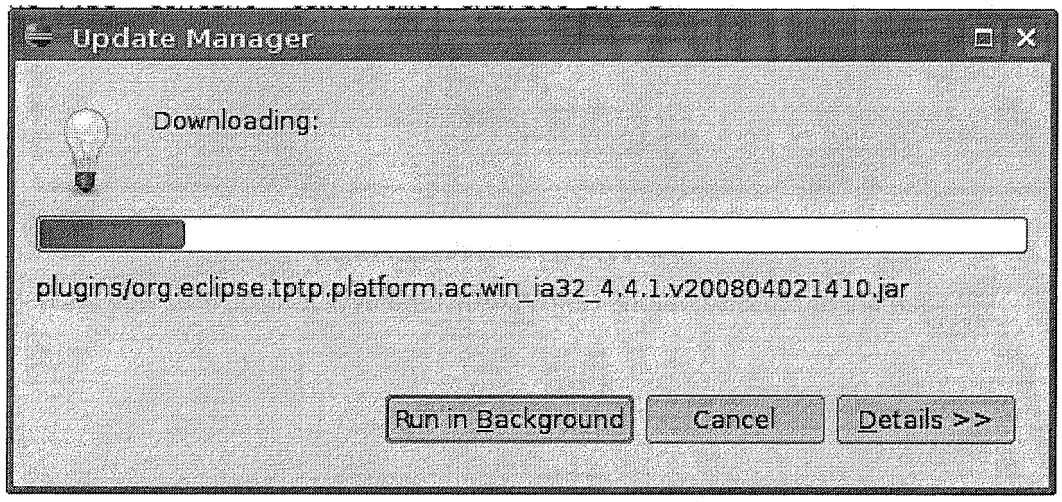

Figure 2.28: Installation

2.1.5.2.4.Crea.te a web application

For making JSP project, needs to create a web application project.

File> New> Other> Web> Dynamic Web Project and Click "Next".

Then give a project a name(it is named as "SMS" means Student Management System). And Click "Finish".

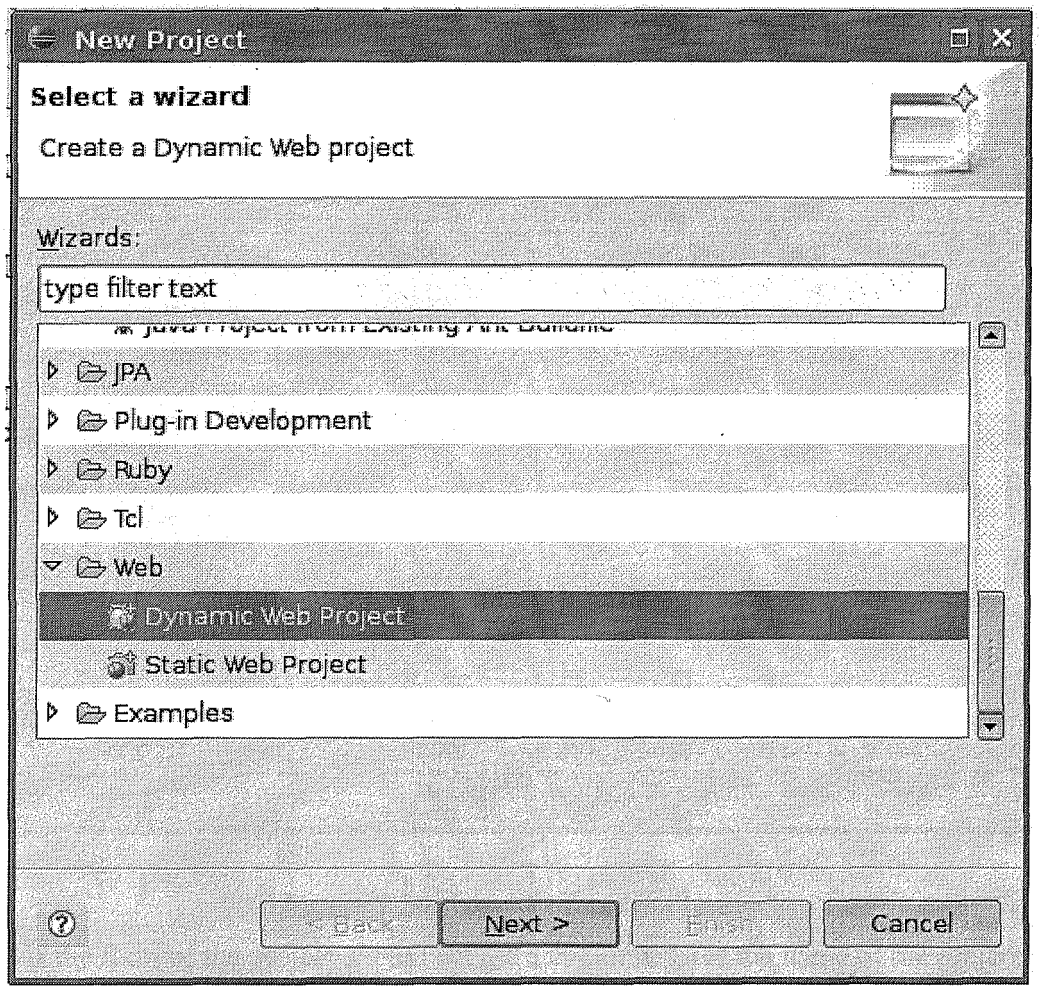

Figure 2.29: Add New Project

To create new web project, Dynamic Web Project should selected in the list.

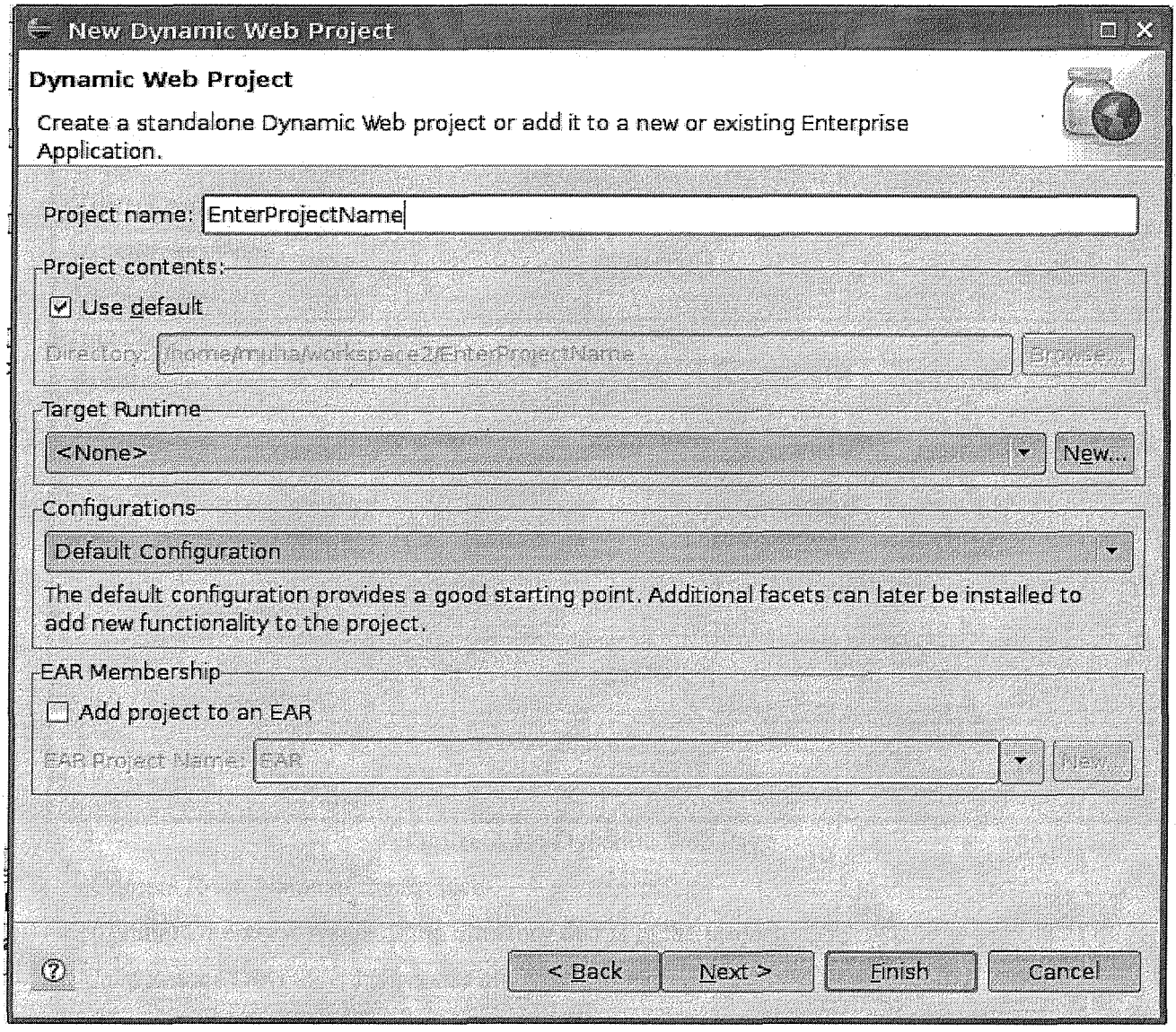

Figure 2.30: Give the Project a Name

This is the form of the project should get the name of it. Give the name of the project to "Project Name" place.

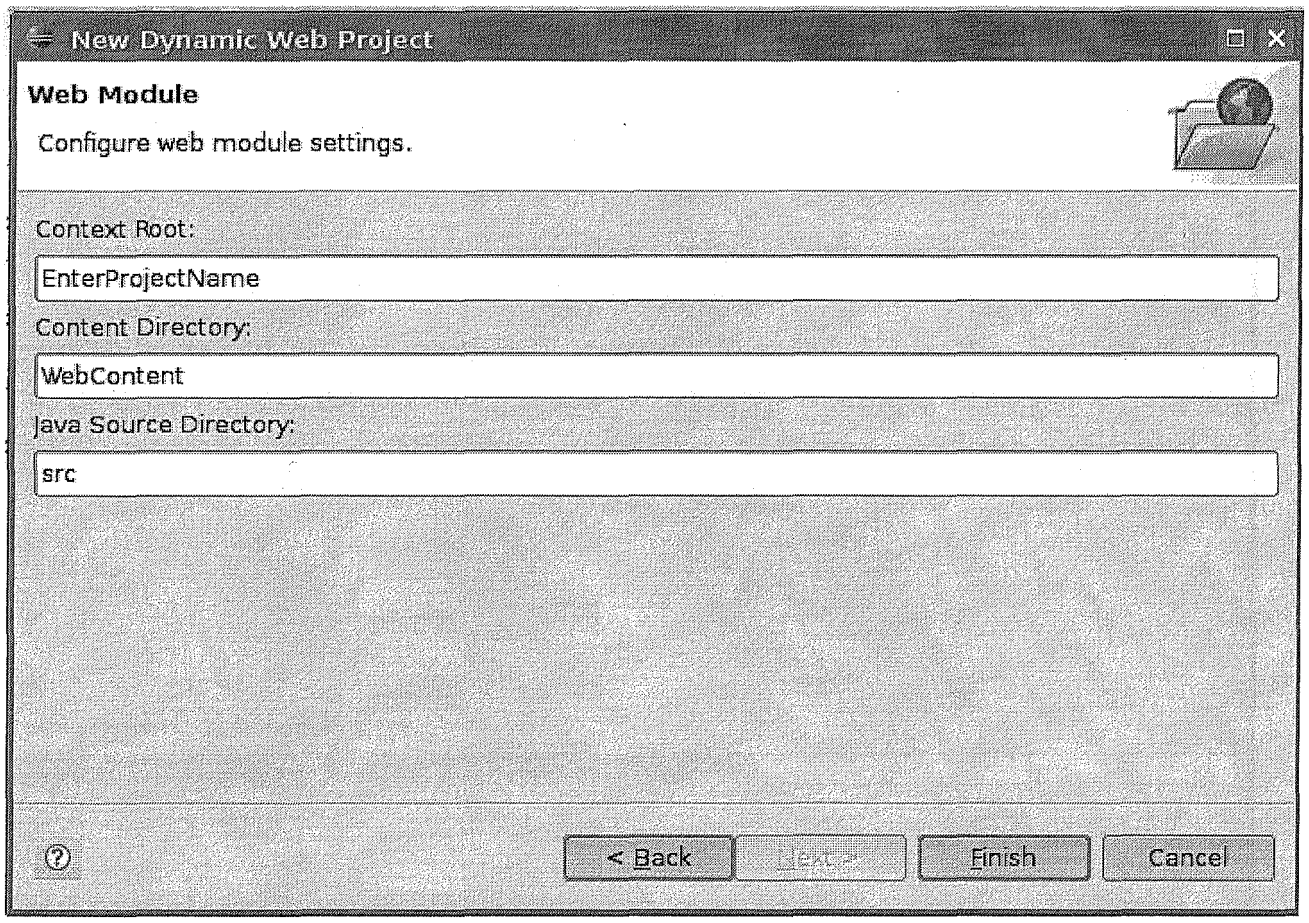

Figure 2.31: Dynamic Web Page

Content Root: Name of the Project,

Content Directory: Name of the directory that is in the project,

Java Source Directory: Java codes are where to placed.

All shown in Figure 2.31.

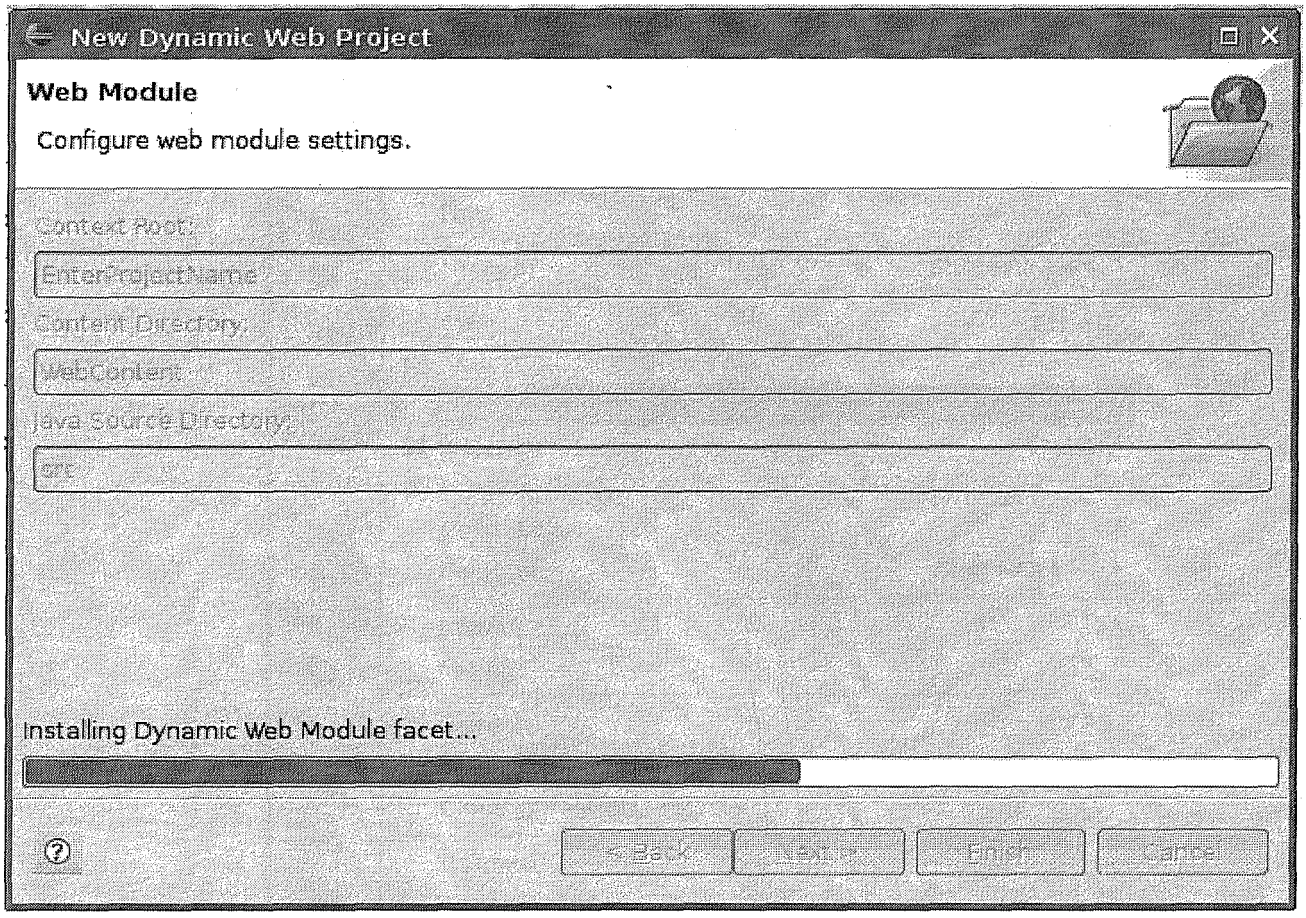

Figure 2.32: Creation of the Project

In Figure 2.32 shows us there is a problem or not in the creation time. If there is, it should show us, if there is no, it will create automatically.

2.1.5.2.5.Make Index Page in Dynamic Web Project

Select the project in the Project Manager the right click> New> Other> Web> JSP > give a name like "index" and Click"Finish".

Now, index.jsp is ready can be edited.

Edit the indexjsp with Java codes (Shown in Figure 2.33).

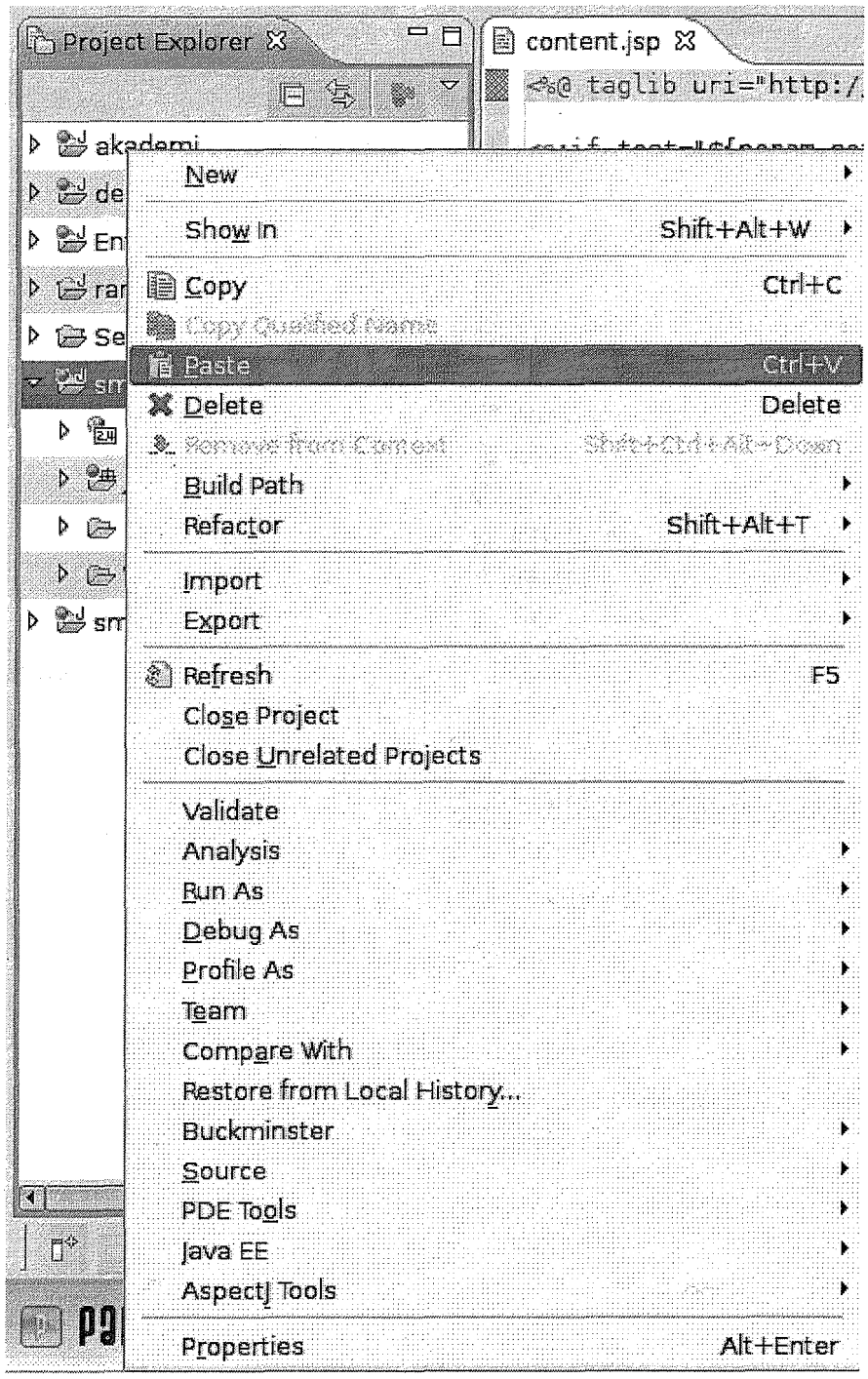

Figure 2.32: Create New JSP page

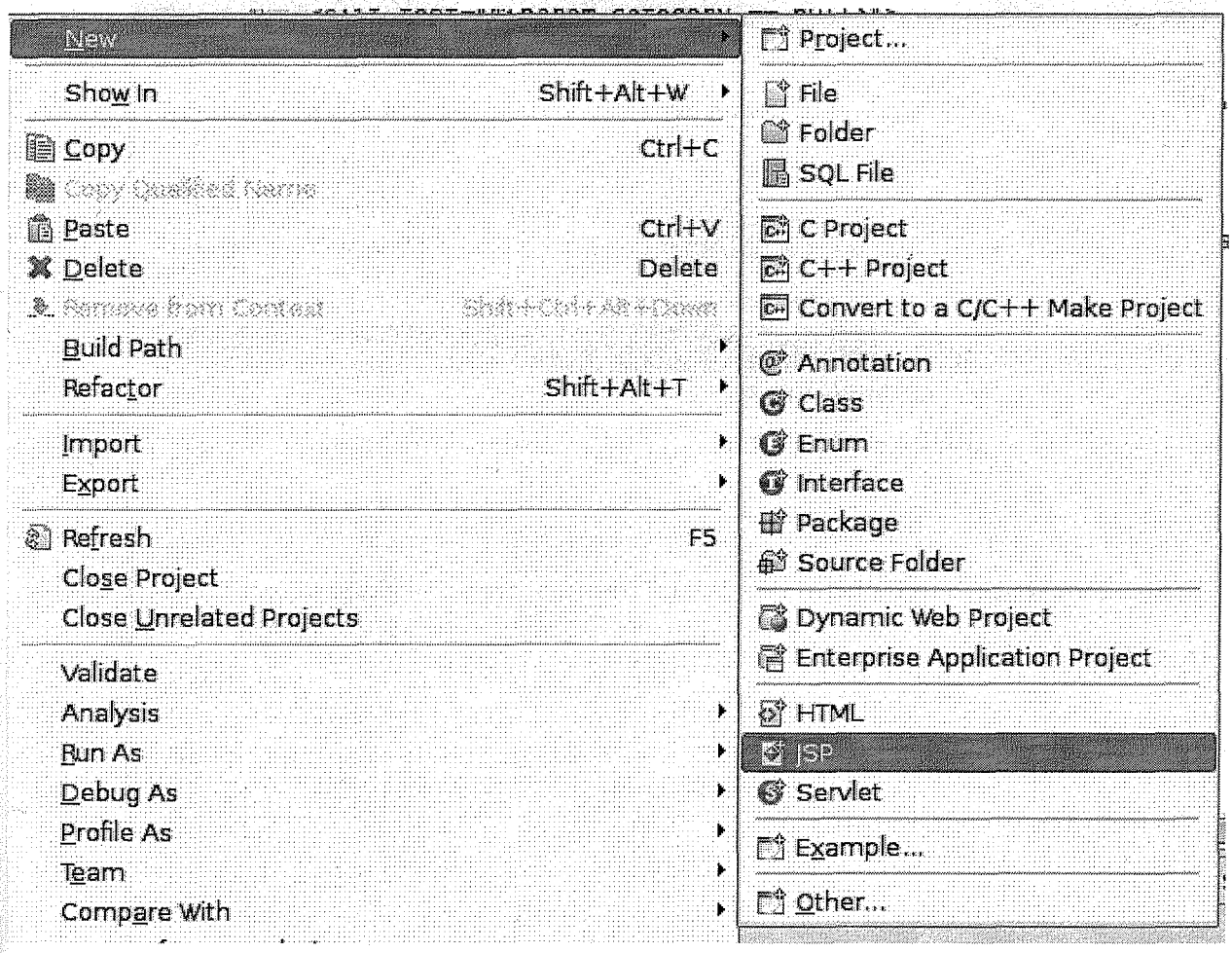

Figure 2.33: Create JSP page

For creating JSP files, should be select New> JSP. (Shown in Figure 2.33)

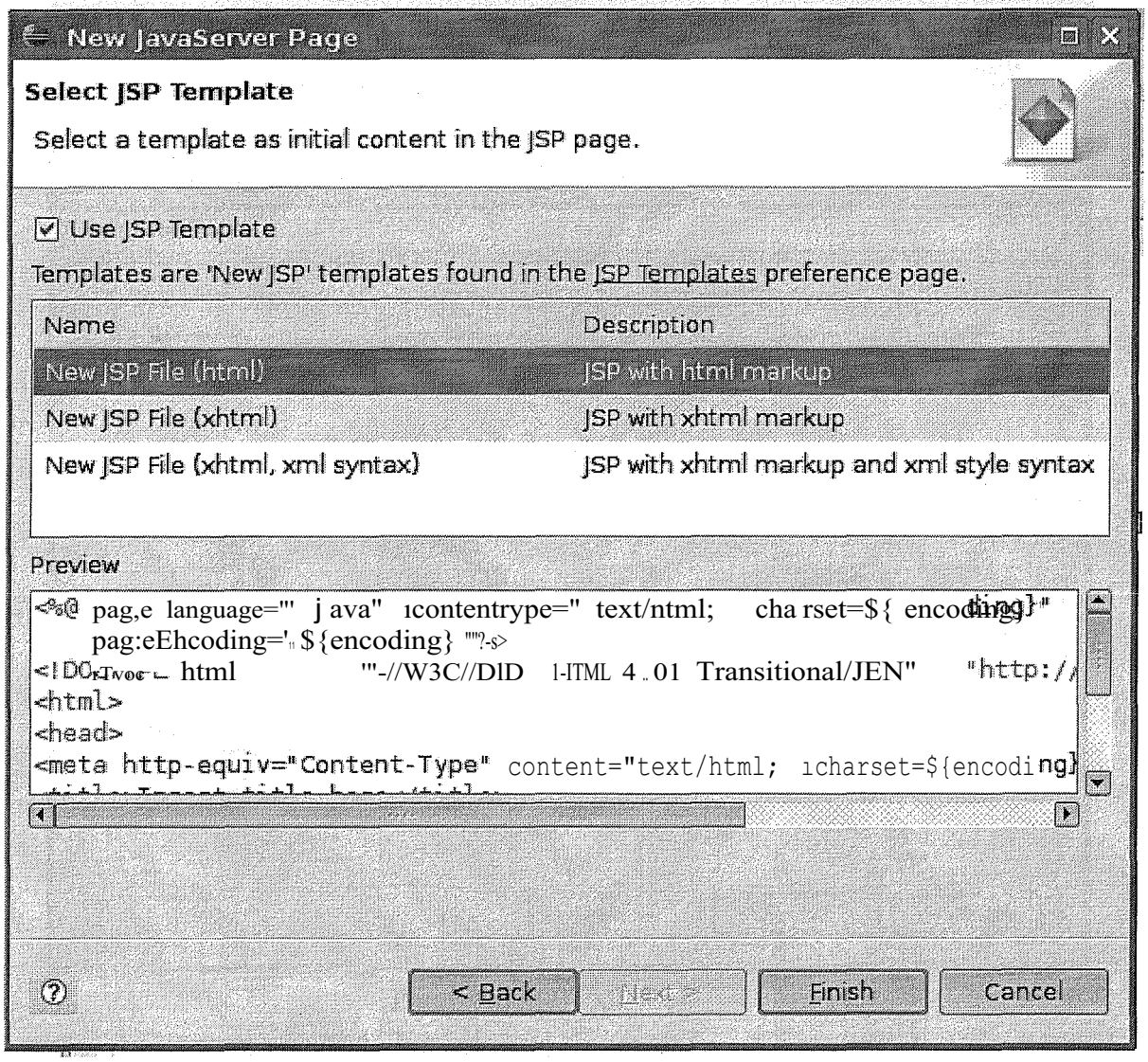

Figure 2.34: Give JSP page a Name

When *ISP* page is created, should be given a name. I have just a give "index" and it creates for me, index.jsp page.

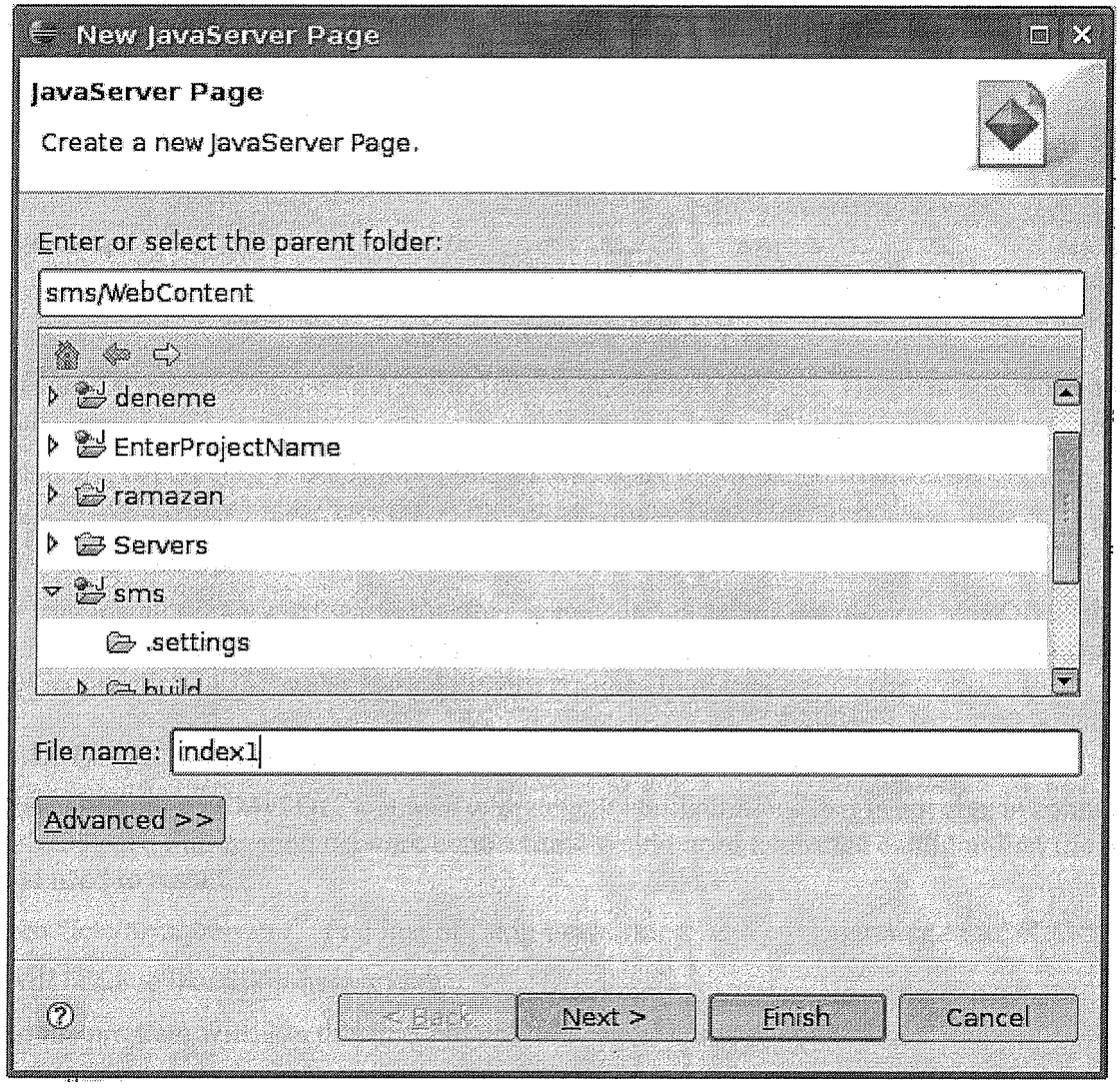

Figure 2.35: HTML Style

Select which HTML code your JSP file will be converted on the server when client wants to open.

2.1.5.2.6.Run with Tomcat

We still create a Dynamic Web Project and index.jsp but needs to connect with Apache Tomcat. It connects automatically-just need to show path of the apache directory.

Running Web Projects in this order.

Select the project from Project Manager that wanted to run.

Run > Run as > Run on Server.

2.2.Download necessary libraries

The project needs to run some drivers for run. MySQL connector for connecting to MySQL database in Java. Upload libs for documents uploading. JSTL and Standard for tag libraries.

MySQL connector can be downloaded by MySQL's web site that is www.mysql.com >

developers zone > Downloads > Connectors > Connector/I > Pick a mirror and download it (Shown in Figure 2.36).

Downfoads

Source and Binaries (tar.gz)	5.1.6	8.2M	Pick a mirror
		MD5: 84641aa4ddc138fc400366021f05cec5	Signature
Source and Binaries (zip)	5.1.6	8.4M	Pick a mirror
		MD5: 831f0e4.54b8alc399ca37f0668dfü585	Signature

Figure 2.36: MySQL Download Page

JSTL and standard comes with Apache Tomcat in example projects. They can download able in Apache web site.

Uploading documents, it needs four libraries which are downloaded from Javazoom.com with this link http://www.javazoom.net/jzservlets/uploadbean/uploadbean.html directly.

2.3.Configurations

For running all of these softwares together, some configurations are needed. MySQL is installed but it does not run on localhost. So it needs open to local. For using tag libraries, uploading documents and MySQL connection in JSP codes should the drivers need to placed.

2.3.1.MySQL Network Access

After installation of MySQL, it comes with default configurations that is not able to connect to it by using 3306 port. For opening network connections should need to change configuration file of MySQL and need to restart.

Open "/etc/mysql/my.cnf', find line of"skip-networking" and put beginning of it"#".

It will become like #skip-networking

And the program will skip this line and will not cancel network connections.

For restarting MySQL, write this to console.

"mysql restart".

2.3.2.MySQL Connection Driver

It should be placed two different way. First one is to place under Apache Tomcat server, the directory lib.

Second way is using Eclipse is to place under project > Web Content> Web-Inf> Lib

2.3.3. Tag and Bean Libraries

For adding these libraries, should be used Eclipse Project manager.

Adding these files to Under Project> Web Content> Web-Inf> Lib.

2.4.How does the system work?

In the project, Eclipse shows the the hierarchy of the project that can be controlled by Project Manager. The project designed by groups of student, instructor, staff and administrator. The project has directories for all different groups user.

The codes is written to Eclipse and databases created by MySQL. There is DB_Student.sql file includes all create tables.

CHAPTER THREE

STUDENT MANf'\GEMENT SYSTEM APPLICATION

Backside of the system, Tomcat and MySQL will run. So, database should be well-designed. Front side, there should be internet explorer program for remote user to access the system.

3.1.Database design

SMS database consist of sixteen tables. Those are ANNOUNCES, BUILDING, COURSES, CRSSECTION, DEPARTMENT, DOCUMENTS, INSTRUCTOR, LOCATION, MAJOR, REGISTRATION, ROOM, SCHEDULE, SECURITY, STAFF, STUDENT and TERM tables.

ANNOUNCES table contains seven fields:

This table for sending announces to students, staffs, instructors or a faculty members. Primary key is AnnouncesID and fields are:

- AnnouncesID
- status
- sendfrom
- senderStatus
- sendto
- theDate
- content

BUILDING table contains two fields:

Building table is about building in the university that are avaible to use. Primary key is BuildingID.

- BuildingID
- BuildingName

COURSES table contains four fields:

This table is the list of all cources in the university. Primary key of it is CourseID.

- CourseID
- Title
- Credits
- PreReq

CRSSECTION table contains six fields:

Crssection table is the list of courses that open in a term with group. Primary key is CsID.

- CsID
- CourseID

- Section
- TeimID
- InstructorID
- MaxST

DEPARTMENT table contains three fields:

This table collects information of the department. Primary key is DeptID.

- DeptID
- DeptDesc
- Dean

DOCUMENTS table contains eleven fields:

This table stores the documents by a student name and file with the access permission.

- DocID
- uploader
- toWho
- fileName
- fileSize
- doc
- theDate
- contentType
- st
- ins
- stf

INSTRUCTOR table contains ten fields:

It saves the informations of instructors. Primary key is InstructorID.

- InstructorID
- Name
- Surname
- Birthdate
- DeptID
- RoomID
- Rphone
- Adress
- Phone
- EMail

LOCATION table contains five fields:

Location table collects the information of rooms. Primary key is RoomID.

- RoomID
- BuildingID
- RoomNo
- Capacity
- RoomType

MAJOR table contains two fields:

This table stores major of department information. Primary key is MajorID.

- MajorID
- MajorDesc

REGISTRATION table contains four fields:

Registration table stores the grades of a student. Primary keys are StudentID and CsID.

- StudentID
- CsID
- Result

ROOM table contains two fields:

Room table saves the information of room type, classroom or laboratory.

- RoomType
- RoomDesc

SCHEDULE table contains four fields:

This table stores schedule information of date of a lecture.

- CsID
- theDate
- RoomID
- ScDesc

SECURITY table contains three fields:

Security table saves information of user name and its password. Passwords are 32 bits encription by md5 function. Primary key is userID.

- UserID
- pswrd
- status

STAFF table contains nine fields:

This table saves the informations of staffes.Primary key 'is StafflD.

- StaftID
- Name
- Surname
- Birthdate
- RoomID
- RPhone
- Adress
- Phone
- EMail

STUDENT table contains ten fields:

Student table collects the informations of students. Primary key is StudentID which is equal to Studen Number.

- StudentID
- Name
- Surname
- Birthdate
- Adress
- Phone
- Email
- StartTerm
- AdvisorID
- MajorID

TERM table contains four fields:

This table collects the information of terms. Primary key is TermID.

- TermID
- TermDesc
- StartDate
- EndDate

3.2.Features of Users

Every user has to permitted because of security and not all user has the features, so it should all different user panel.

3.2.1.Student

- See his/her information
- Update his/her adress and telephone informations

- See grades
- See documents if permitted
- See announces
- See exam schedule
- Send a message to instructor or staff

3.2.2.Instructor

- See his/her information
- Update his/her adress and telephone informations
- Enter grades
- See documents if permitted
- Upload documents
- Manage student information
- See announces
- Send announces to a student
- Send announces to students who takes Same cources.

3.2.3.Staff

- See his/her information
- Update his/her adress and telephone informations
- See documents if permitted
- Manage student information
- Enter exam dates
- See announces
- Send announces to a student
- Send announces to student who started in the same term

3.2.4.Administrator/ Admin

- Control everthing

3.3.Student

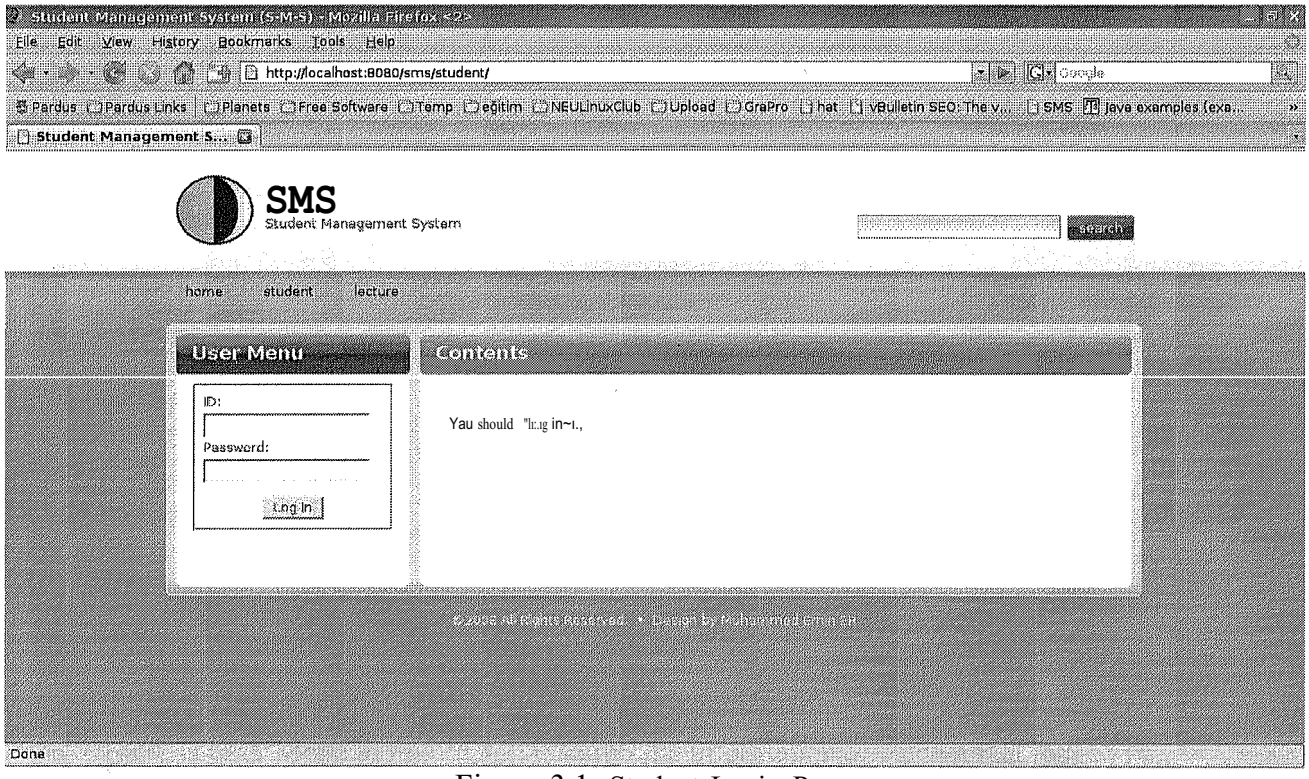

Figure 3.1: Student Login Page

Entering the system, student should enter his/her Student Number (StudentID) and his/her password. ID and password will be checked in the security table and checked if it is true and status is one(one means user should be a student). (Shown in Figure 3.1)

StudentID is 2003151 7 and password 2003151 7 was entered to the system and accepted by the system then go to main menu(Shown in Figure 3.2).

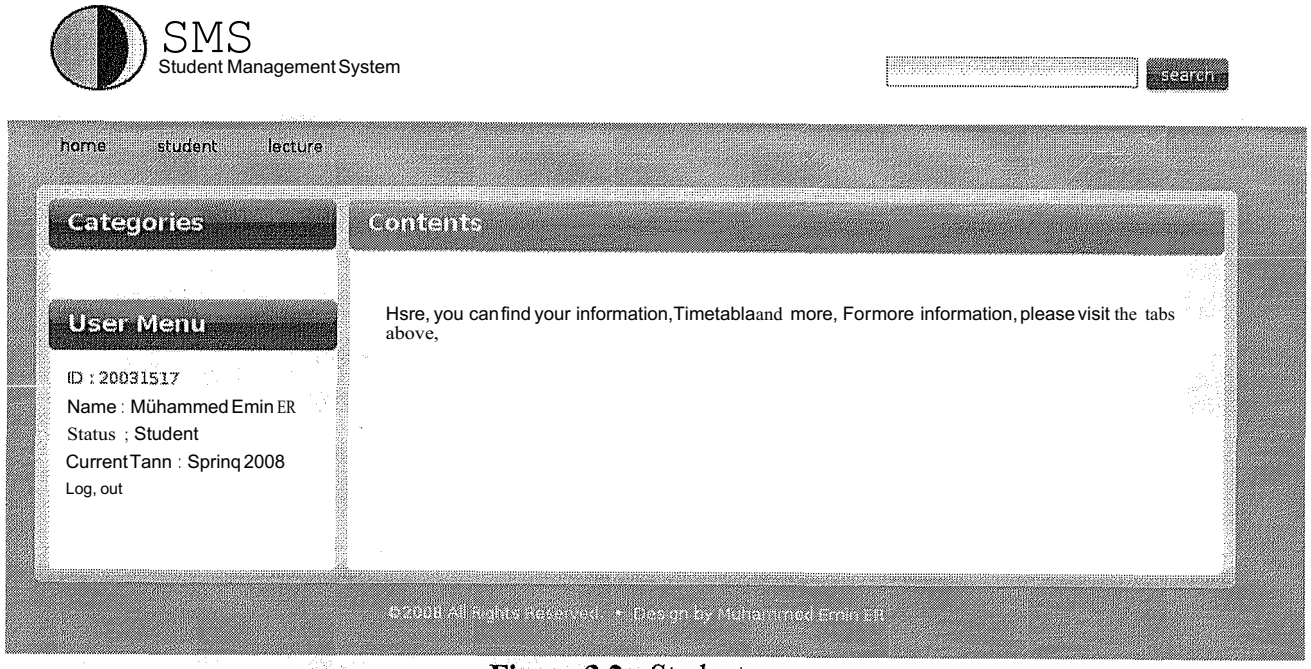

Figure 3.2: Student menu

There is User Menu in Figure 3.2, it shows the information of student that logged in.

ID: 20031517 is the ID of user.

Name: Muhammed Emin ER is the name of user.

Current Term: Spring 2008 is the current term of that year.

Log out is for logging out from system. For security, user should log out with clicking this button.

Figure 3.3: Student Tab

Under student tab, there are information, announces and my documents buttons in the categories (Shown 3.3).

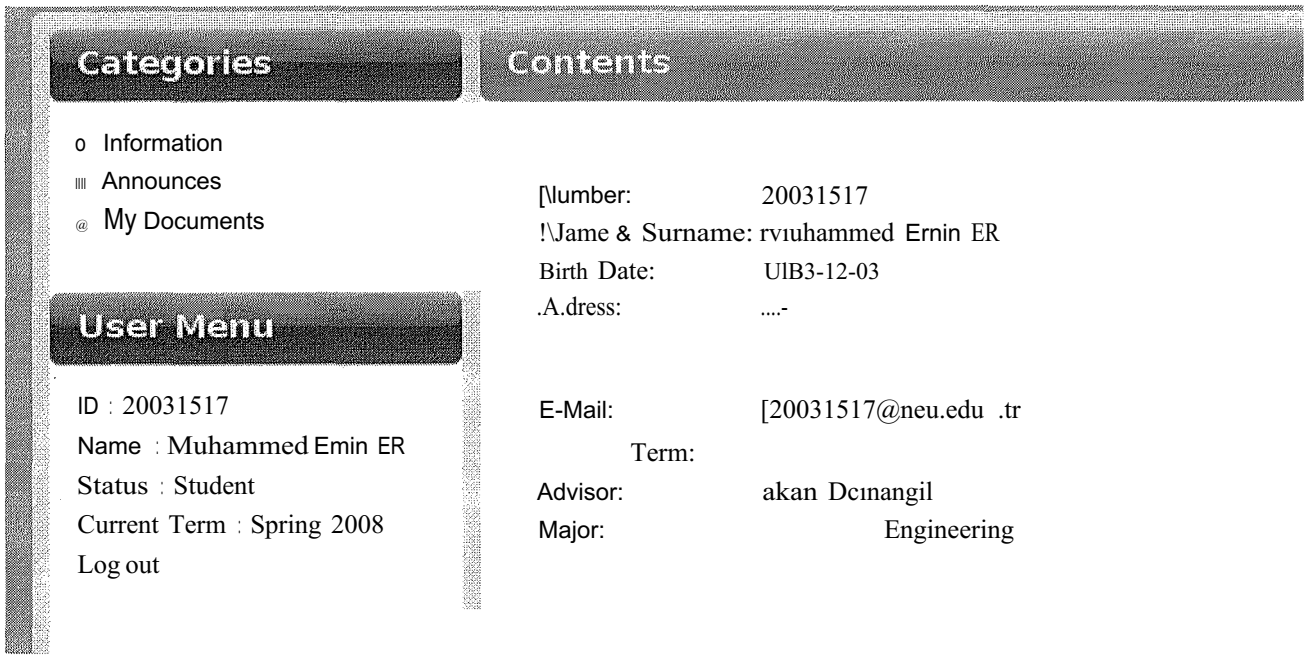

Figure 3.4: Student Information

Student is able to change his/her information and see it. It is just permitted to change address, phone and e-mail informations. In Figure 3.4 is shown the information of Muhammed Emin ER.

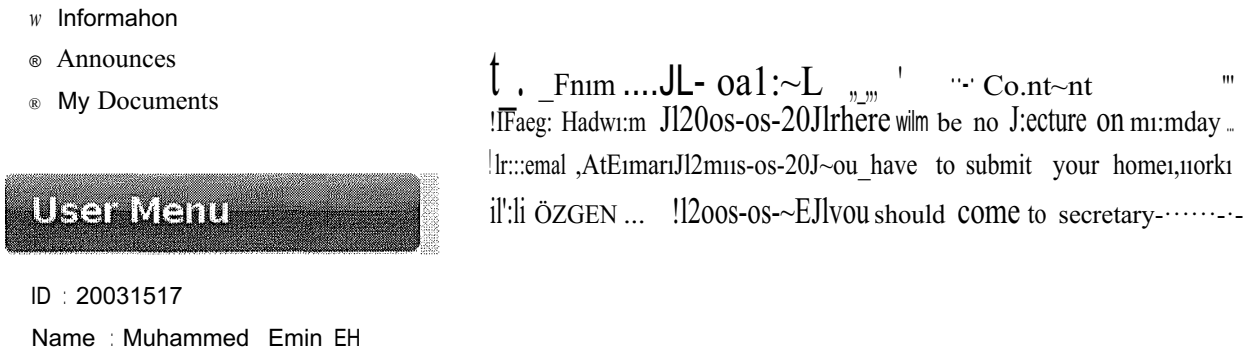

Figure 3.5: Student Announces

Student can see announces sent to him/her. In the Figure 3.5, there are three announces to send him.

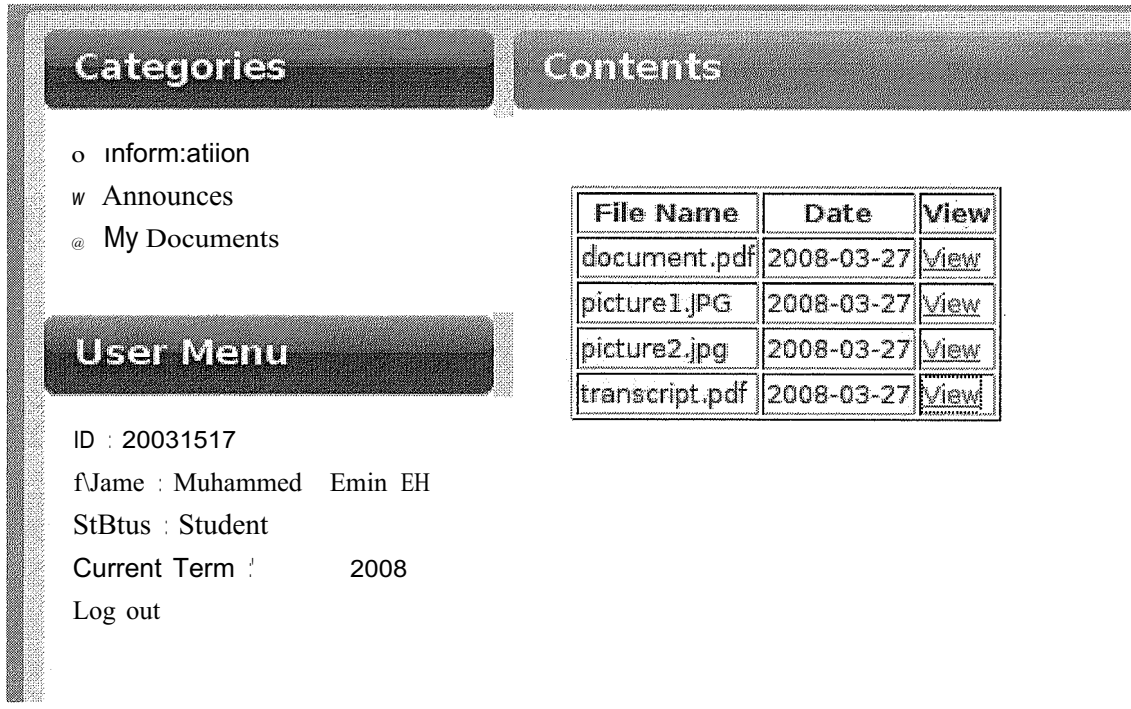

Figure 3.5: My Documents

Student can see his/her documents if permitted to see by uploader. In Figure 3.5 it shows the documents of Muhammed Emin ER. Student can see pdf, jpg ,txt and more formats in there computer with downloading it. Click view link for seeing the document.

In the Figure 3.5 there four documents, for showing transript.pdf click view of it. Then computer will download it and start it with the operating system's program(Shownin Figure 3.6).

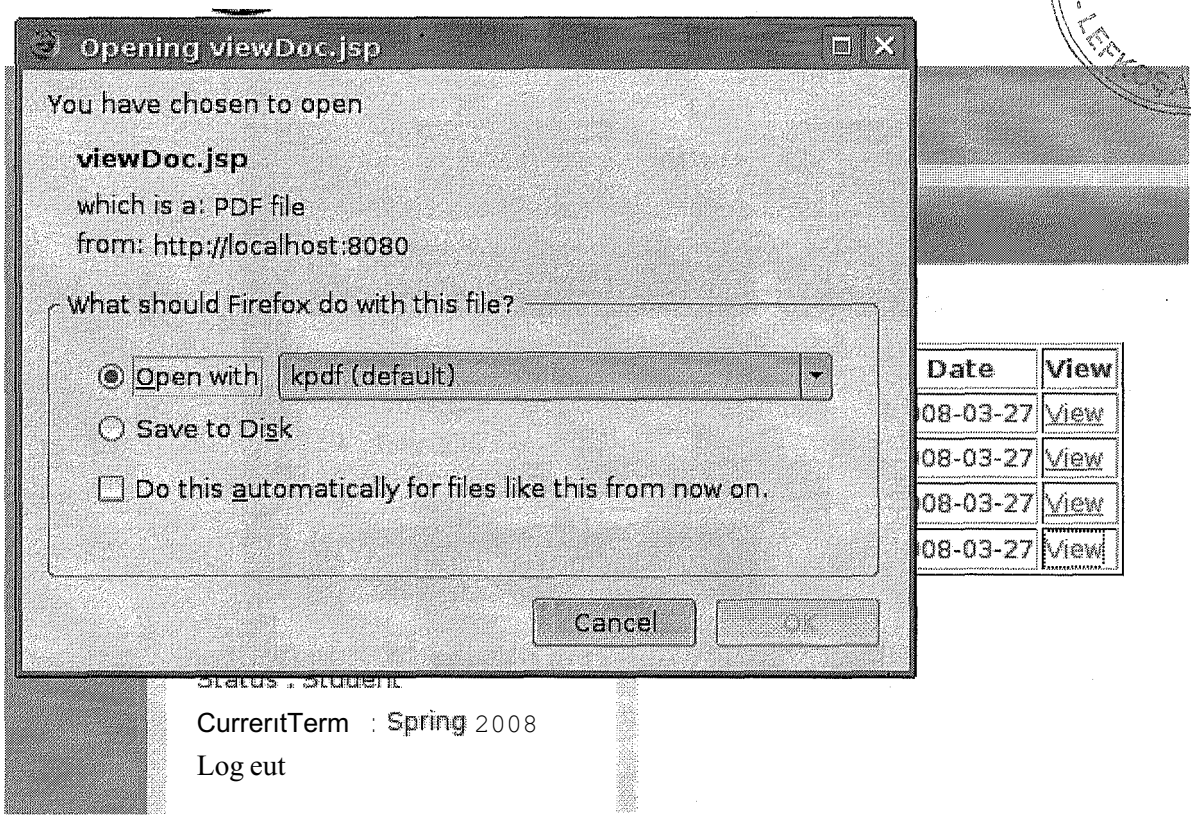

Figure 3.6: Download Document

Then operating system will open **it** with the suitable program (Show in Figure 3.7).

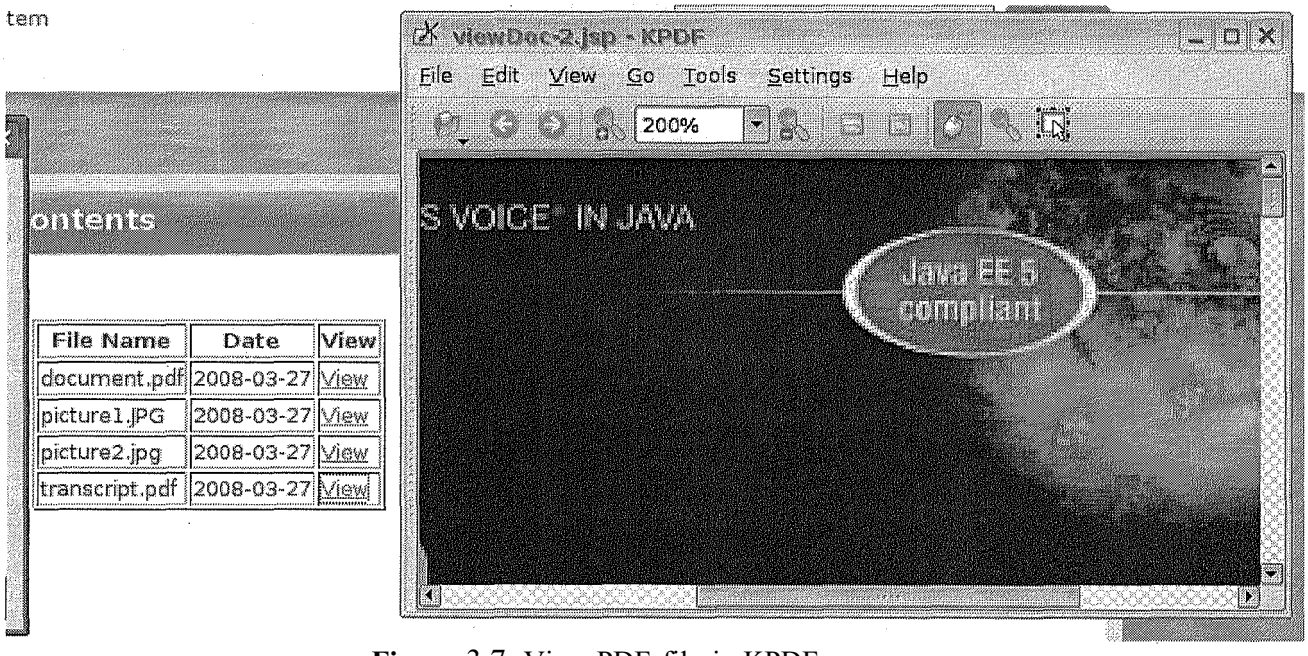

Figure 3.7: View PDF file in KPDF program

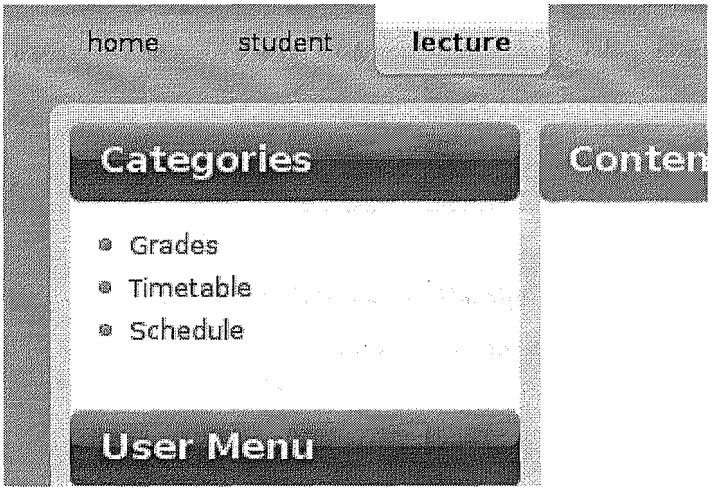

Figure 3.8: Lecture Tab

In lecture section, student can see grades and schedule(Shown in Figure 3.8).

Fall 2004	COM111	BA
Fall 2004	MAT101	FD
Fall 2004	PHY101	AA
Fall 2004	COM141	AA
Fall 2004	ENG101	CC
Fall 2004	COM142	AA
Spring 2005	MAT101	DD
Spring 2005	COM121	CB
Spring 2005	ENG102	CC
Spring 2005	PHY102	DD
Spring 2005	COM210	DD
Fall 2005	MAT102	BB
Fall 2005	MAT112	AA
Fall 2005	COM211	CB
Fall 2005	COM241	DD
Fall 2005	EE207	CC
Spring 2006	MAT201	AA

Figure 3.9: Grades of the Student

Student is able to see his/her grades all terms (Shown in Figure 3.9). First column is Term information, second is the name of course and last is the grades of that cource.

CourseID	Date	Building	Room	Description
COM344-G1	2008-06-09	Architecture Building	17	final exam
MAN402-G1	2008-06-10	Architecture Building	18	2nd midterm
COM420-G1	2008-06-11	Faculty of Pharmacology	13	2nd midterm
COM432-G1	2008-06-12	Ataturk Culture & Congress Centre	2	final exam
COM442-G1	2008-06-13	Ataturk Culture & Congress Centre	3	2nd midterm
COM450-G1	2008-06-14	Ataturk Culture & Congress Centre	4	quiz

Figure 3.10: Schedule.Of the student

It is very important a student can see when there is a exam and exam schedule with its places(Shown Muhammed's schedule in Figure 3. 10).

3.4.Instructor

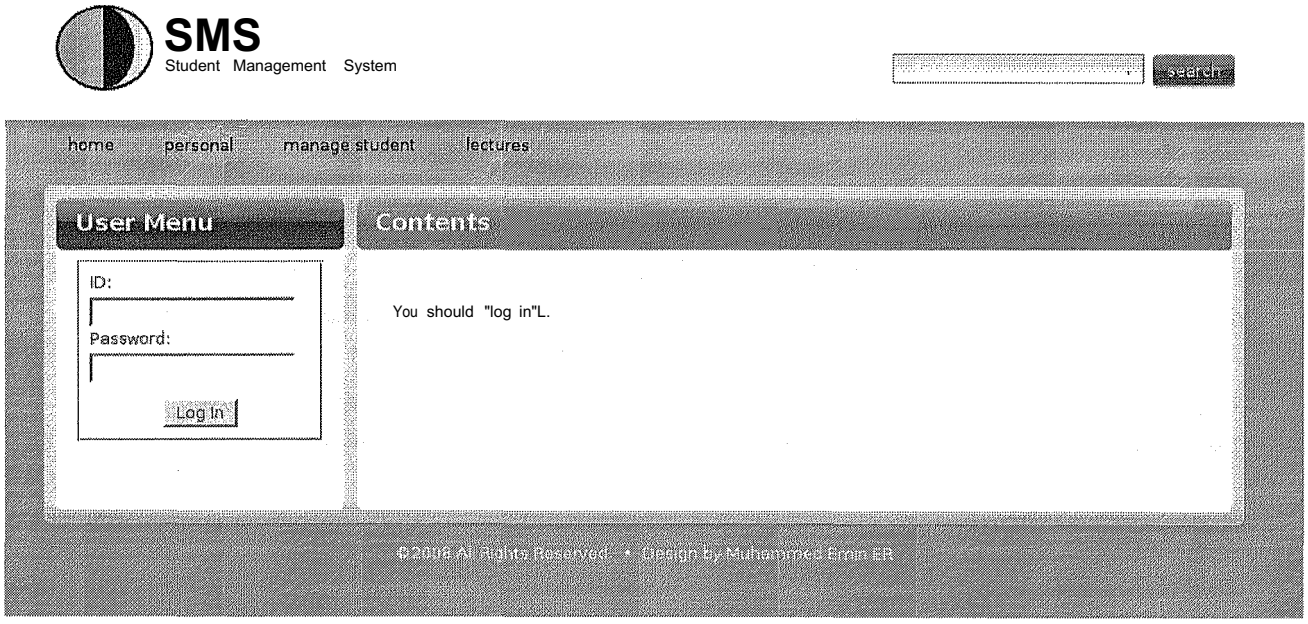

Figure 3.11: Instructor Log in

Entering the system, instructor should enter his/her instructor id (InstructorID) and his/her password. ID and password will be checked in the security table and checked if it is true and status is two(two means user should be instructor)(Shown in Figure 3.11).

Figure 3.12: Instructor logged in.

InstructorID is odanangil and password odanangil was entered to the system and accepted by the system then go to main menu(Shown in Figure 3. 12).

In the Figure 3.12 shows information of the user in the User Menu. ID is odonangil which is user name, Name is the name of user, status for the user status(student, instructor or staff) and Current Term is for the term in that time.

Log out is for logging out, it should be necessary for going out from system because of security.

There are four tabs, home, personal, manage student and lectures. Home is at the beginning the page after log in. Personal is the information table of the instructor. Manage Student is for managing student informations. And lectures for the information about courses which the instructor give.

Personal tab has a two page information and timetable(Shown in Figure 3.13).

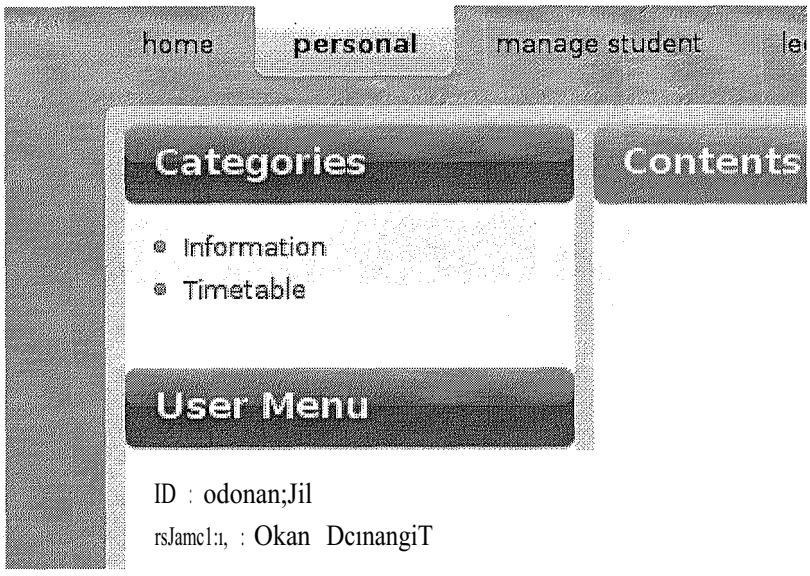

Figure 3.13: personal

Information page.shöws the information of the user(Shown in Figure 3.14).

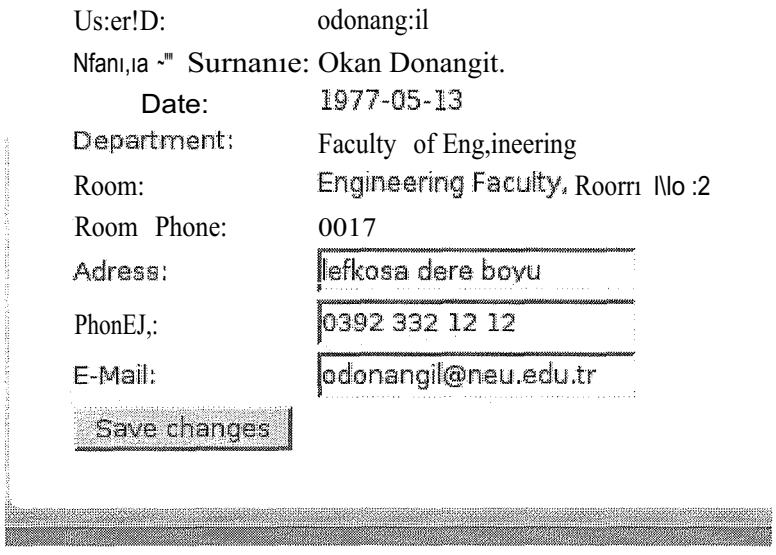

Figure 3.14: Instructor Information

Manage Student is a work for instructor who is the advisor of that student. In this work, there are six functions, Select Student is for selecting student, Information is for seeing the information of selected student, Grades is grades of selected student, Timetable is the timetable of selected student, Document is documents of the student that given permission of the instructor, Upload Documents is for uploading documents(Shown in Figure 3.15).

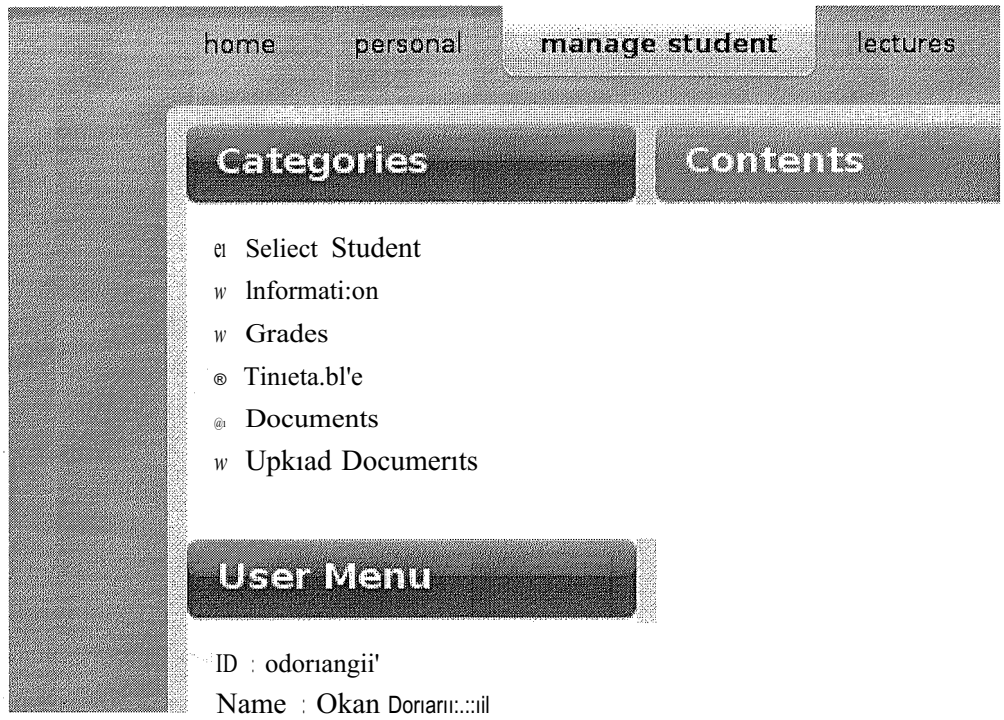

Figure 3.15: Manage Student

For managing studeritiiiformatiön, firstly student should be selected (Shown in figure 3.16)

Figure 3.16: Select Student

Write to StudentID to the empty box and search for him/her if she/he is there(Shown Figure 3.17).

Figure 3.17: Search Student

After searching student with 20031517 is found then shows his information (Shown in Figure 3.18)

Select Student:

Search

Selected Student is:

StudentID:	20031517
Name & Surname:	Muhammed Emin ER
Birthdate:	1983-12-03
Start Term:	Fall 2003
Advisor:	Okan Donangil

Figure3.18: Student is Found

If student is found, the other pages can be accessed. In information page, the instructor is able to change the information of the selected student (Shown in Figure 3.19).

- Select Student
- ınformıatiOn
- Grades
- Timetable
- Documents
- Upl:oad Doctımerıts

User Menu

ID : odonaıngili

Name : Okan Dorıangil

Status : Instructor

Number:	2003.1517
fNl'arne & Surname:	Mühammed Emin ER
Birth Date:	1983-12.-mı
Adress:	konya
Phone:	0533 865 3914
E-rvıaif:,	20031517@neu.edu.tr
Started Term:	F:al'l: 2003
Advisor:	DonangH
Major.:	Computer Engüneering

Save changes

Figure 3.19: Selected Student Information

Grades is for selected student grades(Shown in Figure 3.20).

Selected Student is:

StudentID:	20031517
Name & Surname:	Muhammed Emin ER
Birthdate:	1983-12-03
Start Term:	Fall 2003
Advisor:	Okan Donangil

Grades:

Figure 3.20: Selected StuderıtGrades

Instructor is able to see the selected student'sdocuıiıents(Showhin Figu.re3.21)

StudentID:	20031517
Name & Surname:	Muhammed Emin ER
Birthdate:	1983-12-03
Start Term:	Fall 2003
Advisor:	Okan Donangil

Okan Donangil	picture2.jpg	92403	2008-03-27	
Okan Donangil	transcript.pdf	6893774	2008-03-27	View
				Delete Selected

Figure 3.21: Selected Student Documents

Instructor can show(Shown in Figure 3.22) or deleted (Shown in Figure 3.23) the documents

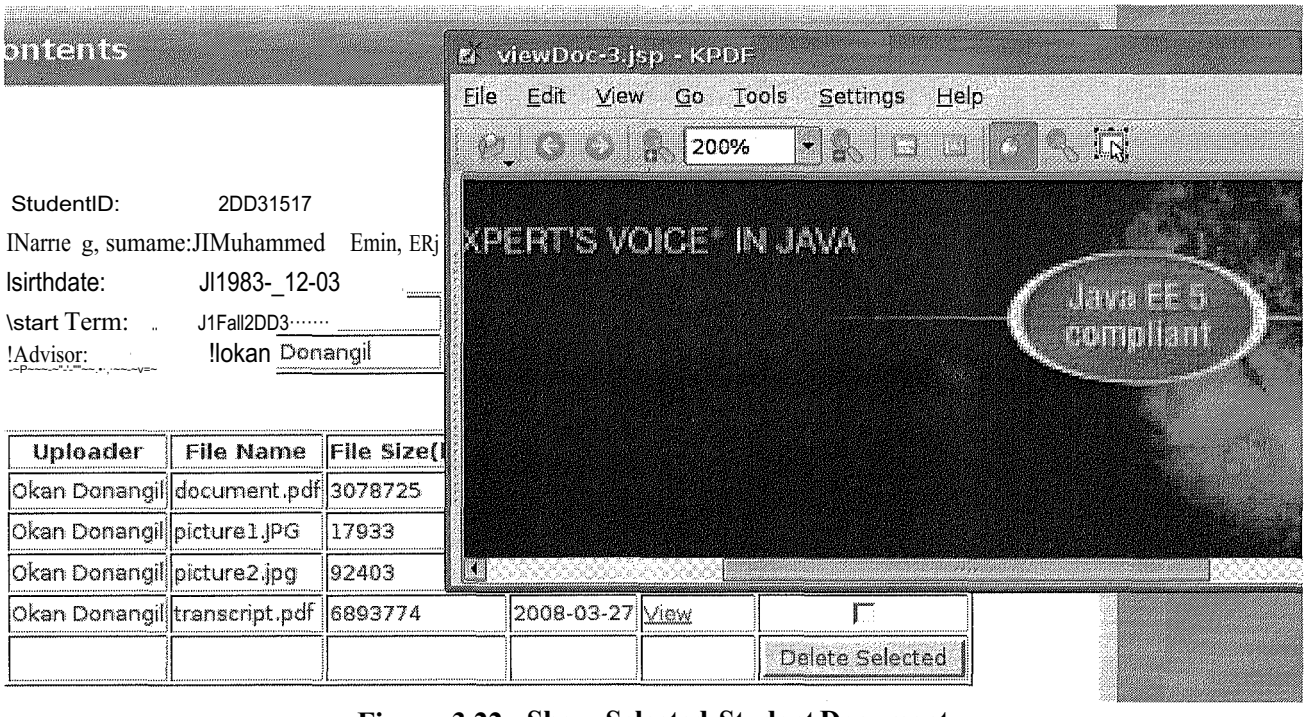

Figure 3.22: Show Selected Student Document

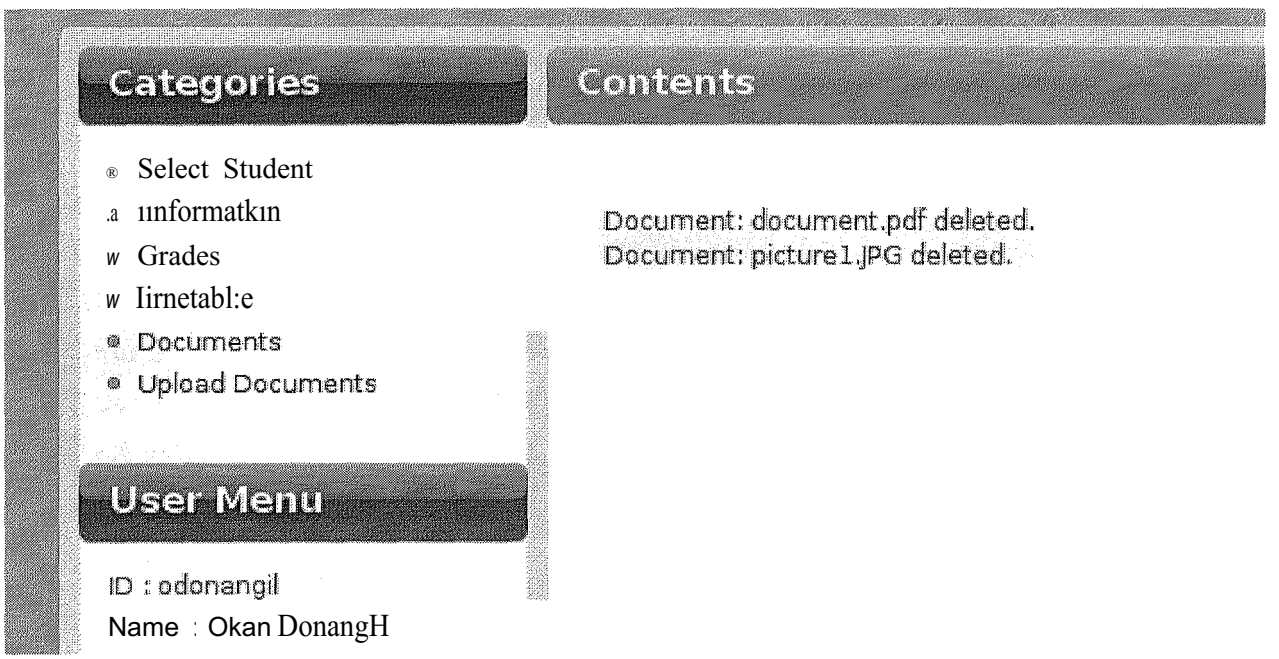

Figure 3.23: Delete Selected Student Document

In.structor can control uploading documents to students, in the Figure 3.24 it shows instructor upload page, in the FigureJ.25 instructor select a document from her/his computer, in the Figure 3.26 instructor upload the fıleaııd there is file information in Figure 3.27.

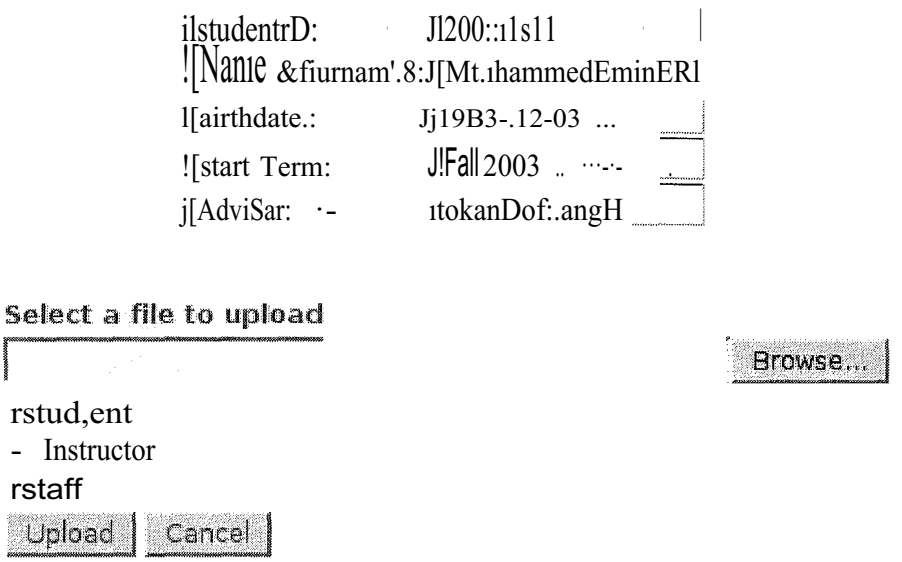

Figure 3.24: Instructor Upload Page

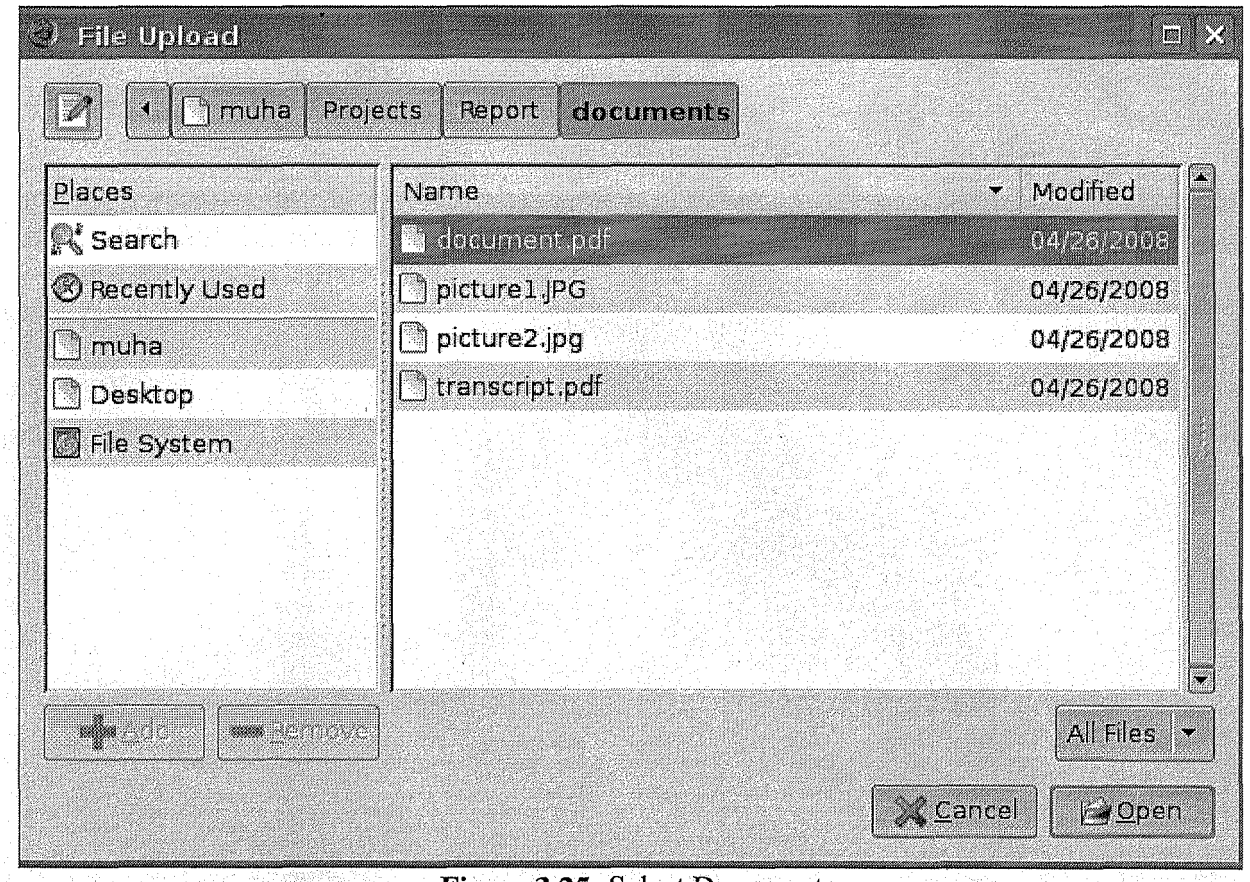

Figure 3.25: Select Document

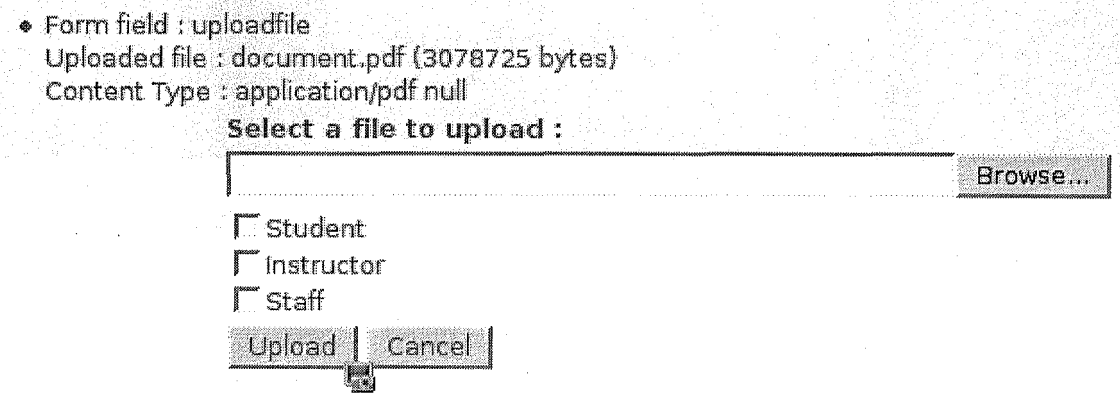

Figure 3.26: Document Uploaded

In Figure 3.26 document.pdfwere upload, it is 3078725 bytes and it's content type is pdf.

3.5.Database Tables

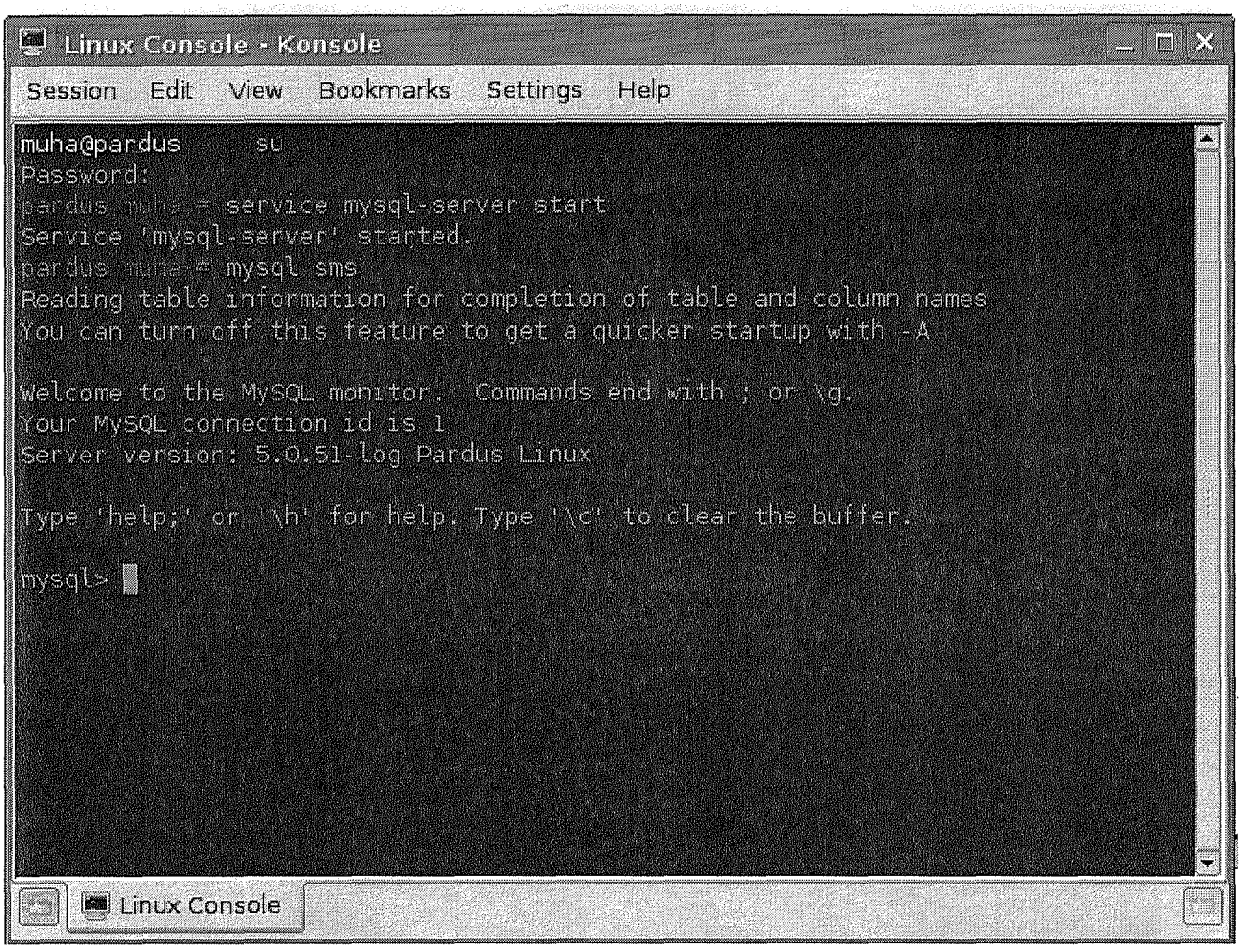

Figure 3.27: Open MySQL

For opening My~QL, firstly, should become super user with typing "su".then the password of root. AfterthatstarttheBervice of mysql-server. Then enter the mysql with using sms database. It automatically enter the database SMS (Shown in Figure 3.27).

```
mysql> show tables;
+---------------+
| Tables_in_sms |
+---------------+
| ANNOUNCES     |
| BUILDING      |
| COURSES       |
| CRSSECTION    |
| DEPARTMENT    |
| DOCUMENTS     |
| INSTRUCTOR    |
| LOCATION      |
| MAJOR         |
| REGISTRATION  |
| ROOM          |
| SCHEDULE      |
| SECURITY      |
| STAFF         |
| STUDENT       |
| TERM          |
+---------------+
16 rows in set (0.00 sec)

mysql>
```

Figure 3.28: List of Tables in SMS database

In MySQL, typing show tables means show all tables which is dependent to SMS database (Shown in Figure 3.28).

After all Figures show us the description of all tables (Figure 3.29 - 3.44)

```
mysql> desc ANNOUNCES;
+--------------+-------------+------+-----+---------+----------------+
| Field        | Type        | Null | Key | Default | Extra          |
+--------------+-------------+------+-----+---------+----------------+
| announcesID  | int(9)      | NO   | PRI | NULL    | auto_increment |
| status       | tinyint(1)  | YES  |     | 1       |                |
| sendfrom     | varchar(12) | YES  |     | NULL    |                |
| senderStatus | int(1)      | YES  |     | NULL    |                |
| sendto       | varchar(12) | YES  |     | NULL    |                |
| theDate      | date        | YES  |     | NULL    |                |
| content      | blob        | YES  |     | NULL    |                |
+--------------+-------------+------+-----+---------+----------------+
7 rows in set (0.00 sec)
```

Figure 3.29: Announces Table

```
mysql> desc BUILDING;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| BuildingID   | varchar(4)  | NO   | PRI | NULL    |       |
| BuildingName | varchar(35) | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Figure 3.30: Building Table

```
mysql> desc COURSES;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| CourseID | varchar(8)  | NO   | PRI | NULL    |       |
| Title    | varchar(30) | YES  |     | NULL    |       |
| Credits  | int(1)      | YES  |     | NULL    |       |
| PreReq   | varchar(8)  | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Figure 3.31: Courses Table

```
mysql> desc CRSSECTION;
+--------------+-----------------+------+-----+---------+----------------+
| Field        | Type            | Null | Key | Default | Extra          |
+--------------+-----------------+------+-----+---------+----------------+
| CsID         | int(6) unsigned | NO   | PRI | NULL    | auto_increment |
| CourseID     | varchar(8)      | YES  |     | NULL    |                |
| Section      | char(2)         | YES  |     | G1      |                |
| TermID       | char(4)         | YES  |     | NULL    |                |
| InstructorID | varchar(12)     | YES  |     | NULL    |                |
| MaxST        | int(3)          | YES  |     | NULL    |                |
+--------------+-----------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

Figure 3.32: Crssection Table

```
mysql> desc DEPARTMENT;
+----------+----------------+------+-----+---------+----------------+
| Field    | Type           | Null | Key | Default | Extra          |
+----------+----------------+------+-----+---------+----------------+
| DeptID   | int(2) unsigned | NO   | PRI | NULL    | auto_increment |
| DeptDesc | varchar(40)    | YES  |     | NULL    |                |
| Dean     | int(5)         | YES  |     | NULL    |                |
+----------+----------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

Figure 3.33: Department Table

```
mysql> desc DOCUMENTS;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| DocID       | int(10)     | NO   | PRI | NULL    | auto_increment |
| uploader    | varchar(12) | NO   |     | NULL    |                |
| towho       | varchar(12) | NO   |     | NULL    |                |
| fileName    | varchar(65) | YES  |     | NULL    |                |
| fileSize    | int(15)     | YES  |     | NULL    |                |
| doc         | mediumblob  | YES  |     | NULL    |                |
| theDate     | date        | YES  |     | NULL    |                |
| contentType | varchar(15) | YES  |     | NULL    |                |
| st          | tinyint(1)  | YES  |     | NULL    |                |
| ins         | tinyint(1)  | YES  |     | NULL    |                |
| stf         | tinyint(1)  | YES  |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
11 rows in set (0.00 sec)
```

Figure 3.34: Document Table

```
mysql> desc INSTRUCTOR;
+--------------+----------------+------+-----+---------+-------+
| Field        | Type           | Null | Key | Default | Extra |
+--------------+----------------+------+-----+---------+-------+
| InstructorID | varchar(12)    | NO   | PRI | NULL    |       |
| Name         | varchar(25)    | NO   |     | NULL    |       |
| Surname      | varchar(20)    | NO   |     | NULL    |       |
| Birthdate    | date           | NO   |     | NULL    |       |
| DeptID       | int(2) unsigned | NO   |     | NULL    |       |
| RoomID       | int(5) unsigned | YES  |     | NULL    |       |
| RPhone       | varchar(4)     | YES  |     | NULL    |       |
| Adress       | varchar(30)    | YES  |     | NULL    |       |
| Phone        | varchar(15)    | YES  |     | NULL    |       |
| EMail        | varchar(25)    | YES  |     | NULL    |       |
+--------------+----------------+------+-----+---------+-------+
10 rows in set (0.00 sec)
```

Figure **3.35:** Instructor Table

```
mysql> desc LOCATION;
+-----------+-----------------+------+-----+---------+----------------+
| Field     | Type            | Null | Key | Default | Extra          |
+-----------+-----------------+------+-----+---------+----------------+
| RoomID    | int(5) unsigned | NO   | PRI | NULL    | auto_increment |
| BuildingID | varchar(4)      | NO   |     | NULL    |                |
| RoomNo    | varchar(3)      | NO   |     | NULL    |                |
| Capacity  | int(4)          | YES  |     | NULL    |                |
| RoomType  | char(1)         | YES  |     | NULL    |                |
+-----------+-----------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

Figure 3.36: Location Table

```
mysql> desc MAJOR;
+-----------+-----------------+------+-----+---------+----------------+
| Field     | Type            | Null | Key | Default | Extra          |
+-----------+-----------------+------+-----+---------+----------------+
| MajorID   | int(3) unsigned | NO   | PRI | NULL    | auto_increment |
| MajorDesc | varchar(40)     | YES  |     | NULL    |                |
+-----------+-----------------+------+-----+---------+----------------+
2 rows in set (0.00 sec)
```

Figure 3.37:Major Table

```
mysql> desc REGISTRATION;
+-----------+------------+------+-----+---------+-------+
| Field     | Type       | Null | Key | Default | Extra |
+-----------+------------+------+-----+---------+-------+
| StudentID | varchar(8) | NO   | PRI |         |       |
| CsID      | int(6)     | NO   | PRI | 0       |       |
| Result    | varchar(2) | YES  |     | NULL    |       |
+-----------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

Figure 3.38: Registratiorr'Iable

```
mysql> desc ROOM;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| RoomType | char(1)     | NO   | PRI | NULL    |       |
| RoomDesc | varchar(10) | NO   |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Figure 3.39: Room Table

```
mysql> desc SCHEDULE;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| CsID    | varchar(8)  | YES  |     | NULL    |       |
| theDate | date        | YES  |     | NULL    |       |
| RoomID  | int(5)      | YES  |     | NULL    |       |
| ScDesc  | varchar(20) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

Figure 3.40: Schedule Table

```
mysql> desc SECURITY;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| userID | varchar(12) | NO   | PRI | NULL    |       |
| pswrd  | char(32)    | YES  |     | NULL    |       |
| status | int(1)      | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

Figure 3.41: Security Table

```
mysql> desc STAFF;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| StaffID   | varchar(12) | NO   | PRI | NULL    |       |
| Name      | varchar(25) | NO   |     | NULL    |       |
| Surname   | varchar(20) | NO   |     | NULL    |       |
| Birthdate | date        | NO   |     | NULL    |       |
| RoomID    | varchar(2)  | YES  |     | NULL    |       |
| RPhone    | varchar(4)  | YES  |     | NULL    |       |
| Adress    | varchar(30) | YES  |     | NULL    |       |
| Phone     | varchar(15) | YES  |     | NULL    |       |
| EMail     | varchar(25) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
9 rows in set (0.00 sec)
```

Figure 3.42: Staff Table

```
mysql> desc TERM;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| TermID    | char(4)     | NO   | PRI | NULL    |       |
| TermDesc  | varchar(11) | YES  |     | NULL    |       |
| StartDate | date        | YES  |     | NULL    |       |
| EndDate   | date        | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Figure 3.43: Term Table

```
mysql> desc STUDENT;
+-----------+-----------------+------+-----+---------+-------+
| Field     | Type            | Null | Key | Default | Extra |
+-----------+-----------------+------+-----+---------+-------+
| StudentID | varchar(8)      | NO   | PRI | NULL    |       |
| Name      | varchar(25)     | NO   |     | NULL    |       |
| Surname   | varchar(20)     | NO   |     | NULL    |       |
| Birthdate | date            | NO   |     | NULL    |       |
| Adress    | varchar(30)     | YES  |     | NULL    |       |
| Phone     | varchar(15)     | YES  |     | NULL    |       |
| EMail     | varchar(25)     | YES  |     | NULL    |       |
| StartTerm | char(4)         | NO   |     | NULL    |       |
| AdvisorID | varchar(12)     | YES  |     | NULL    |       |
| MajorID   | int(3) unsigned | YES  |     | NULL    |       |
+-----------+-----------------+------+-----+---------+-------+
10 rows in set (0.00 sec)
```

Figure 3.44: Student Table

CONCLUSION

This project is designed and developed for NEU and for its educational system. It should be used for managing students information and data.

Whole system divided four section, student, instructor, staff and admin. In this case, all of different specification and features. But admin is the super user of the system. Also there can be more then one admin in the system. But at the end of this project, I know there should be a lot of features. So it can be larger and more develop able. It is also designed for develop able.

With this system, student can see their information, grades and timetable more easy and instructor can manage the students.

REFERENCES

[1] The Source for Java Developers from the World Wide Web http://java.sun.com/ and its documentation side.

[2] Java examples from the World Wide Web http://www.java2s.com/Code/Java/CatalogJava.htm

[3] MySQL from the World Wide Web

APPENDIX A

STUDENT

index.jsp

```
<%@ page contentType="text/html;charset=iso-8859-9"    pageEncoding="iso-8859-9"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd ">
<%@taglib uri="http://java.sun.com/jstl/core_rt"    prefix="c" %>
<html>
<head>
<meta http-equiv="Content-Type"  content="text/html;  charset=iso-8859-9">
<title>Student Management System (S-M-S)</title>
<meta name="keywords"  content="" />
<meta name="description"  content="" />
<link href="default.css"  rel="stylesheet"  type="text/css"  media="screen"  />
</head>

<body>
<!-- start header-->
<div id="header">
        <div id=ilogo">
                <hl>SMS</hl>
                <p>Student Management System</p>
        </div>
        <div id="search">
                <form method="get"  action="">
                        <fieldset>
                        <input id="s" type="text" name="s" value=""/>
                        <input id="x" type="submit" value="Search" />
                        </fieldset>
                </form>
        </div>
</div>
<!-- end header-->

<!-- start menu-->
<div id="menu">
        <ul>
```

```
                <c:import  url="./menu.jsp"  />
        </ul>
</div>
<!-- end menu-->
<!-- start page -->
<div id="page">
        <!-- start content=->
        <div id="content">
                <h1 class="pagetitle">Contents</h  1>
                <div class="post">
                        <c:import  url="./content.jsp"  />
                </div>
        </div>
        <!-- end content -->
        <!-- start sidebar -->
        <div id="sidebar">
                <ul>
                        <c:iftest="${sessionScope.status    == 1}">
                        <li>
                                <h2>Categories</h2>
                                <ul>
                                        <c:import  url="./categories.jsp"  />
                                </ul>
                        </li>
                        </c:if.>-
                        <li>
                                <h2>User  Menu</h2>
                                <ul>
                                        <c:import  url=" ./userMenu.jsp"  />
                                </ul>
                        </Ii>
                </ul>
        </div>
        <!-- end sidebar-->
        <div style="clear:  both;"> </div>
</div>
<!-- end page-->
<div id="footer">
        <p>&copy;2008  All Rights  Reserved.   &bull;   Design  by Muhammed  Emin  ER
</div>
</body>
</html>
```

Categories.jsp

```
st="${param.page  == 'student'}">
        <li><a  href="./?page=st<%@taglib    uri="http://java.sun.com/jstl/core_rt"    prefix="c"  %>
<c:if teudent&category=information   ">Information</ a></li>
```

```
        <li><a href=" ./?page=student&category=announces   ">Announces</a></li>
        <li><a href=" ./?page=student&category=myDoc   ">My Documents</a></li>
</c:if>
<c:iftest="${param.page   -   'lecture'}">
        <li><a href=" ./?page=lecture&category=grades   ">Grades</a></li>
        <li><a href=" ./?page=lecture&category=timeTable">   Timetable</a></li>
        <li><a href=" ./?page=lecture&category=schedule">Schedule</a></li>
</c:if>
```

Configure.jsp

```
<%@ taglib uri="http://java.sun.com/jstl/core   _rt" prefix="c"  %>

<c:set var="currentTerm"  value="SP08"  scope="session"/>
<c:set var="termDesc"  value="Spring  2008" scope="session"/>
```

content.jsp

```
<%@ taglib uri="http://java.sun.com/jstl/core   _rt" prefix="c"  %>

<c:iftest="${param.category    == null}">
        <c:import  url="./category/index.jsp"   />
</c:if>
<c:iftest="${param.category    != null and param.category!="}">
        <c:import  url="./category/${param.category}Jsp"    />
</c:if>
<c:iftest="${sessionScope.status    =O/l  sessionScope.status  =  null}">
        You should  "log in"!..
</c:if>
```

control.jsp

```
<%@ page contentType="text/html;charset=iso-8859-9"    pageEncoding="iso-8859-9"%>
<!DOCTYPE  html PUBLIC  "-//W3C//DTD  HTML 4.01  Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd   ">
<%@taglib   uri="http://java.sun.com/jstl/core_rt"    prefix="c"  %>
<%@page  import=" application Work.*"%>
<%@page  import="j ava. sql.ResultSet"%>
<html>
<head>
<meta http-equiv="Content-Type"   content="text/html;   charset=iso-8859-9">
<title>Controlling   Password</title>
</head>
<body>
<%
String userID = (String)request.getParameter("userID");
String pswrd = (String)request.getParameter("pswrd");
security  sc = new  security();
int status=  sc.getStatus(userID,pswrd);
```

```
if(status = 1){
        //if status = 1
        student st= new student();
        ResultSet rs= st.viewinformation(userID);
        try{
                while(rs.next()){
        %>
        <c:set var="userID" value="<%=rs.getString("ST.StudentID") %>" scope="session"/>
        <c:set var="userName" value="<%=rs.getString("ST.Name")+" "+rs.getString("Surname") %>"
scope="session" />
        <c:set var="status" value="1" scope="session" />
        <c:import url="./configure.jsp" />
        <% }//end while
        }//end try
        catch(Exception e){
        }
}else{
        //id status != 1
        %>
        <c:set var="userID" value="" scope="session"/>
        <c:set var="userName" value="" scope="session" />
        <c:set var="status" value="O" scope="session" />
        <c:set var="currentTerm" value="" scope="session" />
        <c:set var="termDesc" value="" scope="session"/>
        <c:set var="studentID" value="" scope="session"/>
        <%
}
%>
<c:redirect url="/student" />
</body>
</html>
```

default.css

```
body {
        margin: 0;
        padding: 0;
        background: #AAE74A url(images/img04.gif) repeat-x;
}

body, th, td, input, textarea {
        font-family: "Trebuchet MS", Arial, Helvetica, sans-serif;
        font-size: 13px;
        color: #666666;
}

p, ol, ul {
        line-height: 170%;
}
```

```
a:hover {
        text-decoration: none;
}

/*Header*/

#header {
        width: 960px;
        height: 120px;
        margin: 0 auto;
}

#logo {
        float: left;
        height: 120px;
        margin-left: lüpx;
        background: url(images/neulogo.png) no-repeat left center;
}

#logo hl {
        text-transform: uppercase;
}

#logo hl {
        margin: 0;
        padding: 25px 0 0 85px;
        letter-spacing: -2px;
        font-size: 3em;
        font-weight: normal;
        color: #000000;
}

#logo hl a {
        color: #000000;
}

#logo p {
        margin: -1Opx 0 0 2px;
        padding: 0 0 0 85px;
        /*text-transform: lowercase;* /
}

#search {
        float: right;
        width: 280px;
}

#search form {
```

```
        margin: 0;
        padding: 63px 0 0 0;
}

#search fieldset {
        margin: 0;
        padding: 0;
        border: none;
}

#search #s, #search #x {
        float: left;
}

#search #s {
        width: 188px;
        margin: 2px 6px 0 0;
        padding: 2px 5px;
        background: url(images/img02.gif) repeat-x;
        border: 1px solid #ACACAC;
}

#search #x {
        width: 67px;
        height: 28px;
        padding: 0;
        background: #006BFF url(images/img03.gif) no-repeat;
        border: none;
        text-transform: lowercase;
        color: #FFFFFF;
}


/*Menu*/

#menu {
        width: 962px;
        height: 50px;
        margin: 0 auto;
}

#menu ul {
        margin: 0;
        padding: 0;
        list-style: none;
}

#menu li {
        display: block;
```

```
        float: left;
}

#menu a {
        display: block;
        float: left;
        height: 38px;
        padding: 8px 20px 0 20px;
        text-decoration: none;
        text-transform: lowercase;
        color: #000000;
}

#menu a:hover {
        text-decoration: underline;
}

#menu .current_page _item {
        background: url(images/img05.gif) no-repeat;
}

#menu .current_page _item a {
        background: url(images/img06.gif) no-repeat right top;
        font-weight: bold;
}

/* Page*/

#page {
        width: 962px;
        margin: 0 auto;
        background: #FFFFFF url(images/img07.gif) repeat-y;
}

/* Content */

#content {
        float: right;
        width: 700px;
        padding: 11px 11px 0 5px;
        background: url(images/img09.gif) no-repeat;
}

#content a {
        color: #FF8900;
}

.pagetitle {
        height: 33px;
```

```
        margin: 0;
        padding: 8px 0 0 15px;
        background: url(images/imgl3.git)   no-repeat;
        font-size: l.4em;
        color: #FFFFFF;
}

#content #rss-posts {
        display: block;
        margin: -30px 15px Opx Opx;
        padding: 0 20px 0 0;
        background: url(images/rss.git)  no-repeat right center;
        text-align: right;
        font-weight: bold;
        color: #FFFFFF;
}

.post {
        padding: 40px 30px 0 30px;
}

.title {
        margin: 0;
        font-size: 2.4em;
        font-weight: normal;
}

.byline {
        margin: 0 0 20px 0;
}

.meta {
        border-top: 1px dotted #CCCCCC;
        text-align: right;
}

.meta .more, .meta .comments {
        padding-left:  15px;
        background: url(images/img14.git)  no-repeat left center;
}

/* Sidebar*/

#sidebar {
        float: left;
        width: 230px;
        padding: llpx 5px 0 llpx;
        background: url(images/img08.git)  no-repeat;
}
```

```
#sidebar ul {
        margin: 0;
        padding: 0;
        list-style: none;
}

#sidebar li {
        margin-bottom: 20px;
}

#sidebar li ul {
        padding: 1Opx 15px;
}

#sidebar Ii li {
        margin: 0;
        padding-left: 15px;
        background: url(images/imgl2.git)  no-repeat left center;
}

#sidebar h2 {
        height: 33px;
        margin: 0;
        padding: 8px 0 0 15px;
        background: url(images/imgl 1.git) no-repeat;
        font-size: 1.4em;
        color: #FFFFFF;
}

#sidebar a {
        text-decoration: none;
        color: #0065FF;
}

#sidebar a:hover {
        text-decoration: underline;
}

/* Footer */

#footer {
        width: 962px;
        margin: 0 auto;
        padding: 30px 0;
        background: url(images/imglO.git)  no-repeat;
}

#footer p {
```

```
        margin:   0;
        text-align:    center;
        color:  #FFFFFF;
}

#footer a {
        color: #FFFFFF;
}
```

LogOut.jsp

```
<%@page contentType="text/html;charset=iso-8859-91'.pageEncoding="iso-8859-9"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http ://www.w3.org/TR/html4/loose.dtd">
<%@ taglib uri="http://java.sun.com/jstl/core   _rt" prefix="c"  %>
<html>
<head>
<meta http-equiv="Content-Type"   content="text/html;   charset=iso-8859-9">
<title>Logging  Out!! !</title>
</head>
<body>

<c:set var="userID"  value=""  scope="session"/>
<c:set var="userName"  value=""  scope="session"  />
<c:set var="status"  value="O" scope="session"   />
<c:set var="currentTerm"   value=""  scope="session"  />
<c:set var="termDesc"   value=""  scope="session"/>
<c :redirect url="/ student"></ c:redirect>

</body>
</html>
```

menu.jsp

```
<%
String[][] menu = {
                {"home","Home"},
                {"student", "Student"},
                {"lecture", "Lecture"}
};

String cl;

for(int i=O;i<menu.length;i++ ){
        if(menu[i] [0].equals( (String)request.getParameter("page")))    {
                cl=  "class='current_page_item'";
        }else{
                cl="".
        }
```

```
        out.println("<li  "+cl+"><a  href=' ./?page="+menu[i]  [O]+">"+menu[i] [1]+"</a></li>");
}
%>
```

usermenu.jsp

```
<%@ taglib  uri="http://java.sun.com/jstl/core_rt"    prefix="c"  %>

<c:iftest="${sessionScope.status     != 1 or sessionScope.status  == null}">
        <form method="post"  action="./control.jsp">
                <fieldset>
                        ID:<input  id="u" type="text"  name="userID"  value="" />
                        Password:<input  id="u" type="password"  name="pswrd"  value=""/>
                        <p><center><input  id="sb" type="submit"  value="Log  In" /></center>
                </fieldset>
        </form>
<zc.if>
<c:if test="$ {sessionScope.status  == 1}">
        ID:  ${sessionScope.userID}<br>
        Name:  ${sessionScope.userName}<br>
        Status : <c:if test="${ sessionScope.status==  1}">Student</c:if><br>
        Current  Term:  ${sessionScope.termDesc}<br>
        <a href="./logüut.jsp">Log    out</a>
</c:if>
```

/student/catagory/

announces.jsp

```
<%@taglib   uri="http://java.sun.com/jstl/core_rt"    prefix="c"  %>

<%@page  import="application  Work.announces"%>
<%@page  import="j ava. sql.ResultSet"%>
<%@page  import="org.omg.CORBA.Request"%>
<table  border=" 1">
  <tr>
    <th>From</th>
    <th>Date</th>
    <th>Content</th>
  </tr>

  <%
        announces  ann = new  announces();
        ResultSet  rs = ann.getAnnounceinfo(String.   valueüf(session.getAttribute("userID     ")));
        try{
                while(rs.next()){
                        out.println("<tr>");
                        out.println("<td>"+rs.getString("Name")+"      "+rs.getString("Sumame")+"</td>");
                        out.println("<td>"+rs.getString("theDate")+"</td>");
                        out.println("<td>"+rs.getString("content")+"</td>");
```

```
                        out.println("</tr>");
                }
        }catch(Exception   e){

        }
  %>
</table>
```

grades.jsp

```
<%@page   language="java"   contentType="text/html;   charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE  html PUBLIC  "-//W3C//DTD  HTML  4.01  Transitional/JEN"
"http://www.w3.org/TR/html4/loose.dtd   ">
<%@page  import="application  Work.*"%>
<%@page  import="java.sql.  *"%>
<%
        student  st = new  student();
        String userID  = String.valueOf(session.getAttribute("userID"));
        ResultSet  rs=  st.grades(userID);
%>
<center>
<table border=" 1 ">
<%while(rs.next()){   %>
        <tr>
                <td><b><%=rs.getString("TermDesc")%></b></td>
                <td><b><%=rs.getString("CourseID    ")%></b></td>
                <td><b><%==rs.getString("Result")%></b></td>
        </tr>
<%}%>
</table>
</center>
```

index.jsp

```
<%@taglib   uri="http://java.sun.com/jstl/core_rt"    prefix="c"  %>

<c:if test="$ {sessionScope.status   =   1} ">
        Here, you can find your information,Timetable   and more. For more information,  please visit the
tabs above.
</c:i:f>
```

information.jsp

```
<%@page  import="application  Work.*"%>
<%@ taglib  uri="http://java.sun.com/jstl/core   _rt" prefix="c"  %>
<%@page  import='Java.sql.  *"%>
<%
student  st=  new  student();
```

```
if(String.valueüf(request.getParameter("change")).equals("l                ")){
        String  StudentID   = String.valueOf(session.getAttribute("userID"));
        String  Adress   = String.valueüf(request.getParameter("Adress"));
        String  Phone=   String.valueOf(request.getParameter("Phone"));
        String  EMail  = String.valueOf(request.getParameter("EMail"));
        if( st.updateinformation(StudentID        ,Adress,Phone,EMail))      {
                out.println("Changed!      ");
        }
}

String userID = String.valueüf(session.getAttribute("userID"));
ResultSet  rs=  st.viewinformation(userID);
while(rs.next()){
        try{
                out.println("<form   action="  method='get'><table>");
                out.println("<input   type='hidden'  name='category'   value='information'/><input
type='hidden'  name='page'  value='student'/>");
                out.println("<input   type='hidden'  name='change'   value='!'/>");
                out.println("<tr><td>Number:</td><td>"+rs.getString("ST.StudentID")+"</td></tr>");
                out.println("<tr><td>Name   & Sumame:</td><td>"+rs.getString("ST.Name")+"
"+rs.getString("ST.Sumame")+"</td></tr>");
                out.println("<tr><td> Birth Date:</td><td> "+rs.getString(" ST.Birthdate")+ "</td></tr> ");
                out.println("<tr><td>Adress:</td><td>     <input type='text'  name='Adress'
value='"+rs.getString("ST.Adress")+"'    />"+"</td></tr>");
                out.println("<tr><td>Phone:</td><td><input     type='text'  name='Phone'
value='"+rs.getString("ST.Phone")+"'    />"+"</td></tr>");
                out.println("<tr><td>E-Mail:</td><td><input     type='text'  name='EMail'
value='"+rs.getString("ST.EMail")+"'    />"+"</td></tr>");
                out.println("<tr><td>Started   Term:</td><td>"+rs.getString("TermDesc")+    "</td></tr> ");
                out.println("<tr><td>Advisor:</td><td>"+rs.getString("INS.Name")+"
"+rs.getString("INS.Sumame")+"</td></tr>");

        out.println("<tr><td>Major:</td><td>"+rs.getString("MJ.MajorDesc")+"</b>"+"</td></tr>");
                out.println("<tr><td   colspan='2'><input   type='submit'  value='Save  changes'
/></td></tr>");
                out.println("</table></form>");
        }catch(Exception   e){
                out.println( e.to String());
        }
}
rs.close();
%>
```

myDoc.jsp

```
<%@page  import="application  Work.*"%>
<%@page  import="java.sql.  *"%>
<table  border=" I">
```

```
        <tr>
                <th>File Name</th>
                <th>Date</th>
                <th> View</th>
        </tr>
<%
String StudentID = String.valueüf(session.getAttribute("userID"));
student st= new studenu);
ResultSet rs= st.getDocuments(StudentID);
try{
        while( rs.next()){
%>
        <tr>
                <td><%=rs.getString("D0C.fıleName")%></td>
                <td><%=rs.getString("D0C.   theDate ")%><ltd>
                <td><a
href="./category/viewDoc.jsp?DocID=<%=rs.getString("D0C.DocID")%>">View</a></td>
        </tr>
<%} }catch(Exception   e){

}       %>

</table>
```

schedule.jsp

```
<%@page import="application Work.student"%>
<%@page import="java.sql.ResultSet"%>
<table border=" l ">
        <tr>
                <th>CourseID</th>
                <th>Date</th>
                <th>Building</th>
                <th>Room</th>
                <th>Description</th>
        </tr>
        <%
        student st = new studenn);
        String StudentID = String.valueüf(session.getAttribute("userID"));
        String currentTerm = String.valueOf(session.getAttribute("currentTerm    "));
        ResultSet rs= st.getSchedule(StudentID,currentTerm);
        try{
                while(rs.next()){
                        out.println("<tr>");
                        out.println("<td>"+rs.getString("C.CourseID")+"-
"+rs.getString("C.Section")+"</td>");
                        out.println("<td>"+rs.getString("S.theDate")+"</td>");
                        out.println("<td>"+rs.getString("B.BuildingName")+"</td>");
                        out.println("<td>"+rs.getString("L.RoomNo")+"</td>");
```

```
                        out.println("<td>"+rs.getString("S.ScDesc")+"</td>");
                        out.println("</tr>");
                }
        }catch(Exception   e){
                out.println(e.toString());
        }
        %>
</table>
```

viewDoc.jsp

```
<%@page  import="java.sql.  *"%>
<%@page  import="application  Work.*"%>
<%
String DoclD  = (String)request.getParameter("DocID");
String StudentID  = (String)session.getAttribute("userID");
student  st=  new  student();

ResultSet  rs=  st.getDocument(StudentID,Integer.parseint(DocID));
try{
        while( rs.next()) {
                Blob  blob;
                byte[]  bytes;
                blob=  rs.getBlob("doc");
                bytes=  blob.getBytes(l,   (int)blob.length());

                response.setContentType(rs.getString("contentType"));

                response:setContentLength(bytes.length);
           response. getüutputStream().write(bytes    );
                response.flushBuffer();

        }
}catch(Exception   e){

}
rs.close();
%>
```

Appendix B

instructor

index.jsp

```
<%@ page contentType="text/html;charset=iso-8859-9"    pageEncoding="iso-8859-9"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http ://www.w3.org/TR/html4/loose.dtd ">
<%@ taglib uri="http://java.sun.com/jstl/core _rt" pre:fix="c" %>
<html>
<head>
<rrieta http-equiv="Content-Type"  content="text/html;  charset=iso-8859-9">
<title>Student Management System (S-M-S)</title>
<meta name="keywords"  content="" />
<meta name="description"  content=""/>
<link href="default.css"  rel="stylesheet"  type="text/css"  media="screen"  />
</head>

<body>
<!-- start header-->
<div id="header">
        <div id="logo">
                <hl>SMS</hl>
                <p>Student Management System</p>
        </div>
        <div id="search">
                <form method="get'·' action="">
                        <fieldset>
                        <input id="s" type="text" name="s" value=""/>
                        <input id="x" type="submit" value="Search" />
                        </fieldset>
                </form>
        </div>
</div>
<!-- end header-->

<!-- start menu -->
<div id="menu">
        <ul>
                <c:import url="./menu.jsp" />
        </ul>
</div>
<!-- end menu-->
```

```
<!-- start page -->
<div id="page">

        <!-- start content -->
        <div id="content">
                <hl class="pagetitle">Contents</hl>
                <div class="post">
                        <c:import url=" ./content.jsp" />
                </div>
        </div>
        <!-- end content-->

        <!-- start sidebar -->
        <div id="sidebar">
                <ul>
                        <c:iftest="${sessionScope.status == 2}">
                        <Ii>
                                <h2>Categories</h2>
                                <ul>
                                        <c:import url=" ./categories.jsp" />
                                <Jul>
                        <ili>
                        </c:if>
                        <Ii>
                                <h2>User Menu</h2>
                                <ul>
                                        <c:import url="./userMenu.jsp" />
                                </ul>
                        </li>
                </ul>
        </div>
        <!-- end sidebar-->
        <div style="clear: both;"> </div>
</div>
<!-- end page-->
<div id="footer">
        <p>&copy;2008 All Rights Reserved.  &bull;  Design by Muhammed Emin ER
</div>
</body>
</html>
```

categories.jsp

```
<%@taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
```

```
<c:iftest="${param.page = 'personal'}">
        <li><a href=" ./?page=personal&category=personallnfo ">Information</ a></li>
        <li><a href="#">Timetable</a></li>
</c:if>
<c:if test="${param.page == 'student'}">
        <li><a href=" ./?page=student&category=selectStudent">Select Student</a></li>
        <li><a href=" ./?page=student&category=studentlnformation ">lnformation</a></li>
        <li><a href=" ./?page=student&category=grades ">Grades</a></li>
        <li><a href="#">Timetable</a></li>
        <li><a href="./?page=student&category=stDoc">Documents</a></li>
        <li><a href="./?page=student&category=uploadDoc">Upload Documents</a></li>
</c:if>
<c:iftest="${param.page = 'lecture'}">
        <li><a href=" ./?page=lecture&category=grades ">Grades</ a></li>
        <li><a href="./?page=lecture&category=timeTable">Time Table</a></li>
        <li><a href="#">Exam Calendar</a></li>
</c:if>
```

configure.jsp

```
<%@taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>

<c:set var="currentTerm" value="SP08" scope="session"/>
<c:set var="termDesc" value="Spring 2008" scope="session"/>
```

content.jsp

```
<%@ taglib uri="http://java.sun.com/jstl/core _rt" prefix="c" %>

<c:,iftest="${param.category = null}">
        <c:import url="./category/index.jsp" />
</c:if>
<c:iftest="${param.category != null and param.category!="}">
        <c:import url="./category/${param.category}.jsp" />
</c:if>

<c:if test="${sessionScope.status = 0 || sessionScope.status ==null}">
        You should "log in"!..
</c:if>
```

control.jsp

```
<%@ page contentType="text/html;charset=iso-8859-9" pageEncoding="iso-8859-9"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<%@taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
<%@page import="application Work.*"%>
<%@page import="java.sql. *"%>
<html>
<head>
```

```
<meta http-equiv="Content-Type"   content="text/html;   charset=iso-8859-9">
<title>Controlling   Password</title>
</head>
<body>

<%
String userID = (String)request.getParameter("userID");
String pswrd = (String)request.getParameter("pswrd");

security sc = new security();
int status = sc.getStatus(userID,pswrd);
if(status =  2){
        //if status = 2
        instructor ins= new instructor();
        ResultSet rs= ins.viewlnformation(userID);
        try{
                while(rs.next()){
        %>
        <c:set var="userID"  value="<%=rs.getString("INS.lnstructorID")     %>" scope="session"/>
        <c:set var="userN ame" value="<%=rs.getString("INS   .Name")+"
"+rs.getString("INS.Sumame")   %>" scope="session"  />
        <c:set var="status"  value="2"  scope="session"   />
        <c:import  url="./configure.jsp"   />
        <%      }//end while
        }//end try
        catch(Exception   e){

        }
}else{
        //id status  != 2
        %>
        <c:set var="userID"  value=""  scope="session"/>
        <c:set var="userName"   value=""  scope="session"  />
        <c:set var="status"  value="O" scope="session"   />
        <c:set var="currentTerm"   value=""  scope="session"  />
        <c:set var="termDesc"   value=""  scope="session"/>
        <%
}
%>
<c :redirect url="/instructor"></  c:redirect>

</body>
</html>

default.css

body {
        margin:  0;
        padding:  0;
```

```
        background: #AAE74A url(images/img04.gif) repeat-x;
}

body, th, td, input, textarea {
        font-family: "Trebuchet MS", Arial, Helvetica, sans-serif;
        font-size: 13px;
        color: #666666;
}

p, ol, ul {
        line-height: 170%;
}

a:hover {
        text-decoration: none;
}

/*Header*/

#header {
        width: 960px;
        height: 120px;
        margin: 0 auto;
}

#logo {
        float: left;
        height: 120px;
        margin-left: 10px;
        background: url(images/neulogo.png) no-repeat left center;
}

#logo hl {
        text-transform: uppercase;
}

#logo hl {
        margin: 0;
        padding: 25px 0 0 85px;
        letter-spacing: -2px;
        font-size: 3em;
        font-weight: normal;
        color: #000000;
}

#logo hl a {
        color: #000000;
}
```

```
#logo p {
        margin: -1Opx 0 0 2px;
        padding: 0 0 0 85px;
        /*text-transform: lowercase;*/
}

#search {
        float: right;
        width: 280px;
}

#search form {
        margin: 0;
        padding: 63px 0 0 0;
}

#search fieldset {
        margin: 0;
        padding: 0;
        border: none;
}

#search #s, #search #x {
        float: left;
}

#search #s {
        width: 188px;
        margin: 2px 6px 0 0;
        padding: 2px 5px;
        background: url(images/img02.git) repeat-x;
        border: lpx solid #ACACAC;
}

#search #x {
        width: 67px;
        height: 28px;
        padding: 0;
        background: #006BFF url(images/img03.git) no-repeat;
        border: none;
        text-transform: lowercase;
        color: #FFFFFF;
}


/* Menu*/

#menu {
        width: 962px;
```

```
        height: 50px;
        margin: 0 auto;
}

#menu ul {
        margin: 0;
        padding: 0;
        list-style: none;
}

#menu li {
        display: block;
        float: left;
}

#menu a {
        display: block;
        float: left;
        height: 38px;
        padding: 8px 20px 0 20px;
        text-decoration: none;
        text-transform: lowercase;
        color: #000000;
}

#menu a:hover {
        text-decoration: underline;
}

#menu .current_page _item {
        background: url(images/img05.gif) no-repeat;
}

#menu .current_page_item a {
        background: url(images/img06.gif) no-repeat right top;
        font-weight: bold;
}

/* Page*/

#page {
        width: 962px;
        margin: 0 auto;
        background: #FFFFFF url(images/img07.gif) repeat-y;
}

/* Content */

#content {
```

```
        float: right;
        width: 700px;
        padding: 11px 11px 0 5px;
        background: url(images/img09.gif) no-repeat;
}

#content a {
        color: #FF8900;
}

.pagetitle {
        height: 33px;
        margin: 0;
        padding: 8px 0 0 15px;
        background: url(images/img13.gif) no-repeat;
        font-size: 1.4em;
        color: #FFFFFF;
}

#content #rss-posts {
        display: block;
        margin: -30px 15px Opx Opx;
        padding: 0 20px 0 0;
        background: url(images/rss.gif) no-repeat right center;
        text-align: right;
        font-weight: bold;
        color: #FFFFFF;
}

.post {
        padding: 40px 30px 0 30px;
}

.title {
        margin: 0;
        font-size: 2.4em;
        font-weight: normal;
}

.byline {
        margin: 0 0 20px 0;
}

.meta {
        border-top: lpx dotted #CCCCCC;
        text-align: right;
}

.meta .more, .meta .comments {
```

```
        padding-left:  15px;
        background:  url(images/img14.gif)  no-repeat  left center;
}

/* Sidebar */

#sidebar {
        float: left;
        width: 230px;
        padding: 11px 5px 0 11px;
        background:  url(images/img08.gif)  no-repeat;
}

#sidebar ul {
        margin: 0;
        padding: 0;
        list-style: none;
}

#sidebar li {
        margin-bottom:  20px;
}

#sidebar li ul {
        padding: 10px 15px;
}

#sidebar li li {
        margin: 0;
        padding-left:  15px;
        background:  url(images/img12.gif)  no-repeat  left center;
}

#sidebar h2 {
        height: 33px;
        margin: 0;
        padding: 8px 0 0 15px;
        background:  url(images/img 11.gif) no-repeat;
        font-size: 1.4em;
        color: #FFFFFF;
}

#sidebar a {
        text-decoration:  none;
        color: #0065FF;
}

#sidebar a:hover {
        text-decoration:  underline;
```

```
}

/* Footer*/

#footer {
        width: 962px;
        margin: 0 auto;
        padding: 30px 0;
        background: url(images/imglO.gif) no-repeat;
}

#footer p {
        margin: 0;
        text-align: center;
        color: #FFFFFF;
}

#footer a {
        color: #FFFFFF;
}
```

logOut.jsp

```
<%@ page contentType="text/html;charset=iso-8859-9" pageEncoding="iso-8859-9"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional/JEN"
"http://www.w3.org/TR/html4/loose.dtd ">
<%@taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
<html>
<head>
<meta http-equiv=l'Content-Type" content="text/html; charset=iso-8859-9">
<title>Logging Out!! !</title>
</head>
<body>

<c:set var="userID" value="" scope="session"/>
<c:set var="userName" value="" scope="session" />
<c:set var="status" value="O" scope="session" />
<c:set var="currentTerm" value="" scope="session" />
<c:set var="terın.Desc" value="" scope="session"/>
<c:set var="studentNumber" value="" scope="session"/>
<c:redirect url="/instructor"></c:redirect>

</body>
</html>
```

menu.jsp

```
<%
String[][] menu = {
```

```
                {"home","Home"},
                {"personal", "Personal"},
                {"student","Manage  Student"},
                {"lecture", "Lectures"}
};

String cl;

for(int i=O;i<menu.length;i++){
        if(menu[i][O].equals((String)request.getParanieter("page"))){
                cl=  "class='current_page_item'";
        }else{
                cl="".
        }
        out.println("<li  "'+cl+"'><a  href=' ./?page="+menu[i]  [O]+"'>"+menu[i] [ ]+"</a></li>");
}
%>
```

userMenu.jsp

```
<%@taglib uri="http://java.sun.com/jstl/core_rt"    prefix="c"  %>
<c:iftest="${sessionScope.status     != 2 or sessionScope.status  ==null}">
        <form method="post"  action="./control.jsp">
                <fieldset>
                        ID:<input  id="u" type="text"  name="userID"  value="" />
                        Password:<input  id="u" type="password"  name="pswrd"  value=""/>
                        <p><center><input  id="sb" type="submit"  value="Log  In" /></center>
                </fieldset>
        </form>
</c:it>
<c:iftest="${sessionScope.status    =  2}">
        ID : ${sessionScope.userID}<br>
        Name : ${sessionScope.userName  }<br>
        Status : <c:if test="$ {sessionScope.status==2}  ">Instructor</c:it><br>
        Current Term:  ${sessionScope.termDesc}<br>
        <a href="./logüut.jsp">Log    out</a>
</c:it>
```

/instructor/category

deletefroes.jsp

```
<%@page  import="java.sql.  *"%>
<%@page  import="application  Work.*"%>
<%
ResultSet  rs;
String[]  docs;
String userID  = String.valueOf(session.getAttribute("userID"));
instructor  ins = new  instructor();
```

```
docs= request.getParameterValues("docs");

if(docs  != null){
        for(int  i=O;i<docs.length;i++)  {
                try{
                        rs=  ins.getDocument(Integer.parselnt(   docs[i]));
                        if(rs.next()){
                                out.println("Document:   "+rs.getString("fileName"));
                                if(ins.deleteDocs(Integer.  parse Int( docs[i]), user ID)) {
                                        out.println("  deleted.<br>");
                                }else{
                                        out.println(",  you don't have permissions  for delete this document.
<hr>");
                                }
                        }
                }catch(Exception   e){
                        out.println( e.toString());
                }
        }
}else{
        out.println("You  did not select any documents  for deleted!");
}
%>
```

grades.jsp

```
<%@page   Ianguage=tjava"   contentType="text/html;   charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE  html PUBLIC  "-//W3C//DTD  HTML  4.01  Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<%@ taglib  uri="http://java.sun.com/jstl/core_rt"    prefix="c"  %>
<%@page  import="application  Work.*"%>
<%@page  import="java.sql.  *"%>

<c:if test="${ (sessionScope.studentNumber    != ") and  (sessionScope.studentNumber    !=null)}">
        <center>Selected   Student  is:
        <c:import  url=" ./category/studentlnfo.jsp"   /></center>
        <br><br>
        <%
        student  st = new  student();
        String userID  = String.valueOf(session.getAttribute("studentNumber"));
        ResultSet  rs=  st.grades(userID);
        %>
        <center>
        Grades:
        <table  border=" l ">
        <%while(rs.next()){   %>
                <tr>
                        <td><b><%=rs.getString("TermDesc")%></b></td>
```

```
                        <td><b><%=rs.getString("CourseID")%></b></td>
                        <td><b><%=rs.getString("Result")%></b></td>
                </tr>
        <%}%>
        </table>
        </center>
<re.if>
```

personallnfo.jsp

```
<%@page import="application Work.*"%>
<%@taglib uri="http://java.sun.com/jstl/core_rt"  prefix="c" %>
<%@page import="java.sql. *"%>
<%
instructor ins= new instructor();

if(String. valueüf( request.getl'arameterf'change  ")).equals(" l ")){
        String StudentID = String.valueOf(session.getAttribute("userID"));
        String Adress = String.valueOf(request.getParameter("Adress"));
        String Phone= String.valueOf(request.getParameter("Phone"));
        String EMail = String.valueOf(request.getParameter("EMail"));
        if(ins. updatelnformation(StudentID ,Adress,Phone,EMail)) {
                out.println("Changed! ");
        }
}

String userID = String.valueOf(session.getAttribute("userID"));
ResultSet rs= ins.viewlnformation(userID);
while(rs.next()){
        try{
                out.println("<form action=" method='get'><table>");
                out.println("<input type='hidden' name='category' value='personallnfo'/><input
type='hidden' name='page' value='personal'/>");
                out.println("<input type='hidden' name='change' value='l'/>");

        out.println("<tr><td>UserID:</td><td>"+rs.getString("INS.InstructorID")+"</td></tr>");
                out.println("<tr><td>Name & Surname:</td><td>"+rs.getString("INS.Name")+"
"+rs.getString("INS.Surname")+"</td></tr>");
                out.println("<tr><td>Birth
Date:</td><td>"+rs.getString("INS.Birthdate")+"</td></tr>");

        out.println("<tr><td>Department:</td><td>"+rs.getString("DEPT.DeptDesc")+"</td></tr>");
                out.println("<tr><td>Room:</td><td>"+rs.getString("BUI.BuildingName")+",  Room
No: "+rs.getString("LOC.RoomN o")+"</td></tr> ");
                out.println("<tr><td> Room
Phone:</td><td>"+rs.getString("INS.RPhone")+"</td></tr>");
                out.println("<tr><td>Adress:</td><td>  <input type='text' name='Adress'
value='"+rs.getString("INS.Adress")+"  />"+"</td></tr>");
                out.println("<tr><td>Phone:</td><td><input  type='text' name='Phone'
```

```
value='"+rs.getString("INS.Phone")+"'     />"+"</td></tr>");
                    out.println("<tr><td>E-Mail:</td><td><input     type='text'  name='EMail'
value='"+rs.getString("INS.EMail")+"'     />"+"</td></tr>");
                    out.println("<tr><td   colspan='2'><input   type='submit'  value='Save  changes'
/></td></tr> ");
                    out.println("</table></form>");
          }catch(Exception   e){
                    out.println( e.toString() );
          }
}
rs.close();
%>
```

selectStudent.jsp

```
<%@taglib   uri="http://java.sun.com/jstl/core_rt"     prefix="c"  %>
Select  Student:
<br>
<%
out.println("<form   action="  method='get'>");
out.println("<input   type='hidden'  name='category'   value='selectStudent'/><input    type='hidden'
name='page'  value='student'/>");
out.println("<input   type='text'  name='studentNumber'   />");
out.println("<input   type='submit'  name='submit'  value='Search'  />");
out.println("</form>");
%>
<c:iftest="${(param.studentNumber      !=")and   (param.studentNumber    != null)}">
          <c:set  var=" studentNumber"   value="$ {param.studentNumber}"    scope=" session"/>
</c:if>
<c:iftest="${(sessionScope.studentNumber      !=")and   (sessionScope.studentNumber    != null)}">
          <br><br>Selected   Student  is:
          <c:import  url="./category/studentlnfo.jsp"    />
</c:if>
```

stDoc.jsp

```
<%@tag1ib·uri="http://java.sun.com/jstl/core_rt"     prefix="c"  %>
<%@page  import="application  Work.*"%>
<%@page  import="java.sql.*"%>
<c:import  url="./category/studentlnfo.jsp"    />
<br><br>
<form  action=""  method="get">
<table  border=" 1 ">
          <tr>
                    <th>Uploader</th>
                    <th>File  Name</th>
                    <th>File  Size(byte)</th>
```

```
                <th>Date</th>
                <th> View File</th>
                <th>Delete  Selected</th>
        </tr>
<%
String studentNumber  = String.valueüf(session.getAttribute("studentNumber"));
String userlD  = String.valueüf(session.getAttribute("userID"));
instructor  ins = new  instructor();
ResultSet  rs=  ins.getStDoc(studentNumber,userlD);
try{
while(rs.next())  {
%>
        <tr>
                <td><%=rs.getString("INS.Name")+"     "+rs.getString("INS.Surname")    %></td>
                <td><%=rs.getString("fıleN   ame ")%></td>
                <td><%=rs.getString("fileSize")%></td>
                <td><%=rs. getString("theDate  ")%></td>
                <td><a
href="./category/viewDoc.jsp?DocID=<%=rs.getString("DocID")%>">View</a></td>

                <td><center><input   type="checkbox"   name="docs"
value="<%=rs.getString("DocID    ")%>" /></center></td>
        </tr>
<%} }catch(Exception   e){}%>
        <tr>
                <td></td>
                <td></td>
                <td></td>
                <td></td>
                <td></td>
                <td><input  type="submit"   value="Delete   Selected"/></td>
        </tr>
</table>
        <input type="hidden"   name="page"   value="student"   />
        <input type="hidden"   name="category"   value="deleteDocs"   />
</form>
```

studentlnfo.jsp

```
<%@page  import="java.sql.ResultSet"%>
<%@page  import="application  Work.*"%>
<%
student st=  new  student();
String studentNumber  = String.valueüf(session.getAttribute("studentNumber"));
ResultSet  rs=  st.viewlnformation(studentNumber);
out.println("<br><table   border=l>");
while(rs.next()  ){
        try{
```

```
        out.println("<tr><td>StudentID   :</td><td> "+rs.getString(" ST.StudentID ")+"</td></tr> ");
                out.println("<tr><td>Name    & Sumame:</td><td>"+rs.getString("ST.Name")+"
"+rs.getString("ST.Sumame")+"</td></tr>");
                out.println("<tr><td>Birthdate:</td><td>"+rs.getString("ST.Birthdate")+"</td></tr>");
                out.println("<tr><td>Start   Term:</td><td>"+rs.getString("T.TermDesc")+"</td></tr>");
                out.println("<tr><td>  Advisor:</td><td>"+rs.getString("INS    .Name")+"
"+rs.getString("INS.Sumame")+"</td></tr>");
        }catch(Exception   e){

        }
}
rs.close();
out.println("</table>");
%>
```

studentlnformation.jsp

```
<%@page  import="application  Work.*"%>
<%@ taglib  uri="http://java.sun.com/jstl/core   _rt" prefix="c"  %>
<%@page  import="java.sql.*"%>
<%
student  st=  new  student();

if(String.valueOf(request.getParameter("change")).equals("l      ")){
        String  StudentID  = String.valueOf(session.getAttribute("studentNumber"));
        String.Adress  = String.valueüf(  request.getParameter("Adress   "));
        String Phone=  String.valueüf(request.getParameter("Phone"));
        String EMail  = String.valueOf(request.getParameter("EMail"));
        if(st.updatelnformation(StudentID   ,Adress,Phone,EMail))  {
                out.println("Changed!   ");
        }
}

String studentNumber  = String.valueüf( session. getAttribute(" studentNumber"));
ResultSet  rs = st.viewlnformation(studentNumber);
while(rs.next() ){
        try{
                out.println("<form   action="  method=' get'><table>");
                out.println("<input   type='hidden'  name='category'   value='studentlnformation'/><input
type='hidden'  name='page'  value='student'/>");
                out.println("<input   type='hidden'  name='change'   value=' l'/>");
                out.println("<tr><td>  Number:</td><td>"+rs.getString("    ST.StudentID ")+"</td></tr>");
                out.println("<tr><td>Name    & Sumame:</td><td>"+rs.getString("ST.Name")+"
"+rs.getString("ST.Surname")+"</td></tr>");
                out.println("<tr><td>Birth    Date:</td><td>"+rs.getString("ST.Birthdate")+"</td></tr>");
                out.println("<tr><td>Adress:</td><td>     <input type='text'  name='Adress'
value='"+rs.getString("ST.Adress")+"'     />"+"</td></tr>");
                out.println("<tr><td>Phone:</td><td><input     type='text'  name='Phone'
value='"+rs.getString("ST.Phone")+"'     />"+"</td></tr>");
```

```
                out.println("<tr><td>E-Mail:<ltd><td><input     type='text'  name='EMail'
value='"+rs.getString("ST.EMail")+"'    l>"+"<ltd><ltr>");
                out.println("<tr><td>Started    Term:<ltd><td>"+rs.getString("TermDesc")+"<ltd><ltr>");
                out.println("<tr><td>Advisor:<ltd><td>"+rs.getString("INS.Name")+"
"+rs.getString("INS.Surname")+"<ltd><ltr>");

        out.println("<tr><td>Major:<ltd><td>"+rs.getString("MJ.MajorDesc")+"<lb>"+"<ltd><ltr>");
                out.println("<tr><td   colspan='2'><input   type='submit'  value='Save  changes'
l><ltd><ltr>");
                out.println("<ltable><lform>");
        }catch(Exception   e){
                out.println( e.toString());
        }
}
rs.close();
%>
```

uploadlroc.jsp

```
<%@taglib  uri="http:lljava.sun.comljstllcore_rt"    prefix="c"  %>
<%@page  language="java"   import="javazoom.upload.   *,java.util. *" %>
<%@page  import="java.sql.  *"%>
<%@pageimport="java.io.    *"%>
<%@page  import="java.awt.Choice"%>
<jsp:useBean  id="upBean"  scope="page"  class="javazoom.upload.UploadBean"    >
  <jsp:setProperty  name="upBean"  property="folderstore"   value="c:luploads"  />
<ljsp:useBean>

<c:if test="${ (sessionScope.studentNumber    !=")and   (sessionScope.studentNumber    !=null)}">
        <center>
        <c:import  url=" ./categorylstudentlnfo.jsp"    />
        <lcenter><br><br>
        <%
    if (MultipartF ormDataRequest.isMultipartF   ormData(request))
    {
      // Uses MultipartFormDataRequest   to parse the HTTP request.
      MultipartFormDataRequest   mrequest  = new  MultipartFormDataRequest(request);
      String todo = null;
      if (mrequest  != null)  todo = mrequest.getParameter("todo");
            if ( (todo  != null)  &&  (todo.equalslgnoreCase("upload")))
            {
           Hashtable  files=  mrequest.getFiles();
           if ( (files  != null)  &&  (!files.isEmpty())   )
           {
              UploadFile  file=  (UploadFile)  files.get("uploadfile");
              InputStream  fis = file.getlnpuStream();
              if (file  != null)  out.println("<li>Form   field : uploadfile"+"<BR>  Uploaded  file :
"+file.getFileName()+"   ("+file.getFileSize()+"   bytes)"+"<BR>  Content  Type :
"+file.getContentType()  );
```

```
// Uses the bean now to store specified by jsp:setProperty at the top.
upBean.store(mrequest, "uploadfile");


String uploader = String.valueOf(session.getAttribute("userID"));
String toWho = String.valueüf( session. getAttribute(" studentNumber"));

boolean studentPer = true;
boolean instructorPer = true;
boolean staffPer = true;


int per= 0;
String permissions;
permissions = (String)request.getParameter("st");

out.println(permissions );

if(permissions != null){
  //for(int i = O;i<permissions.length;i++) {
  //      per+= Integer.parselnt(permissions[i]);
  //}
  switch(per){
        case 0:
              studentPer = false;
              instructorPer = false;
              staffPer = false;
              break;
        case 1:
              studentPer = true;
              instructorPer = false;
              staffPer = false;
              break;
        case 2:
              studentPer = false;
              instructorPer = true;
              staffPer = false;
              break;
        case 3:
              studentPer = true;
              instructorPer = true;
              staffPer = false;
              break;
        case 4:
              studentPer = false;
              instructorPer = false;
              staffPer = true;
              break;
        case 5:
```

```
                        studentPer = true;
                        instructorPer = false;
                        staflPer = true;
                        break;
                case 6:
                        studentPer = false;
                        instructorPer = true;
                        staflPer = true;
                        break;
                default:
                        studentPer = false;
                                instructorPer = true;
                                staflPer = true;
                                break;
            }

          }


          try{
            String sql = "INSERT INTO
DOCUMENTS(uploader,to Who,fileNarne,fileSize,doc,theDate,contentType,st,ins,stf)
VALUES('"+uploader+"' ,'"+to Who+"' ,'"+file.getFileNarne()+"', "+file.getFileSize()+", ?,'2008-03-
27','"+file.getContentType()+"', "+studentPer+", "+instructorPer+", "+staflPer+") ";
            Class.forNarne("corn.rnysql.jdbc.Driver").newlnstance();
            Connection conn =
DriverManager.getConnection("jdbc :rnysq1://localhost:3306/srns","root","");
            PreparedStaternent pstrnt = conn.prepareStaternent(sql);
            pstrnt.setBinaryStrearn( 1,fis,(in t)file.getFileS ize());
            pstrnt.executeUpdate();
            conn.close();
          }catch(Exception e){
            out.println("errooooo"+e.toString());
          }

        }
        else
        {
         out.println("<li>No uploaded files");
        }
        }
    else out.println("<BR> todo="+todo);
  }
%>

<form rnethod="post" action="" narne="upforrn" enctype="rnultipart/forrn-data">
 <table width="60%" border="O" cellspacing="l" cellpadding="l" align="center" class="stylel ">
  <tr>
   <td align="left"><b>Select a file to upload :</b></td>
```

```
    </tr>
    <tr>
     <td align="left">
      <input type="fıle" name="uploadfıle" size="50">
     <ltd>
    </tr>
    <tr>
     <td align="left">
        <input type="hidden" name="page" value="student" />
                <input type="hidden" name="category" value="uploadDoc" />
                <input type="hidden" name="todo" value="upload">
     <ltd>
    </tr>
    <tr>
     <td>
      <input type="checkbox" name="st" value=" l "/>Student<br>
      <input type="checkbox" name="ins" value="2"/>Instructor<br>
      <input type="checkbox" name="staff' value="4"/>Staff<br>
     <ltd>
    </tr>

    <tr>
     <td>
        <input type="submit" name="Submit" value="Upload">
      <input type="reset" name="Reset" value="Cancel">
     <ltd>
    </tr>
  </table>
</form>
</c:if>
```

viewDoc.jsp

```
<%@page import="java.sql. * "%>
<%@page import="application Work.*"%>
<%
String DocID = (String)request.getParameter("DoclD");
String userID = (String)session.getAttribute("userID");
instructor ins= new instructor();

if(ins.show Doc(user ID,DocID)) {
        ResultSet rs= ins.getDocument(Integer.parseint(DocID));
        try{
                while(rs.next() ){
                        Blob blob;
                        byte[] bytes;

                        blob = rs.getBlob("doc");
                        bytes= blob.getBytes(l,   (int)blob.length());
```

```
                    response.setContentType(  rs.getString(" contentType") );

                    response.setContentLength(bytes.length);
          response. getüutputStream().   write(bytes );
                    response.flushBu:ffer();

                }
        }catch(Exception   e){

        }
        rs.close();

}else{
        out.println("you   cannot  show  this  doc");
}

%>
```

APPENDIXC
Classes

database:

execute Update.java

```
package database;
import java.sql. *;

public class executeUpdate {
        dbConfigure Conf = new dbConfigure();
        public boolean update(String sql){
                boolean result= false;
                try{
                        ClassforName(Conf.getDriverName()).newlnstance();
                        Connection conn =
DriverManager.getConnection(Conf.getConnUrl(),Conf.getDBUser(),Conf.getDBPass());
                        Statement stmt = conn.createStatement();
                        stmt.execute Update( sql);
                        result = true;
                }catch (Exception e) {
                        result = false;
                }
                return result;
        }
}
```

executeQuery.java

```
package database;

import java.sql. *;

public class executeQuery {
        dbConfigure Conf = new dbConfigure();
        public ResultSet getRS(String sql){
                ResultSet RS = null;
                try{
                        ClassforName(Conf.getDriverName()).newlnstance();
                        Connection conn =
DriverManager.getConnection(Conf.getConnUrl(),Conf.getDBUser(),Conf.getDBPass());
                        Statement stmt = conn.createStatement();
                        RS = stmt.executeQuery(sql);
                }catch (Exception e) {

                }
```

```
                return RS;
        }
}
```

dbConfiguration.java

```
package database;

public class dbConfigure {
        private String DriverName = "com.mysql.jdbc.Driver";
        private String ConnUrl = "jdbc:mysql://localhost/sms";
        private String DBUser = "root";
        private String DBPass = "";

        public String getDriverName() {
                return DriverName;
        }
        public String getConnUrl() {
                return ConnUrl;
        }
        public String getDBUser() {
                return DBUser;
        }
        public String getDBPass() {
                return DBPass;
        }
}
```

student.java

```
package application Work;

import java.sql. *;

import database.*;

public class student {
        executeQuery eq;
        executeUpdate eu;
        public ResultSet viewlnformation(String StudentID){
                eq = new executeQuery();
                String sql = "SELECT * FROM STUDENT ST, TERM T,INSTRUCTOR INS, MAJOR
MJ WHERE ST.StudentID='"+StudentID+"' and T.TermID = ST.StartTerm and INS.InstructorID =
ST.AdvisorID and MJ.MajorID = ST.MajorID";
                ResultSet rs= eq.getRS(sql);
                return rs;
        }

        public boolean updatelnformation(String StudentID,String Adress, String Phone, String EMail){
```

```
                eu = new executeUpdate();
                String sql = "UPDATE STUDENT SET Adress='"+Adress+"', Phone='"+Phone+"',
EMail='"+EMail+"' WHERE StudentID='"+StudentID+"'";
                return eu.update(sql);
        }

        public ResultSet grades(String StudentID){
                eq = new executeQuery();
                String sql = "SELECT * FROM REGISTRATION RGS, CRSSECTIONCRS,TERM
TRWHERE RGS.StudentID='"+StudentID+"' and RGS.CsID = CRS.CsID and TR.TermID =
CRS.TermID";
                ResultSet rs= eq.getRS(sql);
                return rs;
        }

        public ResultSet getSchedule(String StudentID,String currentTerm){
                eq = new executeQuery();
                String sql = "SELECT * FROM CRSSECTION C,REGISTRATION R, SCHEDULE
S,LOCATION L, BUILDING B where C.TermID='"+currentTerm+"' and C.CsID = R.CsID and
S.CsID = C.CsID and R.StudentID='"+StudentID+"' and L.RoomID = S.RoomID and B.BuildingID =
L.BuildingID GROUP BY S.theDate";
                ResultSet rs= eq.getRS(sql);
                return rs;
        }

        public ResultSet getDocuments(String StudentID){
                ResultSet rs = null;
                eq = new executeQuery();
                String sql = null;
                sql = "SELECT * FROM DOCUMENTS DOC WHERE DOC.to Who='"+StudentID+"'
and DOC.st= true";
                rs = eq.getRS(sql);
                return rs;
        }

        public ResultSet getDocument(String StudentID,int DocID){
                eq = new executeQuery();
                String sql = "SELECT * FROM DOCUMENTS WHERE DocID = '"+DocID+"' AND
toWho= '"+StudentID+"' AND st= true";
                return eq.getRS(sql);
        }
}
```

security.java

```
package application Work;

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

```
public class security {
        executeQuery eq = new executeQuery();

        public int getStatus(String userID, String pswrd){
                int status= 0;
                ResultSet rs;

                String sql = "SELECT * FROM SECURITY WHERE userID='"+userID+"' AND
pswrd=MDS('"+pswrd+"')";
                rs= eq.getRS(sql);
                try {
                        while(rs.next() ){
                                status= Integer.parseint(rs.getString("status"));
                        }
                } catch (Exception e) {
                        status= 0;
                }
                return status;
        }
}
```

instructor.java

```
package application Work;
import j ava.sq l.ResultSet;
import database.executeQuery;
import database.executeUpdate;
public class instructor {
        executeQuery eq;
        executeUpdate eu;
        public ResultSet viewlnformation(String InstructorID){
                eq = new executeQuery();
                String sql = "SELECT * FROM INSTRUCTOR INS, DEPARTMENT
DEPT,LOCATION LOC,BUILDING BUI WHERE INS.InstructorID='"+InstructorID+"' AND
INS.DeptID = DEPT.DeptID AND LOC.RoomID = INS.RoomID AND BUI.BuildingID =
LOC.BuildingID";
                ResultSet rs= eq.getRS(sql);
                return rs;
        }


        public boolean updatelnformation(String InstructorID,String Adress, String Phone, String
EMail){
```

```
eu = new executeUpdate();
String sql = "UPDATE INSTRUCTOR SET Adress="'+Adress+'", Phone="'+Phone+'",
EMail="'+EMail+'" WHERE InstructorID="'+InstructorID+'"";
return eu.update(sql);
}

public ResultSet getStDoc(String StudentID, String InstructorID){
ResultSet rs = null;
eq = new executeQuery();
String sql = null;
sql = "SELECT * FROM DOCUMENTS F, INSTRUCTOR INS WHERE
F.toWho="'+StudentID+'" and INS.InstructorID="'+InstructorID+'" and (F.uploader="'+InstructorID+'"
or F.ins = true)";
rs= eq.getRS(sql);
return rs;
}

public boolean showDoc(String InstructorID,String DocID){
boolean show = false;
ResultSet rs;
eq = new executeQuery();
String sql = "SELECT * FROM DOCUMENTS WHERE DocID="+DocID+" and
(uploader="'+InstructorID+'" or ins=l)";
rs= eq.getRS(sql);
try{
if(rs.next()){
show=true;
}
}catch (Exception e) {
show = false;
}
return show;
}

public ResultSet getDocument(int DocID){
```

```
            eq = new executetjueryı);
            String sql = "SELECT * FROM DOCUMENTS WHERE DocID = '"+DocID+"'";
            return eq.getRS(sql);
        }

        public boolean deleteDocs(int DocID, String uploader){
            eu = new executeUpdate();
            String sql = "DELETE FROM DOCUMENTS WHERE DocID="+DocID+" and
uploader='"+uploader+"'";
            return eu.update(sql);
        }
}
```

announces.java

```
package application Work;
import java.sql. *;
import database.*;

public class announces {
        executeQuery eq = new executeQuery();

        public ResultSet getAnnounceInfo(String sendto) {
            ResultSet rs;
            String sql = "select
A.announcesID,A.status,A.sendfrom,A.senderStatus,A.theDate,A.content,I.Name,I.Surname from
ANNOUNCES A, INSTRUCTOR I WHERE A.sendfrom = I.InstructorID and A.sendto = '"+sendto+"'
UNION select
A.announcesID,A.status,A.sendfrom,A.senderStatus,A.theDate,A.content,S.Name,S.Sumame from
ANNOUNCES A, STAFF S where A.sendfrom = S.StafflD and A.sendto = '"+sendto+"'";
            rs= eq.getRS(sql);
            return rs;
        }
}
```

APPENDIXD
Database Tables

```
CREATE TABLE STUDENT(
        StudentID varchar(8) not null primary key,
        Name varchar(25) not null,
        Surname varchar(20) not null,
        Birthdate Date not null,
        Adress varchar(30),
        Phone varchar(15),
        EMail varchar(25),
        StartTerm char(4) not null,
        AdvisorID varchar(12),
        MajorID int(3) unsigned
);
CREATE TABLE INSTRUCTOR(
        InstructorID varchar(12) not null primary key,
        Name varchar(25) not null,
        Surname varchar(20) not null,
        Birthdate Date not null,
        DeptID int(2) unsigned not null,
        RoomID int(5) unsigned,
        RPhone varchar(4),
        Adress varchar(30),
        Phone varchar(15),
        EMail varchar(25)
);
CREATE TABLE STAFF(
        StaflID varchar(l2) not null primary key,
        Name varchar(25) not null,
```

```
        Surname varchar(20) not null,
        Birthdate Date not null,
        RoomID varchar(2),
        RPhone varchar( 4),
        Adress varchar(30),
        Phone varchar(l 5),
        EMail varchar(25)
);
CREATE TABLE COURSES(
        CourseID varchar(8) not null primary key,
        Title varchar(30),
        Credits int(l),
        PreReq varchar(8)
);
CREATE TABLE BUILDING(
        BuildingID varchar(4) NOT NULL PRIMARY KEY,
        BuildingName varchar(35)
);
CREATE TABLE LOCATION(
        RoomID int(5) unsigned not null auto_increment Primary Key,
        BuildingID varchar( 4) not null,
        RoomNo varchar(3) not null,
        Capacity int(4),
        RoomType char(l)
);
CREATE TABLE ROOM(
        RoomType char(l) not null primary key,
        RoomDesc varchar(IO) not null
);
CREATE TABLE DEPARTMENT(
        DeptID int(2) unsigned not null auto_increment primary key,
        DeptDesc varchar( 40),
        Dean int(5)
);
```

```
CREATE TABLE MAJOR(
        MajorID int(3) unsigned not null auto_increment primary key,
        MajorDesc varchar( 40)
);
CREATE TABLE TERM(
        TermID CHAR(4) not null primary key,
        TermDesc VARCHAR(ll),
        StartDate DATE,
        EndDate DATE
);
CREATE TABLE CRSSECTION(
        CsID int(6) unsigned not null auto_increment primary key,
        CourseID varchar(8),
        Section char(2) default "G1",
        TermID char(4),
        InstructorID varchar(12),
        MaxST int(3)
);
CREATE TABLE REGISTRATION(
        StudentID varchar(8),
        CsID int( 6),
        Result varchar(2),
        Primary Key(StudentID, CsID)
);
CREATE TABLE DOCUMENTS(
        DocID int(IO) primary key auto_increment,
        uploader varchar(12) not null,
        toWho varchar( 12) not null,
        fileName varchar(65),
        fileSize int(15),
        doc blob(lOOOOOOO),
        theDate Date,
        contentType varchar(15),
        st boolean,
```

```
        ins boolean,
        stfboolean
);
CREATE TABLE SECURITY(
        userID varchar(12) primary key,
        pswrd char(32),
        status int(1)
);
CREATE TABLE ANNOUNCES(
        announcesID int(9) primary key auto_increment,
        status boolean default true,
        sendfrom varchar(12),
        senderStatus int(1),
        sendto varchar(12),
        theDate date,
        content blob
);
CREATE TABLE SCHEDULE(
        CsID varchar(8),
        theDate date,
        RoomID int(5),
        ScDesc varchar(20)
);
```