

Student Java Online Documentation

(Advanced Java project – J2EE)

Sayantani Banerjee (WBGHJI5901)

Arnab Mukherjee (WBGHBW7350)

Rahul Dass (WBGHJI7236)

19th July, 2013

Postgraduate Diploma in Software Engineering (PGDSE)

Project Impact, Jadavpur University, Kolkata – 700032, India.

In collaboration with

Brainware-India IT group, Gariahat Training Centre, Kolkata – 700019, India.

Abstract

This report is an attempt to delineate and discuss two major concepts, Java Server Pages and Java Database Connectivity (accomplished through MS Access), implemented by the powerful and dynamic technology called Java 2 Enterprise Edition. A thorough quantitative investigation of this technology is conducted via a novel project that coincidentally, focuses on a student's requirement for a concise representation of the knowledge needed to understand the heart of Java namely, is the Java 2 Standard Edition.

Acknowledgement

Success is the manifestation of diligence, perseverance, inspiration, motivation and innovation. A successful project is a fruitful culmination of efforts of many people, some directly involves while other that have quietly encouraged and extended their support while being in the background.

We are grateful to **Ms. Deepa Ashok Kumar** (J2EE lecturer and project supervisor) not only for her invaluable guidance but also for her unending encouragement that she has given to me throughout the J2EE course and project. This work is a reflection of her teachings and expertise in the subject that she has passed on to us. This course and project has been challenging and most enthralling experience. We sincerely thank her for her patience, humour and knowledge, and hope to continue forward to better and more exciting days with our careers.

As I, Rahul Dass, am submitting this documentation first on behalf of my team. I declare that the above acknowledgement and this documentation along with this project: 'Java online documentation' in its entirety is truthful, honest and in complete reflection of my team members, Sayantani Banerjee (WBGHJI5901) and Arnab Mukherjee (WBGHBW7350).

Signature of student

Table of contents

Chapter 1: Introduction.

Chapter 2: Design view of project tables.

Chapter 3: Snapshots and Coding.

3.1 Login page.

3.2 New user page.

3.3 User homepage.

3.3.1 Navigation of topics.

3.3.2 MCQ.

3.4 Admin homepage.

3.4.1 Admin privilege: Insert

3.4.2 Admin privilege: Update

Chapter 4: Future scope.

4.1 Login + New user.

4.2 Search bar.

4.3 Feedback form to the admin.

4.4 MCQ.

4.5 Extend admin privileges for the entire project.

Chapter 5: Conclusion.

Chapter 6: References.

Chapter 1: Introduction.

The motivation that propelled this project: ‘Java online documentation’, was predominately based on exploring the possibilities of a powerful member of the Java platform, namely, the Java 2 Enterprise Edition (J2EE). The J2EE application and architecture is an established piece of software that is renowned for being robust in its core foundation: supporting multi-tier web applications and many types of clients that can run on various computing devices. With a vision of inheriting the plethora of advantages that comes with the core of the Java platform: Java 2 Standard Edition (J2SE), which is solely devoted to building desktop applications and experiences, J2EE is a unique technology that targets the needs of a web-oriented Java experience within the framework of an enterprise model.

This project was primarily focused in implementing two aspects of the J2EE technology: Java Server Pages (JSP) and Java Database Connectivity (JDBC). The purpose of using JSP is to handle and create dynamic web content. What makes this component of the technology ideal for handling the client side requirements of this project is by enabling the java code to be written with html code, thus being embedded together, thus eliminating the need for creating two separate files that are dedicated for java and html code respectively: resulting to a high performance in the execution of the web application from a client-side perspective. One of the key components of the JSP Application Programming Interface (API) is the fact that it has access to another API within the J2EE technology called JDBC. This is the nature by which the client-side communicates with the server-side aspects of the web application. In this project, we are using the software ‘Microsoft Access’ to handle the database requirements.

Now that we have given a brief overview of the technology and its respective components that we are implementing, the focus should now be what the ‘Java online documentation’ is all about?

In this project, the motivation is to create a database of a small selection of topics concerning the core J2SE technology and provide a user with access to concise and succinct information with regard to those topics. The user, ideally, being a student would first have to provide a username and password combination, register himself/herself in order to have any access to the database. Once the student has logged in, there is a default set of 4 topics, in the form of hyperlinks, which would take the student to the respective information concerning the topics in the form of questions and answers that we as students would ask when first presented with such topics. From an admin point of view, the three of us as founders of this project, have additionally given ourselves the privilege to insert and update topic names from the user/admin homepages. The nature of such data manipulation has stemmed from the fact that through the web application itself, we are able to dynamically insert new and modify old data into the MS-Access database, without having to actually go to MS-Access itself and perform any changes in data in the window application.

Finally, we conclude this report by stating the future scope that we would have loved to implement, however due to the complex nature of the code, ambition intentions of the project and severe time constraints, we were unable to explore more exotic possibilities.

However, considering the project as a whole: the ‘Java online documentation’, we believe that we have self-explored and taken the knowledge gained during lectures to the next level. By constantly challenging ourselves to better design and implement our code, while especially during the ‘debugging’ phase of the project, having to force ourselves to think for solutions to errors: helped us gain tremendous self-confidence and invaluable experience when dealing with a large-scale project for the first time in our careers.

Chapter 2: Design view of project tables.

In this chapter, we present the design structure: field names and their respective data types, for the four major tables that were used in this project.

a) Login table

Field name	Data type
First Name	Text
Last Name	Text
id	Text
password	Text
re-pass	Text
user_type	Text

b) Topic table

Field name	Data type
topic_id	Number
topic_name	Text

c) Details table

Field name	Data type
qs_no	Number
topic_id	Number
Question	Memo
Info	Memo

d) MCQ table

Field name	Data type
Topic_id	Number
Ques_Num	AutoNumber
Question	Text

Option A	Text
Option B	Text
Answer	Text

DFD



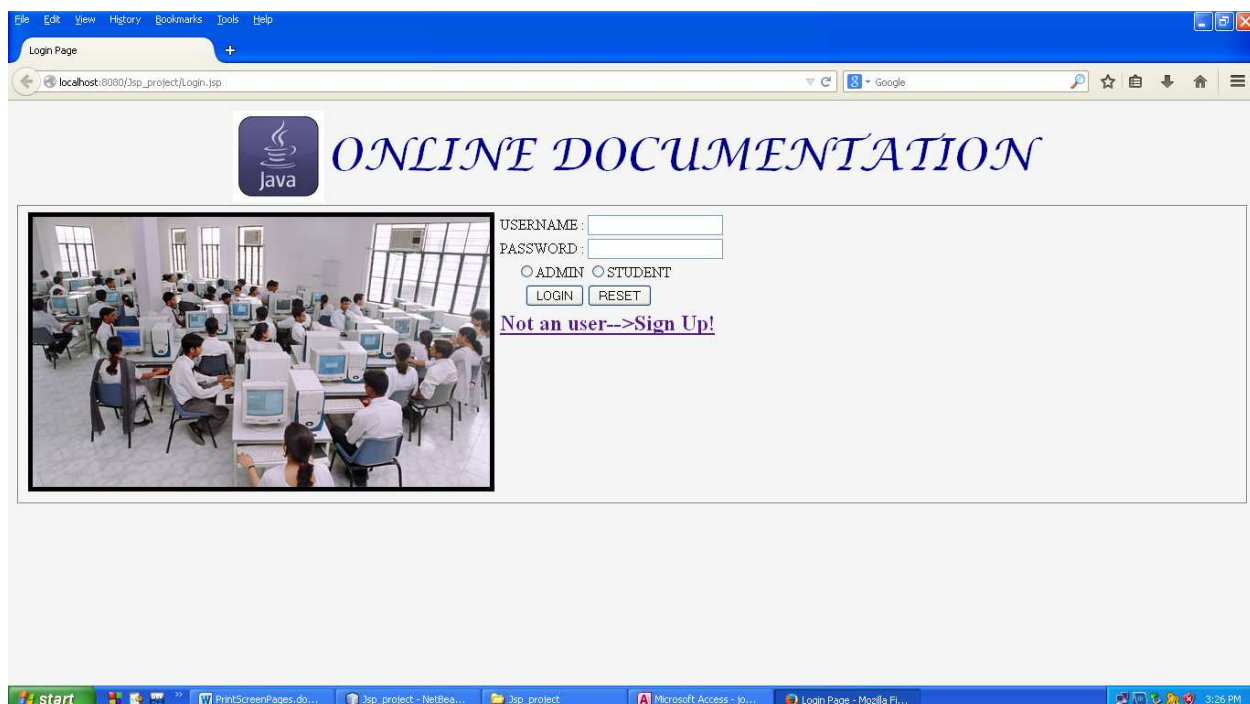
Chapter 3: Snapshots and Coding.

In this chapter, we present the logic and procedure for the every section of the project. Starting with an explanation of how the user would attempt to navigate through the web application, followed by a snapshot of the various scenarios that the user may or may not undergo, finally ending with the actual JSP code that was written

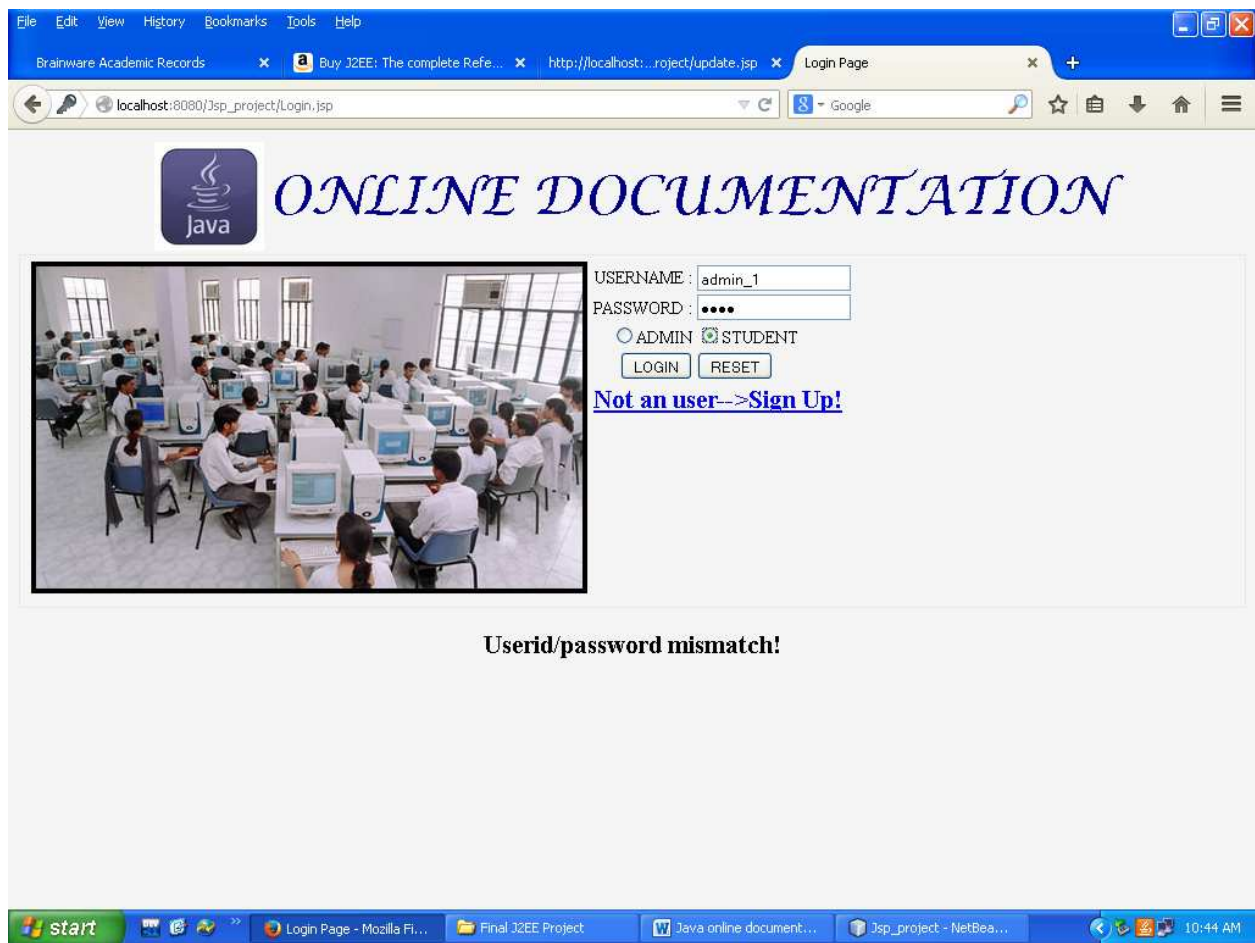
3.1 Login page.

In this section, the user will attempt to access the ‘Java online documentation’ database by inputting a username/password combination and depending on the designated privilege, the user would have to select either the ‘Admin’ or ‘Student’ status. By default, every ‘new’ user that would be created would be classified as a ‘student’ as explained further in section 3.2.

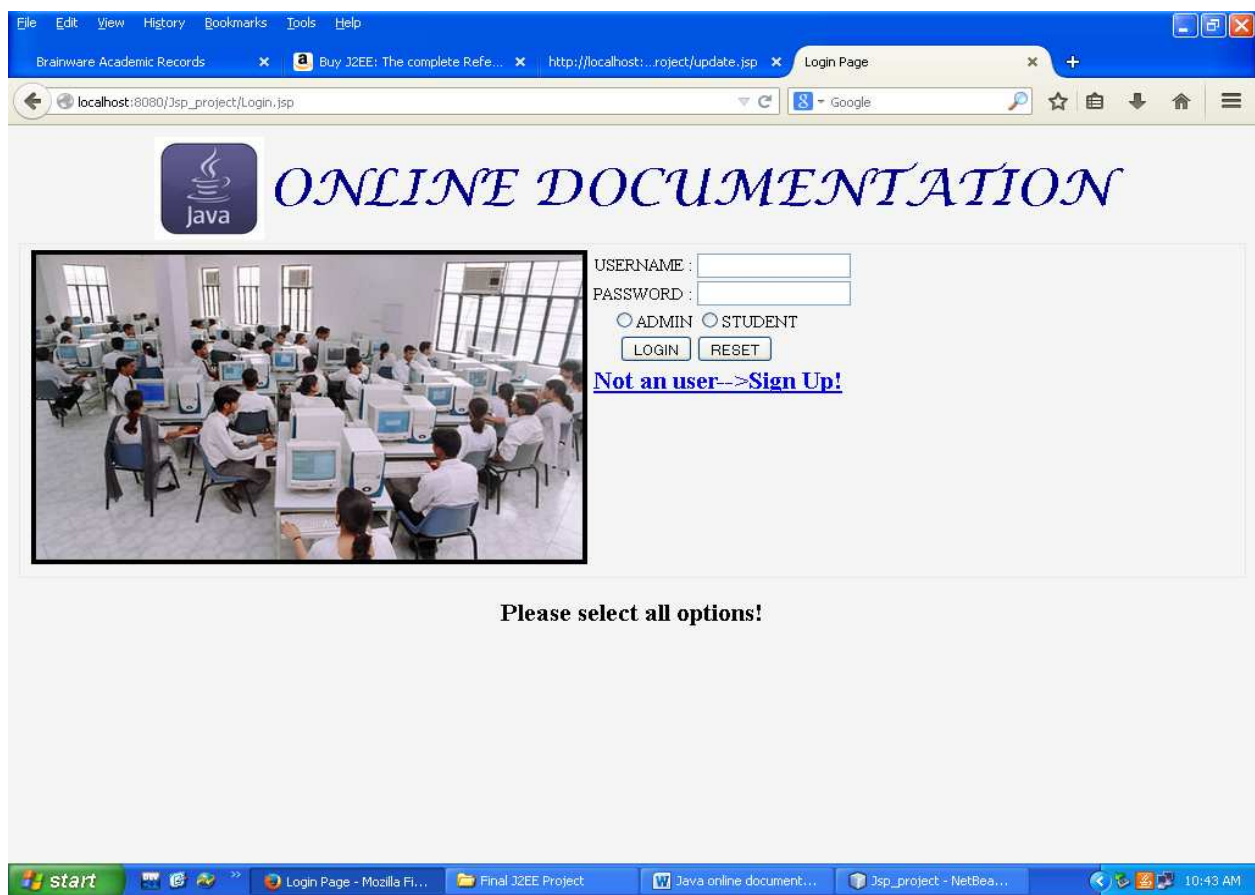
Login page snapshot:



If the user gives an invalid username/password combination, the appropriate error-handling message will be posted for the user:



If the user does not select any options, the appropriate error-handling message will be posted for the user:



Login page JSP code:

```
<%@page import="java.sql.*" %>

<html>

    <head>

        <title>Login Page</title>

    </head>

    <body bgcolor="WhiteSmoke">

        <form method="post" action="Login.jsp">

            <table align="center">

                <tr><td align="right">

                    </td><td><font face="Lucida Calligraphy" size="18"
color="#00008B"> ONLINE DOCUMENTATION</font></td></tr>

            </table>

            <fieldset>

                <table>

                    <tr><td align="right"><label>USERNAME :</label></td><td><input type="text"
name="username"/></td></tr>

                    <tr><td align="right"><label>PASSWORD :</label></td><td><input type="password"
name="password"/></td></tr>

                    <tr><td align="right"><input type="radio" name="usertype" value="a"/>ADMIN</td><td><input
type="radio" name="usertype" value="s"/>STUDENT</td></tr>

                    <tr><td align="right"><input type="submit" name="submit" value="LOGIN"/> </td><td><input
type="reset" name="reset" value="RESET"/></td></tr>

                    <tr><td colspan="2"><h2><a href="newuser.jsp">Not an user-->Sign Up!</h2></a></td></tr>

                </table>

            </fieldset>

        </form>

    <%!

        Connection con;

        Statement st;

        ResultSet rs;
```



```

String id,pass,utype,invID;

%>
<%
try
{
    id=request.getParameter("username");
    pass=request.getParameter("password");
    utype=request.getParameter("usertype");

    if(request.getParameter("submit").equals("LOGIN"))
    {
        if(id!="" && pass!="" && utype!="")
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

            String myDB = "jdbc:odbc:Driver={Microsoft Access Driver
(*.mdb)};DBQ=E:/Jsp_project/jod.MDB";

            Connection con = DriverManager.getConnection(myDB);

            st=con.createStatement();

            rs=st.executeQuery("select * from Login where id='"+id+"' and password='"+pass+"'");

            if(rs.next())
            {
                invID="";

                if(utype.equals(rs.getString(6)))
                {

                    session.setAttribute("username",id);
                    session.setAttribute("password",pass);

                    if(utype.equals("a"))
                    {
                        %>

```

```

        <jsp:forward page="admin_home.jsp"/>
    <%
    }
    else if(utype.equals("s"))
    {
    %>
        <jsp:forward page="user_home.jsp"/>
    <%
    }
    }
    else
    {
        invID="Invalid login";
        out.println(invID);
    }
}
else
    out.println("Userid/password mismatch");
}
else
    out.println("Please select all options");
}
} catch(Exception e){ }
%>
</table></body></html>

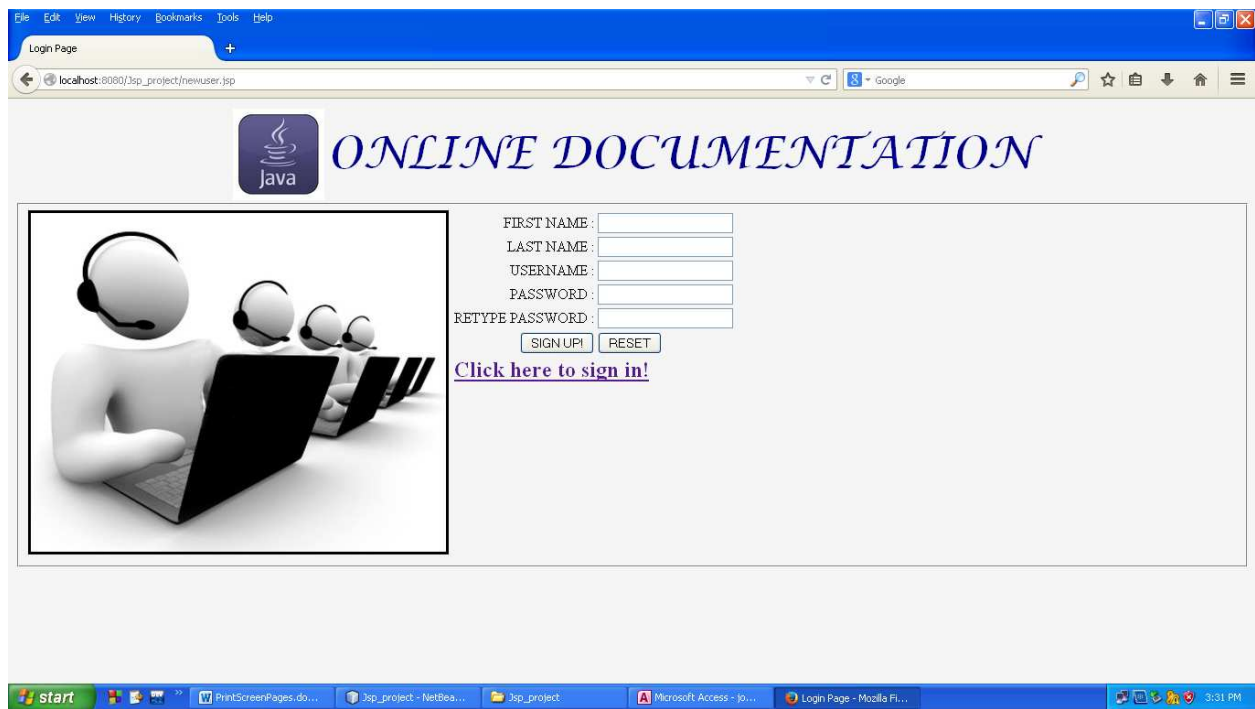
```

3.2 New user page.

In this section, a new user would first have to register himself/herself in order to access the database, by clicking on the ‘Not a user→Sign up!’ hyperlink. The user will be redirected to a new page where there is a ‘registration form’, to fill in the user’s details. The data that the user enters will be saved in the ‘Login table’, whose structure was defined in Chapter 2. After the user has completed inputting the data, by clicking on the ‘Click here to sign in!’ hyperlink, the newly created user will be redirected again to a fresh login page to sign in.

It should be noted that, other than the three authors of this project, all registered users would be given the privilege of a user. By implementing the concept of sessions, once the user has logged in, they would have a restricted access to the database, as compared to an administrator: representing each of the three authors of this project.

New user page snapshot:



apshot:

NewUser.jsp code:

```
<%@page import="java.sql.*" %>

<html>

    <head>

        <title>New User page</title>

    </head>

<body bgcolor="WhiteSmoke">

<form method="post" action="newuser.jsp">

<table align="center">

    <tr><td align="right">

        </td><td><font face="Lucida Calligraphy" size="18"
color="#00008B"> ONLINE DOCUMENTATION</font></td></tr>

    </table>

<fieldset>

<table>
```

```

        <tr><td align="right"><label>FIRST NAME :</label></td><td><input type="text"
name="fname"/></td></tr>

        <tr><td align="right"><label>LAST NAME :</label></td><td><input type="text"
name="lname"/></td></tr>

        <tr><td align="right"><label>USERNAME :</label></td><td><input type="text"
name="username"/></td></tr>

        <tr><td align="right"><label>PASSWORD :</label> </td><td><input type="password"
name="password"/></td></tr>

        <tr><td align="right"><label>RETYPE PASSWORD :</label></td><td> <input type="password"
name="rp"/></td></tr>

        <tr><td align="right"><input type="submit" name="submit" value="SIGN UP!"/> </td><td><input
type="reset" name="reset" value="RESET"/></td></tr>

        <tr><td colspan="2"><h2><a href="Login.jsp">Click here to sign in!</a></h2></td></tr>

</table>

</fieldset>

</form>

```

```

<%!

```

```

    Connection conn;

```

```

    Statement st1,st2;

```

```

    ResultSet rt;

```

```

    String fn,ln,us,ps,rp;

```

```

    int r;

```

```

    %>

```

```

<%

```

```

try

```

```

{

```

```

    fn=request.getParameter("fname");

```

```

    ln=request.getParameter("lname");

```

```

    us=request.getParameter("username");

```

```

    ps=request.getParameter("password");

```

```

    rp=request.getParameter("rp");

```

```

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

String myDB = "jdbc:odbc:Driver={Microsoft Access Driver
(*.mdb)};DBQ=E:/Jsp_project/jod.MDB";

conn = DriverManager.getConnection(myDB, "", "");

st1=conn.createStatement();

st2=conn.createStatement();

rt=st1.executeQuery("select * from Login where id='"+us+"'");


if(!rt.next())

{

    if(ps.equals(rp))

    {

        r = st2.executeUpdate("insert into Login values('"+fn+"','"+ln+"','"+us+"','"+
            +ps+"','"+rp+"','s')");

    }

    else

    {

        out.println("Passwords do not match!!!");

    }


    if(r!=0)

    {

        %>

        <h4 align="center">New user created!</h4>

        <%

    }

}

} catch(Exception e){}

%>

```

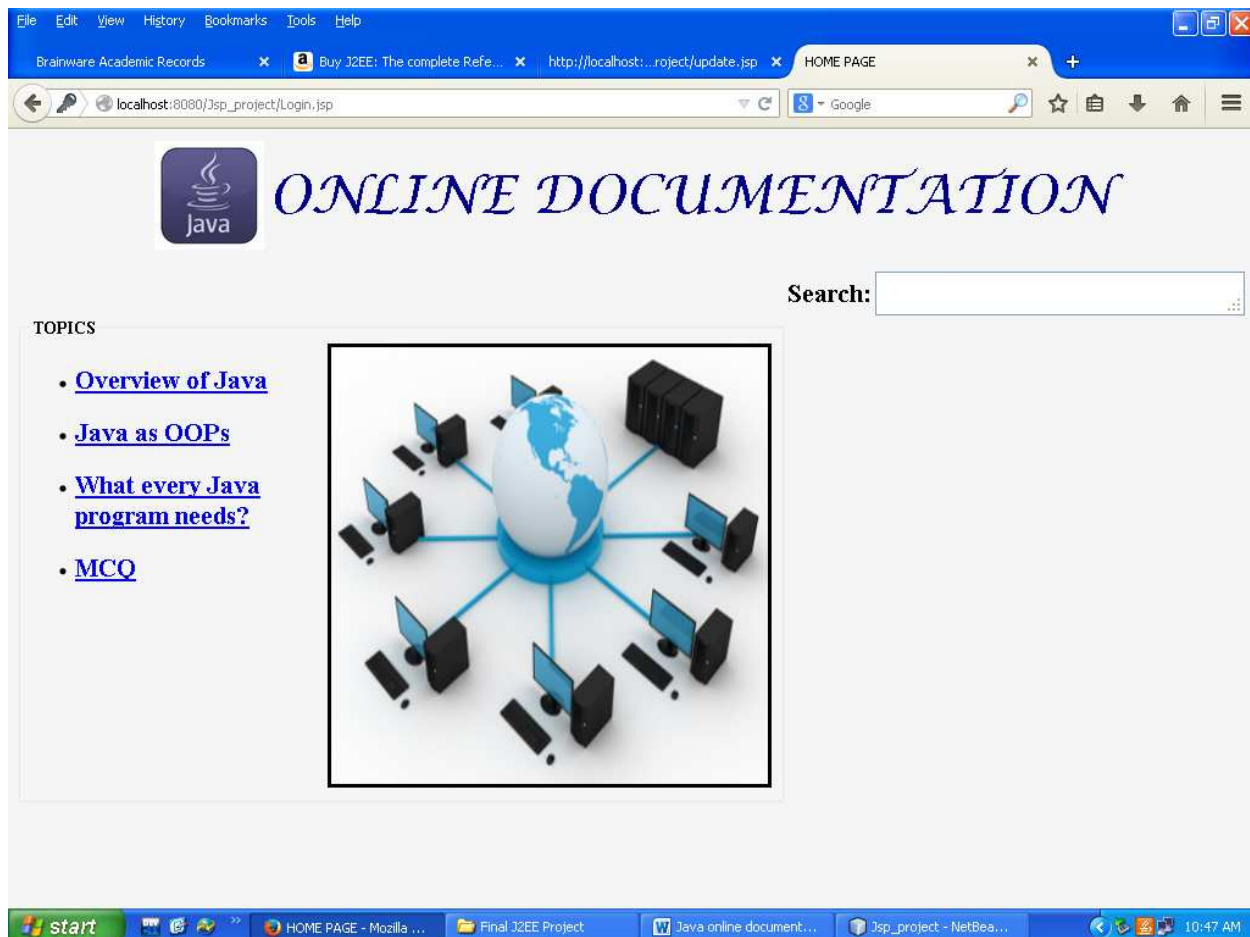
</body></html>

3.3 User homepage.

In this section, once the user has logged in, he/she will be able to see the list of topics that have been chosen from the J2SE technology, in the form of hyperlinks. Clicking each hyperlink would redirect the user forward into the database, as will be discussed in section 3.4.

Due to time constraints, the search bar at the top-right hand corner of the screen was the preliminary implementations of some of the future scopes for this project as will be discussed further in section 4.2.

User homepage snapshot:



User homepage code:

```
<%@page import="java.sql.*" %>

<html>

    <head>

        <title> User Home Page</title>

    </head>

    <body bgcolor="WhiteSmoke">

        <form method="post" action="1.jsp">

            <table align="center">
```

```

        <tr><td align="right">

            </td><td><font face="Lucida Calligraphy" size="18"
color="#00008B"> ONLINE DOCUMENTATION</font></td></tr>

    </table>

<table align="right">

<tr><td><label><h2>Search:</h2></label></td><td><textarea rows="1" cols="40"></textarea></td></tr></td>

</table><br/><br/><br/>

<fieldset>

    <legend><b>TOPICS</b></legend>

    <ul>

<%!

    Connection con;

    Statement st;

    ResultSet rs;

    int tid;

    String u,p,student,admin,tnm;

%>

<%

    try

    {

        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

        String myDB = "jdbc:odbc:Driver={Microsoft Access Driver (*.mdb)};DBQ=E:/jsp_project/jod.MDB";

        Connection con = DriverManager.getConnection(myDB);

        st=con.createStatement();

        rs=st.executeQuery("select * from topic");

        while(rs.next())

        {

            tid=rs.getInt(1);

            tnm=rs.getString(2);

            //session.setAttribute("tid", new Integer(tid));

```

```

%>

<li><h2><a href="1.jsp">

    <%=tnm%></a></h2></li>

<%

}

}catch(Exception e){}

%>

</ul>

</fieldset>

</body>

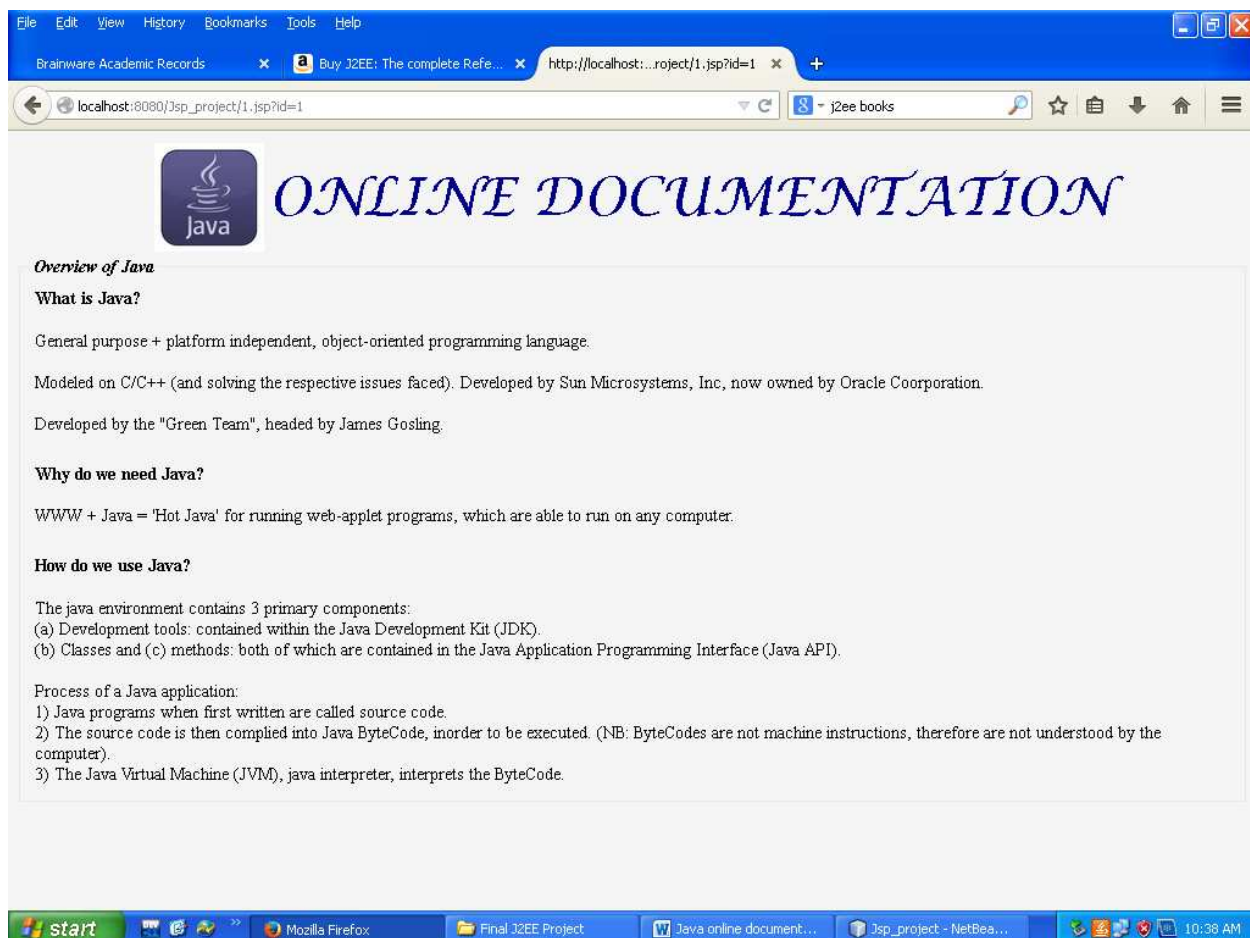
</html>

```

3.3.1 Navigation of topics.

After the user clicks on the various hyperlinks representing the J2SE topics, the user will be presented with information in the form of “immediate questions and answers”, that we administrators as students have asked when first understanding the J2SE technology.

‘Overview of topics’ snapshot:



‘Java as OOPs’ snapshot:

The screenshot shows a web browser window with the address bar displaying 'localhost:8080/jsp_project/1.jsp?id=2'. The page title is 'ONLINE DOCUMENTATION' with a Java logo. The content is titled 'Java as OOPs' and 'What is OOPs?'. It defines an object as a bundle of related variables and methods, and a class as a prototype that defines variables and methods common to all objects. It also states that objects are referred to as 'instances of a class'. The 'Main features of OOPs' are listed as follows:

- 1) Encapsulation: objects are self-contained in terms of data elements and the actions upon which will act on the data elements. It ensures objects are distinct and allows them to communicate with each other through programs.
- 2) Inheritance: Each subclass inherits variable declarations (states) from its superclass. But, subclasses can have variables and methods of their own (i.e. can be created!).
- 3) Abstraction: Focuses on the essential details of an object. Note: An abstract class in Java cannot have an instance.
- 4) Polymorphism: Behaviour that varies depending on the class in which behaviour is invoked (i.e. 2 or more classes react differently to the same message).

The taskbar at the bottom shows the Start button, Mozilla Firefox, and several open applications including 'Final J2EE Project', 'Java online document...', and 'Jsp_project - NetBea...'. The system clock shows 10:39 AM.

‘What every Java program needs?’ snapshot:

The screenshot shows a web browser window with the address bar displaying 'localhost:8080/jsp_project/1.jsp?id=3'. The page title is 'ONLINE DOCUMENTATION' with a Java logo. The content is titled 'What every Java program needs?'. It explains the 'public static void main(String args[])' method. 'public' means the content can be accessed from any classes within the same package or from different packages. 'Static' means that no object creation or reference creation of the class is required. 'void' means no return type is required. 'main' is the program execution start from the main function. 'String(args[])' is for passing command line arguments.

Simple Program To Print Something:

```
Class A
{
    public static void main(String args[])
    {
        System.out.println("Hello Students");
    }
}
```

Output: Hello Students.

Program to add two numbers:

```
Class B
{
    public static void main(String args[])
    {
        int a=5, b=10, c;
```

The taskbar at the bottom shows the Start button, Mozilla Firefox, and several open applications including 'Final J2EE Project', 'Java online document...', and 'Jsp_project - NetBea...'. The system clock shows 10:39 AM.

The screenshot shows a web browser window with the address bar at `localhost:8080/jsp_project/1.jsp?id=3`. The page content includes two Java code snippets. The first snippet, labeled 'Class B', defines a `main` method that calculates the sum of `a=5` and `b=10`, resulting in `c=15`. The output shown is `a=5 b=10` and `The sum is = 15`. The second snippet, labeled 'Class test', defines a `main` method that checks if a year is a leap year. It uses the logic `if(year%400==0 || year%4==0 && year%100!=0)`. The output shows `2000 is a leap year.` and `1700 is not a leap year.`. The browser's taskbar at the bottom shows several open applications, including Mozilla Firefox, Final J2EE Project, Java online document..., and Jsp_project - NetBea....

```
Class B
{
    public static void main(String args[])
    {
        int a=5, b=10, c;
        System.out.println("a="+a+" b="+b);
        c=a+b;
        System.out.println("The Sum is =" +c);
    }
}

Output: a=5 b=10
        The sum is = 15

Program to check whether a year is leap year or not:

Class test
{
    public static void main(String args[])
    {
        int year;

        year=Integer.parseInt(args[0]);

        if(year%400==0 || year%4==0 && year%100!=0)
            System.out.println(year+"is a leap year.");
        else
            System.out.println(year+"is not a leap year.");
    }
}

Output: 2000 is a leap year.
        1700 is not a leap year.
```

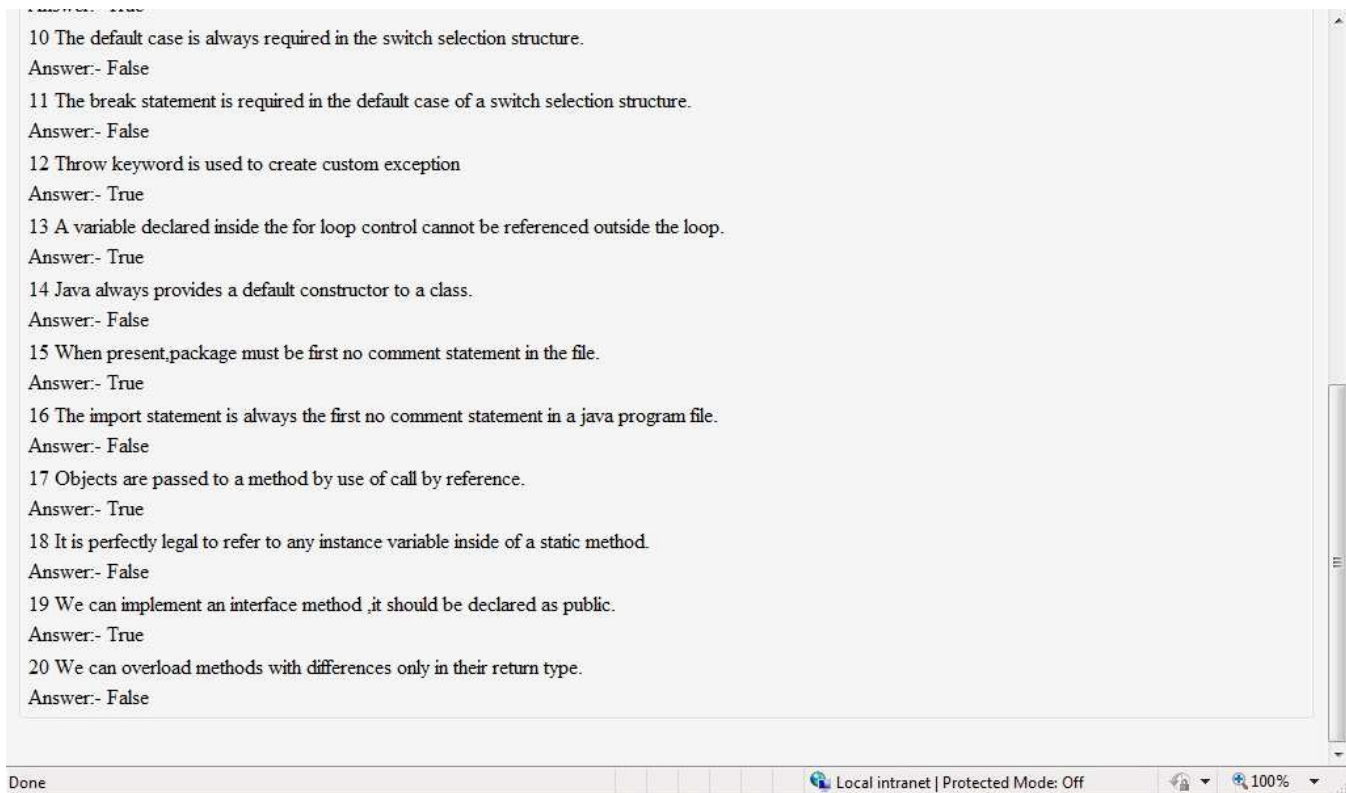
‘MCQ’ snapshot:

The screenshot displays a web page titled 'ONLINE DOCUMENTATION' with a Java logo. Below the title, there is a section for 'MCQ Exam' containing nine multiple-choice questions about Java. Each question is followed by its correct answer. The questions cover topics such as file naming conventions, method naming rules, operator usage, declaration placement, operator precedence, bitwise operations, conditional statements, logical expressions, and interface methods. The page is viewed in a browser window with a status bar at the bottom indicating 'Local intranet | Protected Mode: Off' and a zoom level of 100%.

ONLINE DOCUMENTATION

MCQ Exam

- 1 The name of a java program file must match the name of the class with the extension java
Answer:- True
- 2 Two methods cannot have the same name in java.
Answer:- False
- 3 The modulus operator(%) can be used only with integer operands.
Answer:- False
- 4 Declarations can appear anywhere in the body of a java method.
Answer:- True
- 5 All the bitwise operators have the same level of precedence in java.
Answer:- True
- 6 When x is a positive number the operation `x>>2` & `x>>>2` both produce the same result.
Answer:- True
- 7 If `a=10` & `b=15` then the statement `x=(a>b)?a:b`; assigns the value 15 to x.
Answer:- True
- 8 In evaluating a logical expression of type "Boolean expression 1 && Boolean expression 2", both the boolean expressions are not always evaluated
Answer:- True
- 9 In an interface all the methods declared are abstract
Answer:- True



Navigation of topics code:

```
<%@page import="java.sql.*" %>
```

```
<html>
```

```
<head>
```

```
<title></title>
```

```
<link href="style1.css" rel="stylesheet" type="text/css"/>
```

```
</head>
```

```
<body bgcolor="WhiteSmoke">
```

```
<form method="post" action="1.jsp">
```

```
<table align="center">
```

```
<tr><td align="right">
```

```
</td><td><font face="Lucida Calligraphy" size="18"
color="#00008B"> ONLINE DOCUMENTATION</font></td></tr>
```

```
</table>
```

```
<fieldset>
```

```
<%!
```

```
Connection con;
```

```
Statement st,st1;
```

```

ResultSet rs2,rs1,rs3,rs4,rs5;

String id,Question,Info,question1,opt[],a,nm;

int tid,quesnum,count=0,k=0,m=0,i;

%>

<%

try
{

    tid=Integer.parseInt(session.getAttribute("tid").toString());

    out.println("Answer is= "+tid);

    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

    String myDB = "jdbc:odbc:Driver={Microsoft Access Driver
(*.mdb)};DBQ=E:/Jsp_project/jod.MDB";

    Connection con = DriverManager.getConnection(myDB);

    st=con.createStatement();

    st1=con.createStatement();

    rs1=st.executeQuery("select * from topic where topic_id="+tid);

    if(rs1.next())

    {

        %>

        <legend><b><em><%=rs1.getString(2)%></em></b></legend>

        <%

    }

    if(tid!=6)

    {

        rs2=st.executeQuery("select * from details where topic_id="+tid);

        while(rs2.next())

        {

            Question=rs2.getString(3);

            Info=rs2.getString(4);

            %>

```

```

<table>

    <tr><td>

<strong><%=Question%></strong></td></tr>

<tr><td><p id="ans">

<font><%=Info%></font>

        </p></td></tr>

</table>

<%

}

}

else

{

//session.setAttribute("tid", new Integer(tid));

//tid=Integer.parseInt(session.getAttribute("tid").toString());

rs3=st.executeQuery("select * from MCQ where topic_id="+tid);

rs3.next();

quesnum=rs3.getInt(2);

question1=rs3.getString(3);

%>

    <table>

        <tr>

            <%=quesnum%>

            <%=question1%></tr>

        <tr><td>

<input type="radio" name="an" value="True">  True

<input type="radio" name="an" value="False">  False

        </p></td></tr>

    </table>

    <input type="submit" name="ak" value="Next">

<%

```

```

        if(request.getParameter("ak").equals("Next"))
        {
            //tid=Integer.parseInt(session.getAttribute("tid").toString());

            rs3=st.executeQuery("select * from MCQ where topic_id="+tid);

            rs3.next();

            }

        }

    } catch(Exception e){}

%>

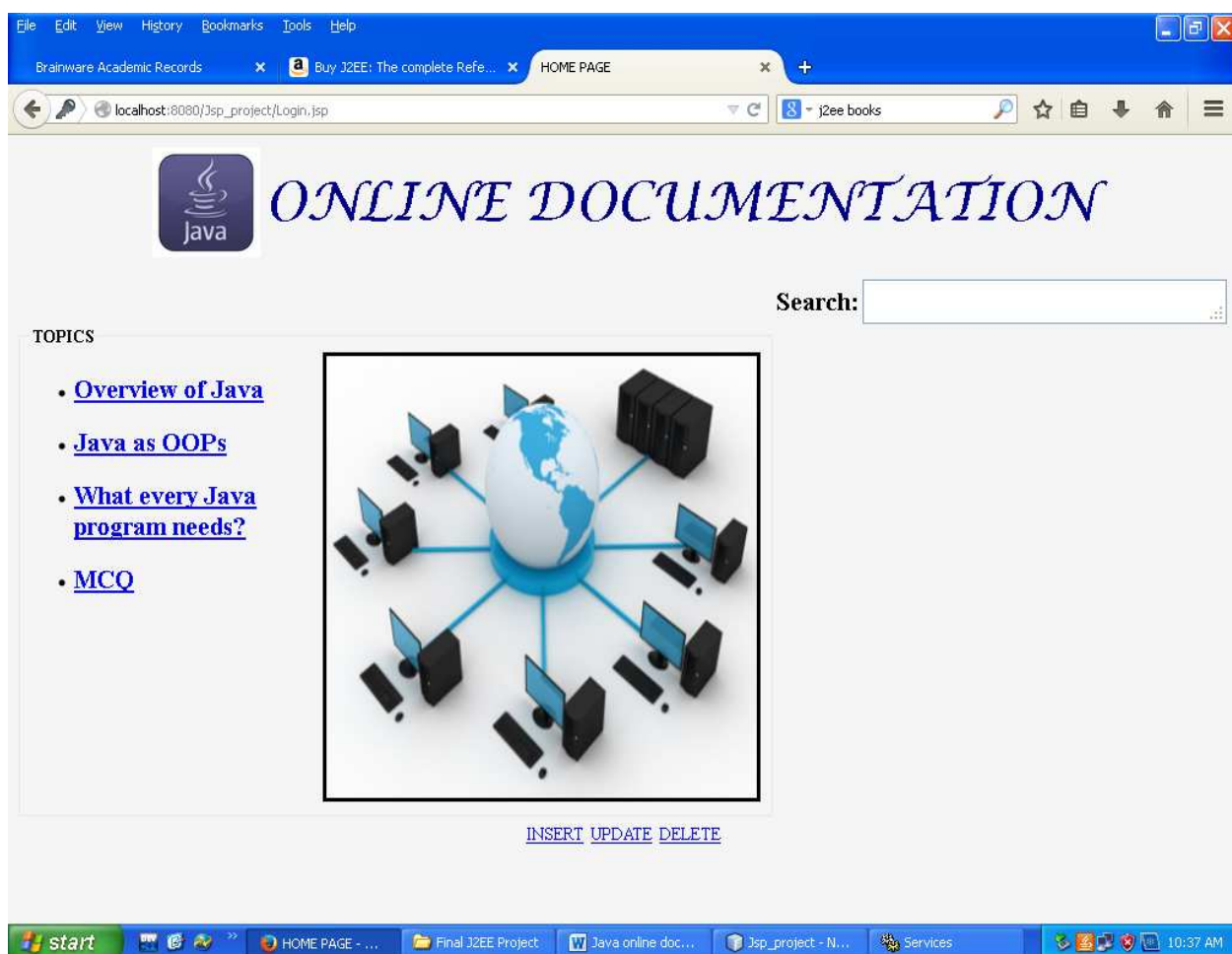
</fieldset></form></body></html>

```

3.4 Admin homepage.

As an administrator, when logging into the system, we attain access to the same set of information that a student would, but in addition, we would also have the privilege to insert, update and delete information from the respective tables that hold the data. Such opportunities are presented in the form of hyperlinks, at the bottom of the snapshot below.

Admin homepage snapshot:



Admin homepage JSP code:

```
<%@page import="java.sql.*" %>

<html>

    <head>

        <title>HOME PAGE</title>

    </head>

    <body bgcolor="WhiteSmoke">

        <form method="post" action="">

            <table align="center">

                <tr><td align="right">

                    </td><td><font face="Lucida Calligraphy" size="18"
color="#00008B"> ONLINE DOCUMENTATION</font></td></tr>

                </table>

                <table align="right">

                    <tr><td><label><h2>Search:</h2></label></td><td><textarea rows="1"
cols="40"></textarea></td></tr></table>

                    <br/><br/><br/>

                    <fieldset>

                        <legend><b>TOPICS</b></legend>

                        <ul>

                            <%!

                                Connection con;

                                Statement st;

                                ResultSet rs;

                                int tid;

                                String tnm;

                                %>

                                <%

                                    try

                                    {
```

```

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

String myDB = "jdbc:odbc:Driver={Microsoft Access Driver
(*.mdb)};DBQ=E:/Jsp_project/jod.MDB";

Connection con = DriverManager.getConnection(myDB);

st=con.createStatement();

rs=st.executeQuery("select * from topic");

while(rs.next())

{

    tid=rs.getInt(1);

    tnm=rs.getString(2);

    %>

    <li><h2><a href="1.jsp?id=<%=tid%>">

        <%=tnm%></a></h2></li>

    <%

    }

%>

</ul>

</fieldset>

<table align="center" cellspacing="5">

    <tr>

        <td style="vertical-align: top"><form method="post" action="insert.jsp"><a
href="insert.jsp">INSERT</a></form></td>

        <td><form method="post" action="update.jsp"><a
href="update.jsp">UPDATE</a></form></td>

        <td><form method="post" action="delete.jsp"><a
href="delete.jsp">DELETE</a></form></td>

    </tr>

</table>

<%

/*

if(request.getParameter("update").equals("UPDATE"))

```



```

{
    response.sendRedirect("update.jsp");
}

else if(request.getParameter("delete").equals("DELETE"))

{
}*/

} catch(Exception e){}

%>

</body> </html>

```

3.4.1 Admin privilege: Insert

In order for an admin to insert new data in the user/admin homepages, by clicking on the ‘insert’ hyperlink, the admin will be redirected to a new page as shown in the snapshot below.

Insert.jsp snapshot:

Insert a new topic name :

Topic_ID	Topic_Name
1	Overview of Java
2	Java as OOPs
3	What every Java program needs?
4	MCQ

After successfully inserting a topic name:

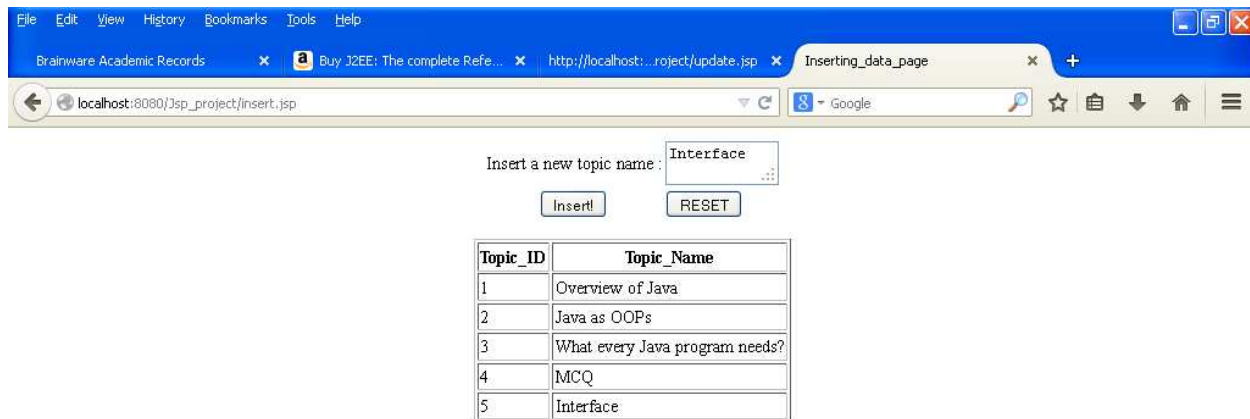


Topic_ID	Topic_Name
1	Overview of Java
2	Java as OOPs
3	What every Java program needs?
4	MCQ

Record inserted!



After refreshing the page, we can see the topic name inserted in the database (on the web):



Topic_ID	Topic_Name
1	Overview of Java
2	Java as OOPs
3	What every Java program needs?
4	MCQ
5	Interface



Upon entering an invalid entry in the text area:

Insert a new topic name :

Topic_ID	Topic_Name
1	Overview of Java
2	Java as OOPs
3	What every Java program needs?
4	MCQ

Please enter a topic name!



Insert.jsp code:

```
<%@page import="java.sql.*" %>

<html>

  <head><title>Inserting_data_page</title>

  </head>

  <body>

    <form method="post" action="insert.jsp">

<table align="center">

  <tr>

    <td align="center"><label>Insert a new topic name :</label></td>

    <td><textarea rows="1" cols="10" name="tn"></textarea></td>

  </tr>

  <tr>

    <td align="center"><input type="submit" name="insert" value="Insert!"/> </td>

    <td><input type="reset" name="reset" value="RESET"/></td>

  </tr>

</table>

</body>

</html>
```

```

        </tr>

</table>

</form>

<%!
    Connection con;

    Statement st, st1, st2, st3;

    ResultSet rs, rs1, rs2;

    String tpcNme, y, empty;

    int a, x, b;

%>

<table border="1" cellpadding="2" cellspacing="2" align="center">

    <tr>

        <th>Topic_ID</th><th>Topic_Name</th>

    </tr>

<%

    try

    {

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

        String myDB = "jdbc:odbc:Driver={Microsoft Access Driver
(*.mdb)};DBQ=E:/Jsp_project/jod.MDB";

        Connection con = DriverManager.getConnection(myDB);

        st=con.createStatement();

        st1=con.createStatement();

        st2=con.createStatement();

        st3=con.createStatement();

        rs=st.executeQuery("select * from topic");

        b=1;

        if(b==1 || b==2) {

            while(rs.next()) {

%>

```

```

<tr>

    <td><%=rs.getInt(1)%></td>

    <td><%=rs.getString(2)%></td>

</tr>

<%
    }
    }
%>

</table>

<%
    if(request.getParameter("insert").equals("Insert!"))
    {
        tpcNme = request.getParameter("tn");

        if(tpcNme.length()!=0) {
            empty="";

            rs2=st3.executeQuery("select * from topic where topic_name='"+tpcNme+"'");
            if(!rs2.next()) {
                //
                //y = rs2.getString(tpcNme);
            rs1=st2.executeQuery("select max(topic_id) from topic");
            if(rs1.next()) {
                x = rs1.getInt(1);
                x++;
            }
            else{
                x=1;
            }
            a = st1.executeUpdate("insert into topic values (" +x+", '"+tpcNme+"")");

```

```

    if(a!=0)
    {
        b=2;

        rs=st.executeQuery("select * from topic");

        rs.next();

        %>

        <h1 align="center">Record inserted!</h1>

        <%

        }

        }

        }

    else {

        empty="Please enter a topic name!";

        out.println(empty);

        }

    }

} catch(Exception e){}

    %>

</body>

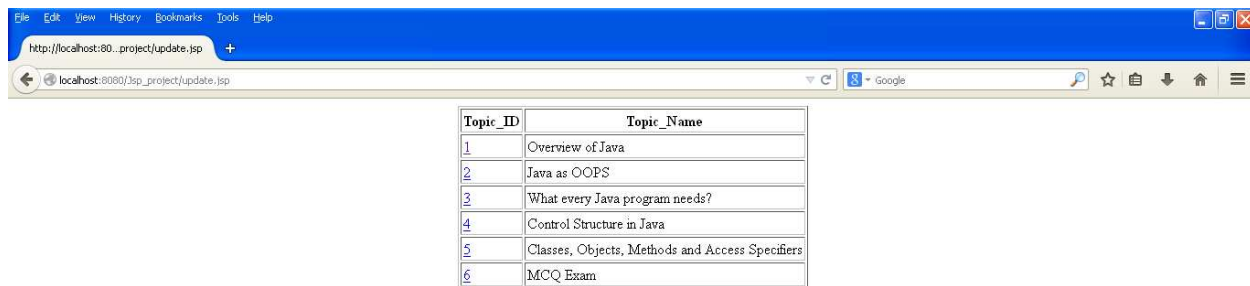
</html>

```

3.4.2 Admin privilege: Update

In this section, we attempted to provide the second privilege that an administrator would attain, in the form of updating the existing topics within the user/admin homepages, by clicking on the 'update' hyperlink.

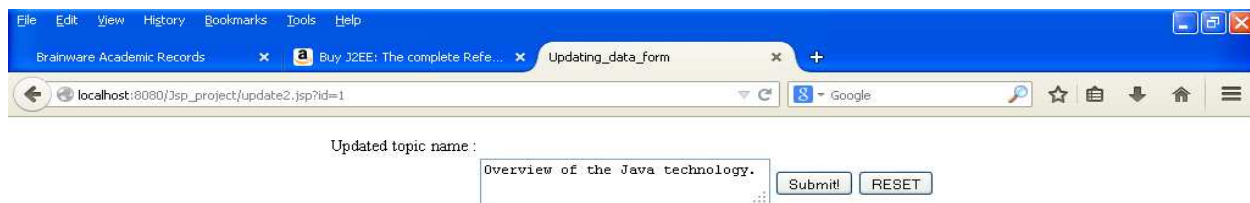
Update.jsp snapshot:



Topic_ID	Topic_Name
1	Overview of Java
2	Java as OOPS
3	What every Java program needs?
4	Control Structure in Java
5	Classes, Objects, Methods and Access Specifiers
6	MCQ Exam



After selecting the topic_ID hyperlink:



Updated topic name :

Overview of the Java technology.

Submit! RESET

Record updated!



Update.jsp code:

```
<%@page import="java.sql.*" %>

<%!

    Connection con;

    Statement st;

    ResultSet rs;

    String tpcNme;

    int tid;

%>

<table border="1" cellpadding="2" cellspacing="2" align="center">

    <tr>

        <th>Topic_ID</th><th>Topic_Name</th>

    </tr>

<%

    try

    {

        tpcNme = request.getParameter("tn");

        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

        String myDB = "jdbc:odbc:Driver={Microsoft Access Driver
(*.mdb)};DBQ=E:/Jsp_project/jod.MDB";

        Connection con = DriverManager.getConnection(myDB);

        st=con.createStatement();

        rs=st.executeQuery("select * from topic");

        while(rs.next()) {

            tid=rs.getInt(1);

%>

    <tr>
```



```

        <td><a href="update2.jsp?id=<%=tid%>"><%=tid%></a></td>

        <td><%=rs.getString(2)%></td>

    </tr>

<%

    }

%>

</table>

<%

    int a = st.executeUpdate("insert into topic (topic_name) values ('"+tpcNme+"')");

    if(a>0)
    {
        %>

        <h1>Record updated!</h1>

        <%

        }

    } catch(Exception e){}

    %>

</body>

</html>

```

Chapter 4: Future scope

In this section, we would like to elaborate what we wished to further include in the project, but was unable to do so due to time constraints.

4.1 Login + user.

By using a javascript pop-out form, we would have liked to have the ‘new user registration form’ appear through a pop-out window upon clicking the sign up hyperlink, instead of being redirected to a new window, thereby losing the attention of the new user.

After filling in the same details as mentioned in section 3.2, there would be three events that would take place at the time of clicking the submission button in the pop-out window.

- 1) The Login table would have to be updated with the contact information of the new user.
- 2) The existing log-in page should be refreshed, thus allowing for new username/password combinations to be accepted for access to the database.
- 3) The pop-out window should close, shifting the focus not to a new login page, but to the existing page that was already open and running in the background.

4.2 Search bar.

This represents the most ambitious part of the project during the design phase. The search bar would allow the user in the user/admin homepage to type a J2SE keyword and upon clicking a submit button, the resulting affect would be a redirection to a new page, with results from a complete scan of the entire database that holds various tables corresponding to different information. All results would be displayed in the form of hyperlinks, giving the user the complete freedom to choose how he/she would like to access the information depending on their particular needs.

Essentially, aesthetically it appears that the search bar has already been implemented in the project, however with a critical lack in functionality, as administrators, we did not anticipate the complexity in handling the code of such a feature. However, as a future scope for this project, we believe that the ‘search bar’ would definitely distinguish itself as a standout feature for this project.

4.3 Feedback form to the admin.

After logging into the database, the user would have a dedicated section in every page of this web application to communicate with the admins. This will be achieved by have a two-textbox, submit and reset button combination. The first textbox, illustrating the topic of the feedback, with regard to the page the user is currently on. The second textbox, slightly larger in nature would allow the user to convey his/her message to the admin. Upon submitting the feedback form, the admin would get a notification, when logging into the database the following time.

4.4 MCQ.

Upon looking at the snapshot of the MCQ section, we were intending for this to be the dynamic section for the student user. By testing his/her knowledge with MCQ style questions that would tally up a total mark the student would achieve, depending on the options selected.

The dynamic component comes from the fact that there would be a checking of the options selected at the time of submitting the MCQ form, with the MCQ table in the database. Furthermore, at the time of displaying the student’s result, we would have liked to show the questions and their respective answers, that the student attempted incorrectly. Thereby, establishing a deeper connection with the user and encourage him/her to either re-attempt the questions or dig further into understanding the J2SE technology.

4.5 Extend admin privileges for the entire project.

The only privileges that we were able to design and implement, were the insert and update sections for the user/admin homepages. An essential aspect regarding the future scope of the project would be, to not only include the code for deleting topics but also extending the concept of the privileges (i.e. inserting, updating, deleting data) that an admin would have in a project such as this to the entire database.

Chapter 5: Conclusion.

The aim of the project: 'Java online documentation', was to explore two of the most important concepts, JSP and JDBC (through MS-Access), in the powerful technology J2EE. We were able to create dynamic webpages, by implementing the ideal framework of JSP, which allows for html code and java code to be embedded and compliment one another.

While the client side of the project was handled by JSP, the server-side connection that the JDBC API was responsible for was achieved by using the MS-Access software. Creation of an entire database, split into logical and appropriate tables that correspond to different sections of project, made handling the enormous amount of data simple and easy to follow, when returning to the project.

The project was essentially broken down in to 3 parts. First, by creating an authentication page, where the user would have to log into the system in order to access the project's database, with the option for the creation of a new user available as well. Secondly, upon successfully logging in, the student now has access to select topics chosen from the J2SE technology. Depending on what the user wishes to accomplish, either gain succinct knowledge in the form of immediate questions and answers corresponding to the topics listed or test oneself with the MCQ section. Thirdly, as admins, we grant further privileges to ourselves by allowing the manipulation of data, specifically to the user/admin homepages (due to time constraints), through inserting and updating the database.

Finally, due to the ambitious nature of this novel project and severe time constraints, we have outlined our vision in Chapter 4: what we hope would have made this a solid and distinguished project, fully encapsulating the power and exploiting the capabilities within the J2EE framework, in order to make it a rich and personal experience for the user when seeking to gain an insight into J2SE.

As founders of this project, we take great pride in the way we have implemented classroom techniques in learning and understanding how to design, implement, debug and execute an advanced Java project. The unique and ambitious nature for our first major project has been an incredibly rewarding experience, but most importantly, it has given us a tremendous boost in confidence in coding and forced us to constantly think of ways to improve our design and code. We hope to continue this momentum for more exciting projects to be conducted in the near future.

Chapter 6: References.

- [1] Brainware Research Team., **(2011)** *Mastering Advanced Java*, Brainware Consultancy Private Limited.
- [2] Brainware Research Team., **(2007)** *Mastering Java*, Brainware Consultancy Private Limited.
- [3] Keogh J., **(2002)** *J2EE: The Complete reference*, McGraw Hill Education (India) Private Limited.
- [4] Brainware Research Team., **(2011)** *Microsoft Access*, Brainware Consultancy Private Limited.