1. **Implement a java program to demonstrate creating an ArrayList, adding elements, removing elements, sorting elements of ArrayList. Also illustrate the use of toArray() method**

Program:

```
import java.util.ArrayList; import
java.util.Collections; public class
ArrayListExample { public static void
main(String[] args) {
ArrayList<Integer> al = new ArrayList<Integer>();
System.out.println("Initial size of al: " +al.size());
al.add(5); al.add(4); al.add(3); al.add(2); al.add(1);
al.add(1,51);
System.out.println("Before Sorting:"+al);
Collections.sort(al);
System.out.println("After Sorting:"+al); al.remove(2);
System.out.println("Size of al after deletions:"+al.size());
System.out.println("Elements are:"+al);
Integer ia[] = new Integer[al.size()]; ia
= al.toArray(ia);
int i;
System.out.println("The elements of the array are as follows:");
for(i=0;i<ia.length;i++) {          System.out.println(ia[i]);
}
}
}
```

Output:

```
Initial size of al: 0
Before Sorting:[5, 51, 4, 3, 2, 1]
After Sorting:[1, 2, 3, 4, 5, 51]
Size of al after deletions: 5
Elements are:[1, 2, 4, 5, 51]
The elements of the array are as follows:
 1   2  4  5  51
```

2. Develop a program to read random numbers between a given range that are multiples of 2 and 5, sort the numbers according to tens place using comparator.

Program:

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.Iterator; import
java.util.Random;
class NumberComparator implements Comparator<Integer>{
@Override
public int compare(Integer o1, Integer o2) {
        if(o1==o2)
                return 0;
        else if(o1>o2)
                return 1;
else
                return -1;
        }
}
public class Program2 { public static
void main(String[] args) {
Random random=new Random();
ArrayList al=new ArrayList();
System.out.println("The Generated Random Numbers are:="); for(int
i=1;i<=10;i++)
{
 int                x=random.nextInt(1000);
System.out.print("    "+x);     if(x%2==0  ||
x%5==0)   al.add(x);
}
System.out.print("\n");
System.out.println("Multiples of 2s and 5s Generated Random numbers generated");
Iterator itr=al.iterator(); while(itr.hasNext())
{
        System.out.print("  "+itr.next());;
}
System.out.println();
Collections.sort(al,new NumberComparator());
System.out.println("Sorted Multiples of 2s and 5s Generated Random");
Iterator itr1=al.iterator(); while(itr1.hasNext())
{ System.out.print("  "+itr1.next());;
                    }
          }

}
```

Output:

```
The Generated Random Numbers are:=
  440  723  931  864  524  879  947  779  685  95
Multiples of 2s and 5s Generated Random numbers generated
  440  864  524  685  95
Sorted Multiples of 2s and 5s Generated Random
  95  440  524  685  864
```

3. Implement a java program to illustrate storing user defined classes in collection.

Program:

```
import java.util.*; class
Address {
        private String
name,street,city;,state,code;
        Address(String n, String s, String c,String st, String cd) {
                name = n;
               street = s;
                city = c;
                state = st;
               code = cd;
        }
        public String toString() {
                return name + "\n" + street + "\n" +city + " " + state + " " + code;
        }
}
public class UserDefinedCollection { public
static void main(String[] args) {
LinkedList<Address>   ml   =   new   LinkedList<Address>();   ml.add(new
Address("J.W. West", "11 Oak Ave", "Urbana", "IL", "61801")); ml.add(new
Address("Ralph Baker", "1142 Maple Lane","Mahomet", "IL",
"61853"));
ml.add(new Address("Tom Carlton", "867 Elm St","Champaign", "IL", "61820"));
for(Address element : ml)
        System.out.println(element + "\n");
}
}
```

Output:

```
J.W. West
11 Oak Ave
Urbana IL 61801
Ralph Baker
1142 Maple Lane
Mahomet IL 61853
Tom Carlton
867 Elm St
Champaign IL 61820
```

4. Implement a java program to illustrate the use of different types of string class constructors.

Program:

```
public class StringConstructor { public
static void main(String[] args) {
System.out.println("String Type 1 Constructor Example");
char chars[ ] = { 'a', 'b', 'c','d','e','f' };
String s = new String(chars);
System.out.println(s);

System.out.println("String Type 2 Constructor Example");
s = new String(chars, 2, 3);
System.out.println(s);

System.out.println("String Type 3 Constructor Example");
String s1 = new String(chars);
String s2 = new String(s1);
System.out.println(s2);

System.out.println("String Type 4 Constructor Example");
byte ascii[ ] = {65, 66, 67, 68, 69, 70 };
s1 = new String(ascii);
System.out.println(s1); s2 =
new String(ascii, 2, 3);
System.out.println(s2);

}
}
```

Output:

```
String Type 1 Constructor Example
abcdef
String Type 2 Constructor Example cde
String Type 3 Constructor Example
abcdef
String Type 4 Constructor Example
ABCDEF
CDE
```

5. **Implement a java program to illustrate the use of different types of character extraction, string comparison, string search and string modification methods.**
   Program:

```
import java.util.Scanner;
public class StringMethods {
public static void main(String[] args) {
                    Scanner sc=new Scanner(System.in);
                    int choice,ind;
          String s1,s2;
boolean b;
char ch=0;

       do {
                    System.out.println("Enter your Choice");
                    System.out.println("1 for String Extraction");
                    System.out.println("2 for String Comparision");
                    System.out.println("4 for Modification");
                    System.out.println("3 for String Search");
                    System.out.println("5 for Exit");
                    choice=sc.nextInt();
                   switch(choice) {
                    case 1: System.out.println("Enter a String");
                              s1=sc.next();
                             ch = s1.charAt(1);
                              System.out.println(ch);
                               int start = 0;
                                int end = s1.length()-1;
                             char buf[ ] = new char[end - start];
                             s1.getChars(start, 6, buf, 0);
                             System.out.println(buf);
                             break;
                    case 2: System.out.println("Enter First String");
                                s1=sc.next();
                              System.out.println("Enter Second String");
                               s2=sc.next();
                               b=s1.equals(s2);
                            if(b)
```

```
                              System.out.println("Two Strings are Equal");
                             else
                              System.out.println("Strings are not Equal");
                              break;
          case 3:
                                System.out.println("Enter the main String");
                                s1=sc.next();
                                System.out.println("Enter the string to be searched");
                      s2=sc.next();
     ind=s1.indexOf(s2);
                                if(ind>=0)
                                    System.out.println("String is present");
                                else
                                    System.out.println("String is not Present");
                                break;
               case 4: System.out.println("Enter a String to be converted into
     Lowercase");
                            System.out.println("Lowercase String
     is:="+sc.next().toLowerCase());
                            System.out.println("Enter a String to be converted into
     Uppercase");
                            System.out.println("Uppercase String
     is:="+sc.next().toUpperCase());
                            break;
               case 5: System.out.println("Invalid Choice");
                            System.exit(1);
               }
               }
               while(ch>0 || ch<=5);
          }

     }
```

Output:

```
Enter your Choice
1 for String Extraction
2 for String Comparision
4 for Modification
3 for String Aearch
5 for Exit
```

```
1
Enter a String
welcome
e
welcom
Enter your Choice
1 for String Extraction
2 for String Comparision
4 for Modification
3 for String Aearch
5 for Exit
2
Enter First String
Hello
Enter Second String
Hai
Strings are not Equal
Enter your Choice
1 for String Extraction
2 for String Comparision
4 for Modification
3 for String Aearch
5 for Exit
3
Enter the main String
welcome
Enter the string to be searched
come
String is present
Enter your Choice
1 for String Extraction
2 for String Comparision
4 for Modification
3 for String Aearch
5 for Exit
4
Enter a String to be converted into Lowercase
WELCOME
Lowercase String is:=welcome
Enter a String to be converted into Uppercase
hello
Uppercase String is:=HELLO
Enter your Choice
```

1 for String Extraction
2 for String Comparision
4 for Modification
3 for String Aearch
5 for Exit

6. **Implement a java program to illustrate the use of different types of StringBuffer methods**

Program:

```
public class StringBufferMethods {
public static void main(String[] args) {
        StringBuffer sb=new StringBuffer("Welcome");
        System.out.println("1 String Buffer Capacity:="+sb.capacity());
        System.out.println("2 String Buffer Length"+sb.length());
        System.out.println("3 String Buffer reverse:= "+sb.reverse());
        System.out.println("4 String Buffer Char at Index:="+sb.charAt(0));
        System.out.println("5 Substring:="+sb.substring(6));
        System.out.println("6 Append to StringBuffer:="+sb.append("RIT"));
        System.out.println("7 Insert into String Buffer:"+sb.insert(6, " to "));
        System.out.println("8 String Buffer equals:="+sb.equals("Hai"));
        sb.delete(0, 5);
        System.out.println("9 After deletion String Buffer:=:"+sb);
        sb.replace(5,7, "was");
        System.out.println("10 After Replace String Buffer:="+sb);
}

}
```

Output:

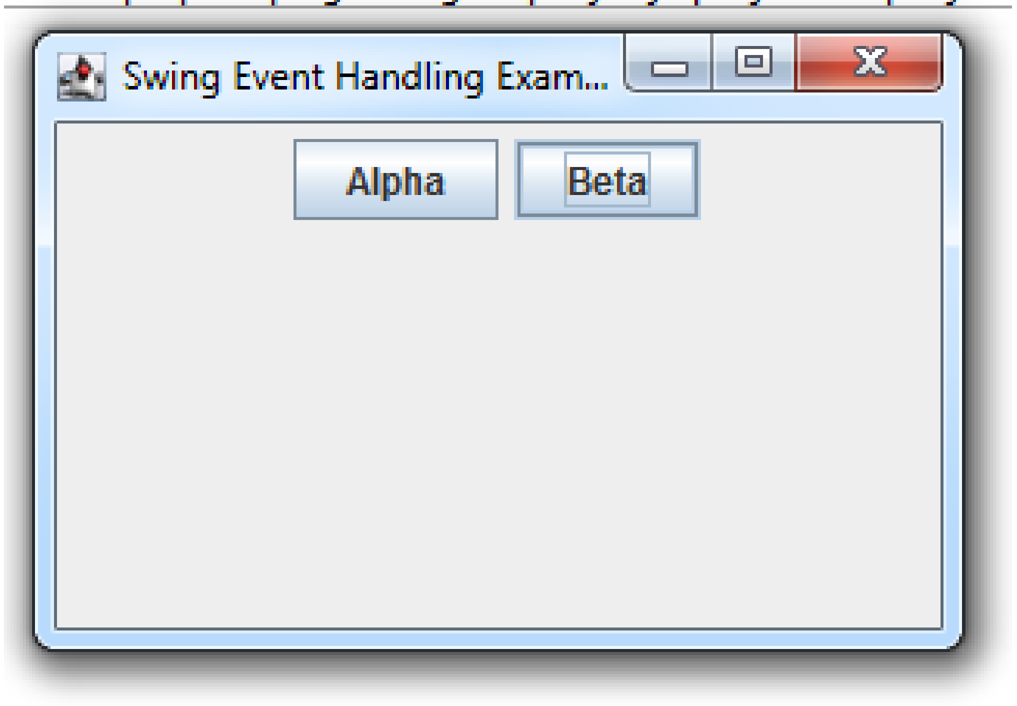
```
        1 String Buffer Capacity:=23
        2 String Buffer Length7
        3 String Buffer reverse:= emocleW
        4 String Buffer Char at Index:=e
        5 Substring:=W
        6 Append to StringBuffer:=emocleWRIT
        7 Insert into String Buffer:emocle to WRIT
        8 String Buffer equals:=false
        9 After deletion String Buffer:=:e to WRIT
        10 After Replace String Buffer:=e to wasIT
```

7. **Demonstrate a swing event handling application that creates 2 buttons Alpha and Beta and displays the text "Alpha pressed" when alpha button is clicked and "Beta pressed"**

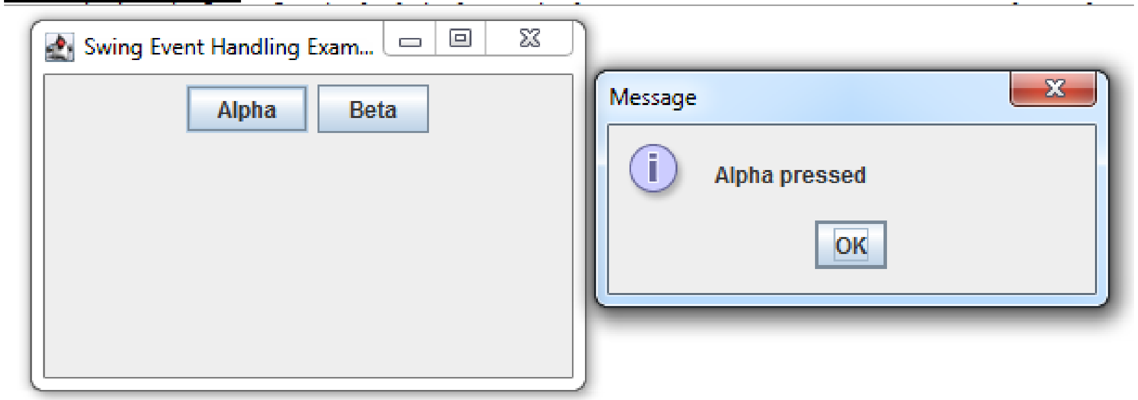
Program:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class EventDemo {
   public static void main(String[] args) {
    JFrame frame = new JFrame("Swing Event Handling Example");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    // Create a new JPanel
    JPanel panel = new JPanel();
       // Create two JButtons
    JButton alphaButton = new JButton("Alpha");
    JButton betaButton = new JButton("Beta");
       // Create an ActionListener for the Alpha button
    ActionListener alphaListener = new ActionListener() {
            @Override
     public void actionPerformed(ActionEvent e) {
            JOptionPane.showMessageDialog(frame, "Alpha pressed");
     }
      };
      ActionListener betaListener = new ActionListener() {
       @Override
       public void actionPerformed(ActionEvent e) {
       JOptionPane.showMessageDialog(frame, "Beta pressed");
       }
};
  // Add ActionListeners to the buttons
     alphaButton.addActionListener(alphaListener);
     betaButton.addActionListener(betaListener);
     // Add the buttons to the panel
     panel.add(alphaButton);
     panel.add(betaButton);
     // Add the panel to the frame
      frame.getContentPane().add(panel, BorderLayout.CENTER);
      frame.setSize(300, 200);
      frame.setVisible(true);
      }
    }
```

Output screen 1

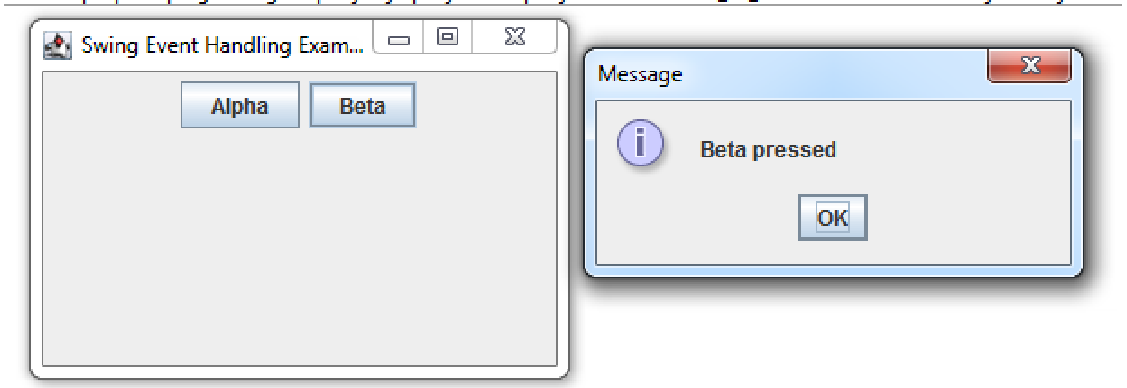


Output screen 2



Alpha pressed

Output screen 3



Beta pressed

8. A program to display greeting message on the browser "Hello UserName", "How Are You?", accept username from the client using servlet.

Program: File 1(user.html)

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor='cyan'>

<form action='./FirstServlet' method='get'>

<table>
<tr>
  <td>Enter Username:</td>
  <td><input type='text' name='uname'></td>
</tr>
<tr>
  <td><input type='reset' value='REFRESH'></td>
  <td><input type='submit' value='SIGN-IN'></td>
</tr>
</table>
</form>

</body>
</html>
```

Program: File 2(FirstServlet.html)

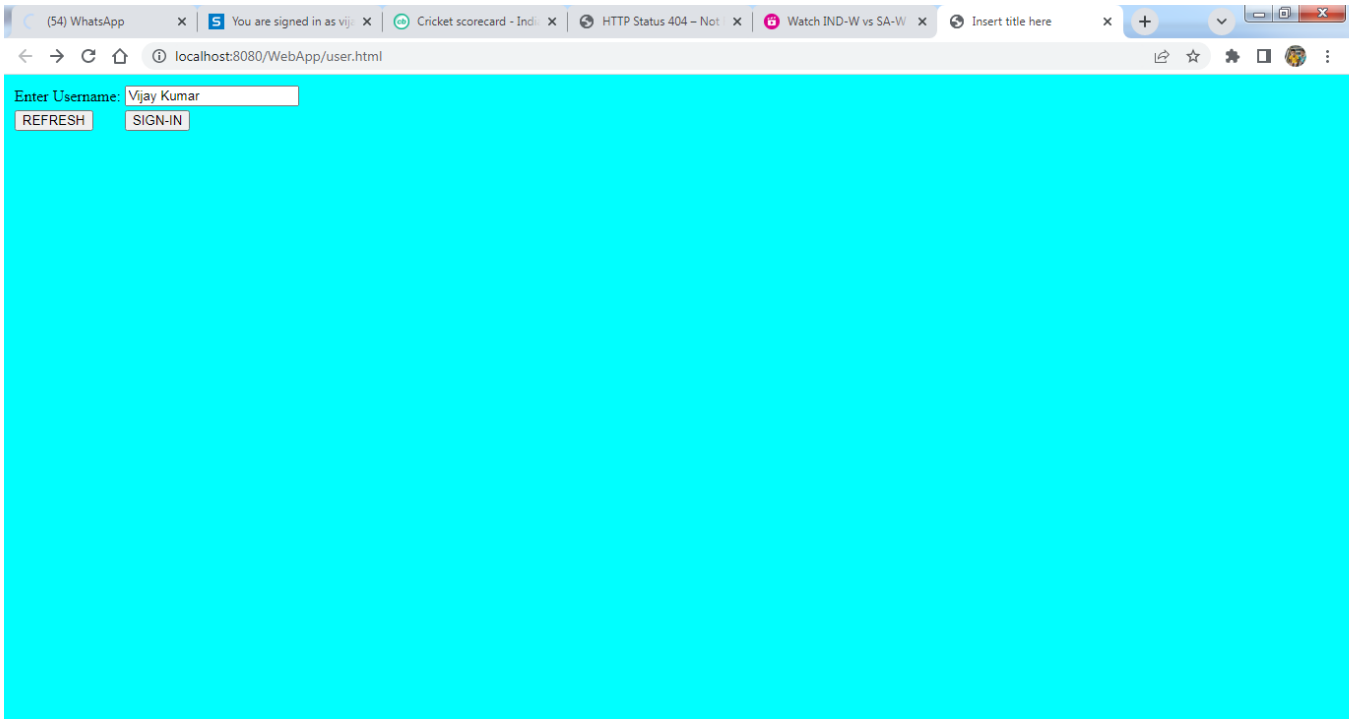
```
package com;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
```

```java
public class FirstServlet extends HttpServlet {
protected void doGet(HttpServletRequest req,
HttpServletResponse res) throws ServletException,
IOException {
    res.setContentType("text/html");
    PrintWriter out=res.getWriter();
    String uname=req.getParameter("uname");
    out.println("<body bgcolor='yellow'>");
    out.println("<b><font color='red'");
    out.println("Hai User: "+uname+" How are You?");
    }
}
```

Program: File 3(web.xml)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app>
  <servlet>
      <display-name>FirstServlet</display-name>
    <servlet-name>FirstServlet</servlet-name>
    <servlet-class>com.FirstServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>FirstServlet</servlet-name>
    <url-pattern>/FirstServlet</url-pattern>
  </servlet-mapping>
</web-app>
```

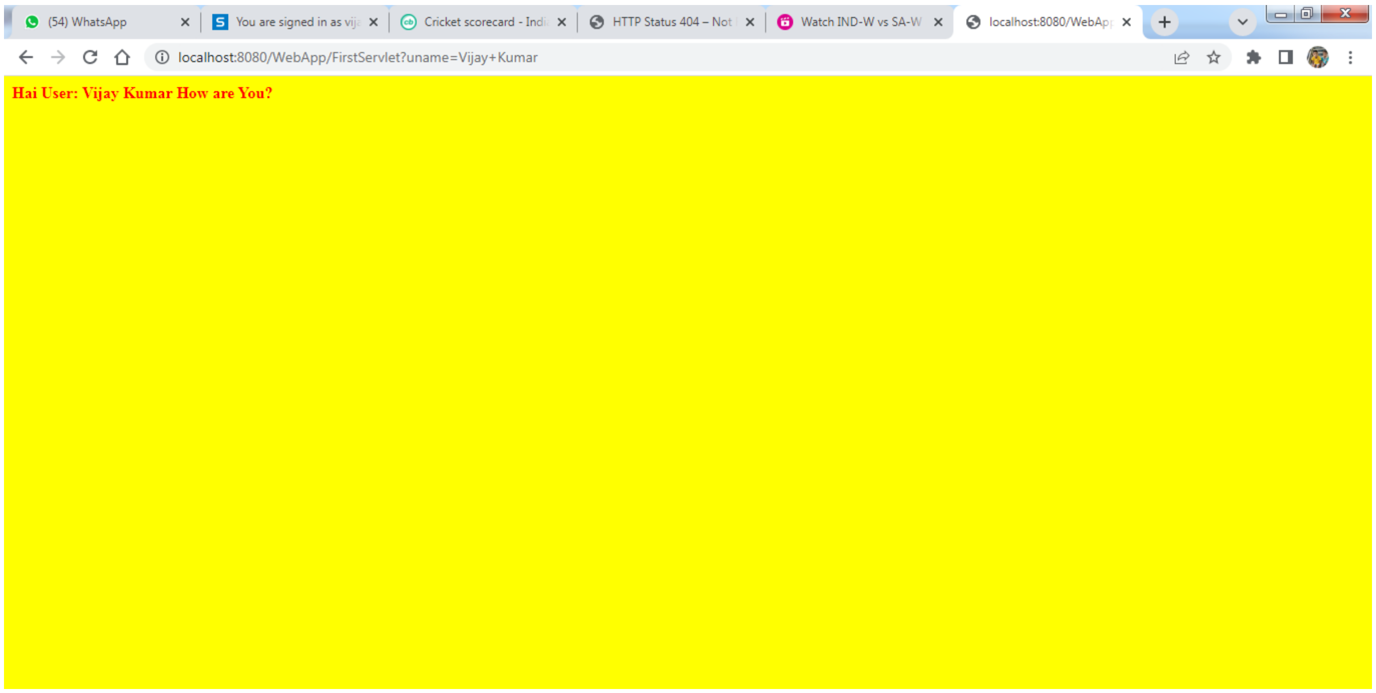
Output screen 1:



Enter Username: Vijay Kumar

REFRESH SIGN-IN

Output screen 2:



Hai User: Vijay Kumar How are You?

9. **A program A servlet program to display the name, USN, and total marks by accepting student detail**

Program: File 1(Student.html)

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor='wheat'>

<form action='./StudentData' method='get'>

<table>
<tr>
  <td>Enter Student Name:</td>
  <td><input type='text' name='sname'></td>
</tr>
<tr>
  <td>Enter Student USN:</td>
  <td><input type='text' name='usn'></td>
</tr>
<tr>
  <td>Enter Student's Total Marks:</td>
  <td><input type='text' name='smarks'></td>
</tr>
<tr>
  <td><input type='reset' value='REFRESH'></td>
  <td><input type='submit' value='Display></td>
</tr>
</table>
</form>

</body>
</html>
```

Program: File 2(StudentData.java)

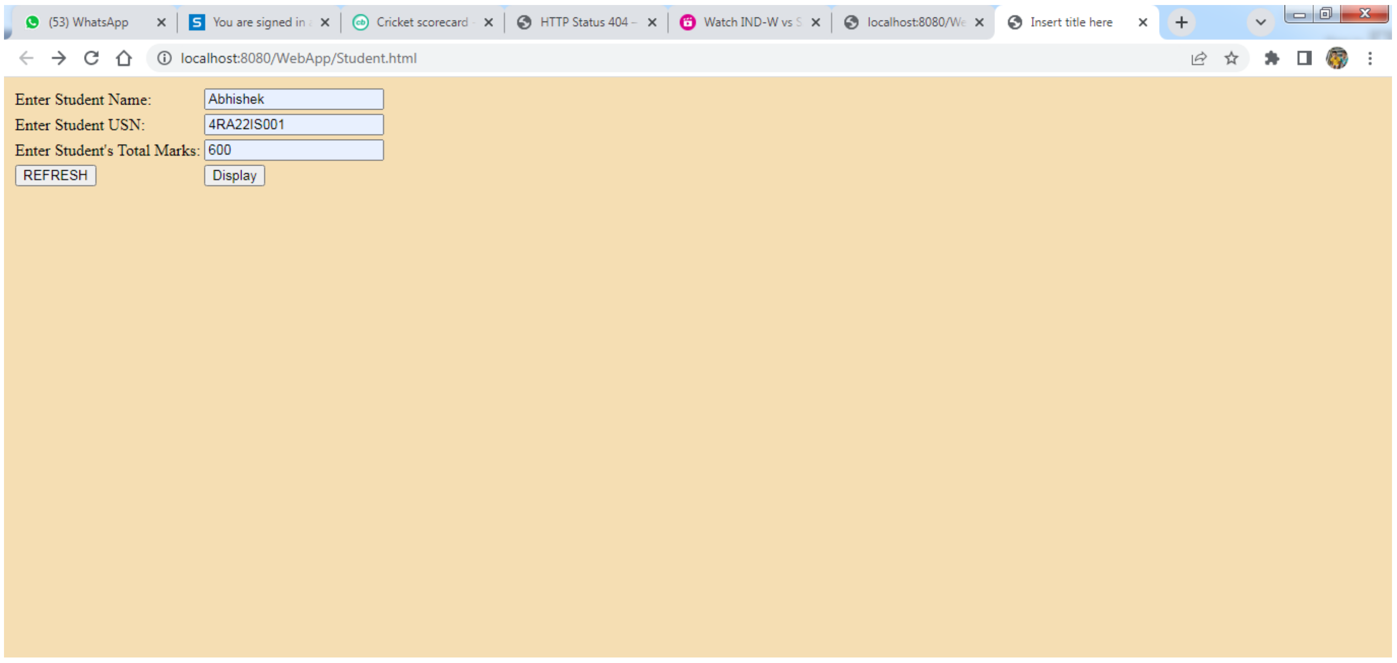
```
package com;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
```

```java
import java.io.IOException;
import java.io.PrintWriter;

public class StudentData extends HttpServlet {
protected void doGet(HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException
{
        res.setContentType("text/html");
        PrintWriter out=res.getWriter();
        String uname=req.getParameter("sname");
        String usn=req.getParameter("usn");
        String totalMarks=req.getParameter("smarks");
        int marks=Integer.parseInt(totalMarks);
        out.println("<body bgcolor='cyan'>");
        out.println("The student Details are as follows:");
        out.println("<table border='1'><tr><td>");
        out.println("Student Name:
        "+uname+"</td></tr><tr><td>usn:"+usn+"</td></tr>");
                                        out.println("<tr><td>Total
        Marks:="+totalMarks+"</td></tr></table>");

}

}
```

Program: File 3(web.xml)

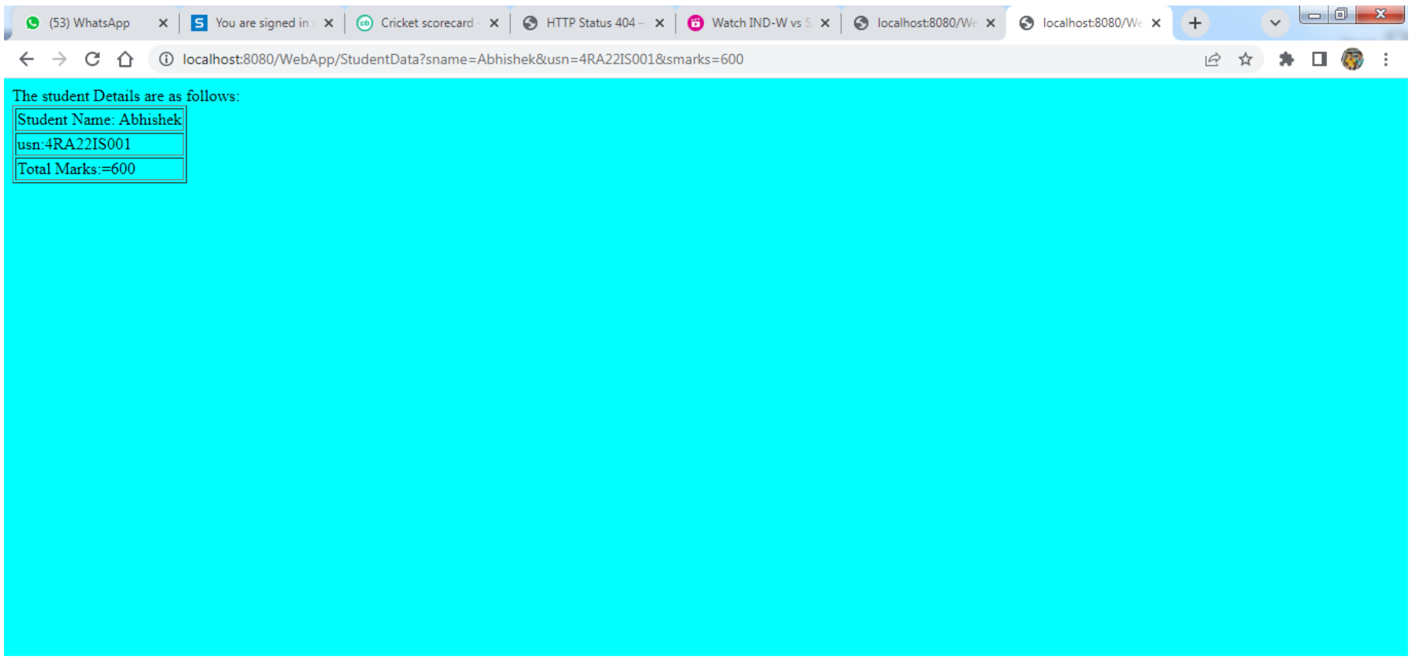
```xml
<web-app>
<servlet>
   <description></description>
   <display-name>StudentData</display-name>
   <servlet-name>StudentData</servlet-name>
   <servlet-class>com.StudentData</servlet-class>
  </servlet>
  <servlet-mapping>
   <servlet-name>StudentData</servlet-name>
   <url-pattern>/StudentData</url-pattern>
  </servlet-mapping>
</web-app>
```

Output screen 1:



Enter Student Name: Abhishek
Enter Student USN: 4RA22IS001
Enter Student's Total Marks: 600
REFRESH Display

Output screen 2:



The student Details are as follows:

Student Name: Abhishek
usn:4RA22IS001
Total Marks:=600

10. A Java program to create and read the cookie for the given cookie name as "EMPID" and its value as "AN2356".

Program: File 1 Client File (cookie.html)

```
<html>
<head>
<title>Insert title here</title>
</head>
<body>
<form action="servlet1" method="post">
Name:<input type="text" name="userName"/><br/>
<input type="submit" value="go"/>
</form>
</body>
</html>
```

Program: File 2 Server File (Servlet1.java)

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Servlet1 extends HttpServlet
{
public void doPost(HttpServletRequest reqreq, HttpServletResponse res)throws
ServletException,IOException
{
		res.setContentType("text/html");
		PrintWriter pw = res.getWriter();
		String n=request.getParameter("userName");
		pw.print("Welcome "+n);
		Cookie ck=new Cookie("uname",n);//creating cookie object
		res.addCookie(ck);//adding cookie in the response
		pw.print("<form action='servlet2' method='post'>");
		pw.print("<input type='submit' value='go'>");
		pw.print("</form>");
}
}
```

Program :File 3 Server File (Servlet2.java)

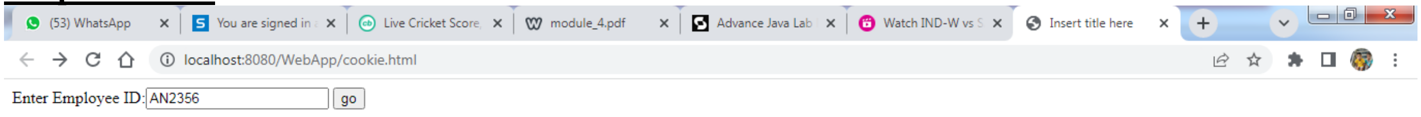
```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Servlet2 extends HttpServlet
{
```

```
public void doPost(HttpServletRequest req, HttpServletResponse res)throws
ServletException,IOException
{
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        Cookie ck[]=req.getCookies();
        out.print("Hello "+ck[0].getValue());
 }

}
```

Program: File 4 Deployment descriptor File (web.xml)

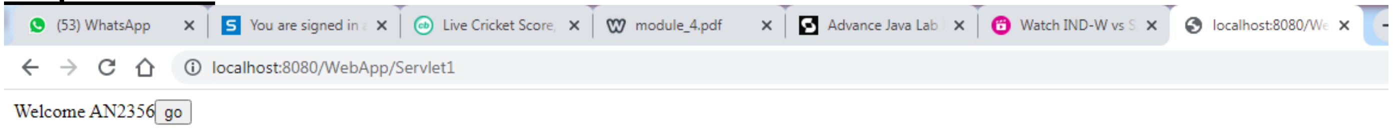
```
<web-app>
        <servlet>
                <servlet-name>s1</servlet-name>
                <servlet-class>Servlet1</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>s1</servlet-name>
                <url-pattern>/servlet1</url-pattern>
        </servlet-mapping>
        <servlet>
                <servlet-name>s2</servlet-name>
                <servlet-class>Servlet2</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>s2</servlet-name>
                <url-pattern>/servlet2</url-pattern>
        </servlet-mapping>
</web-app>
```

Output screen1


localhost:8080/WebApp/cookie.html

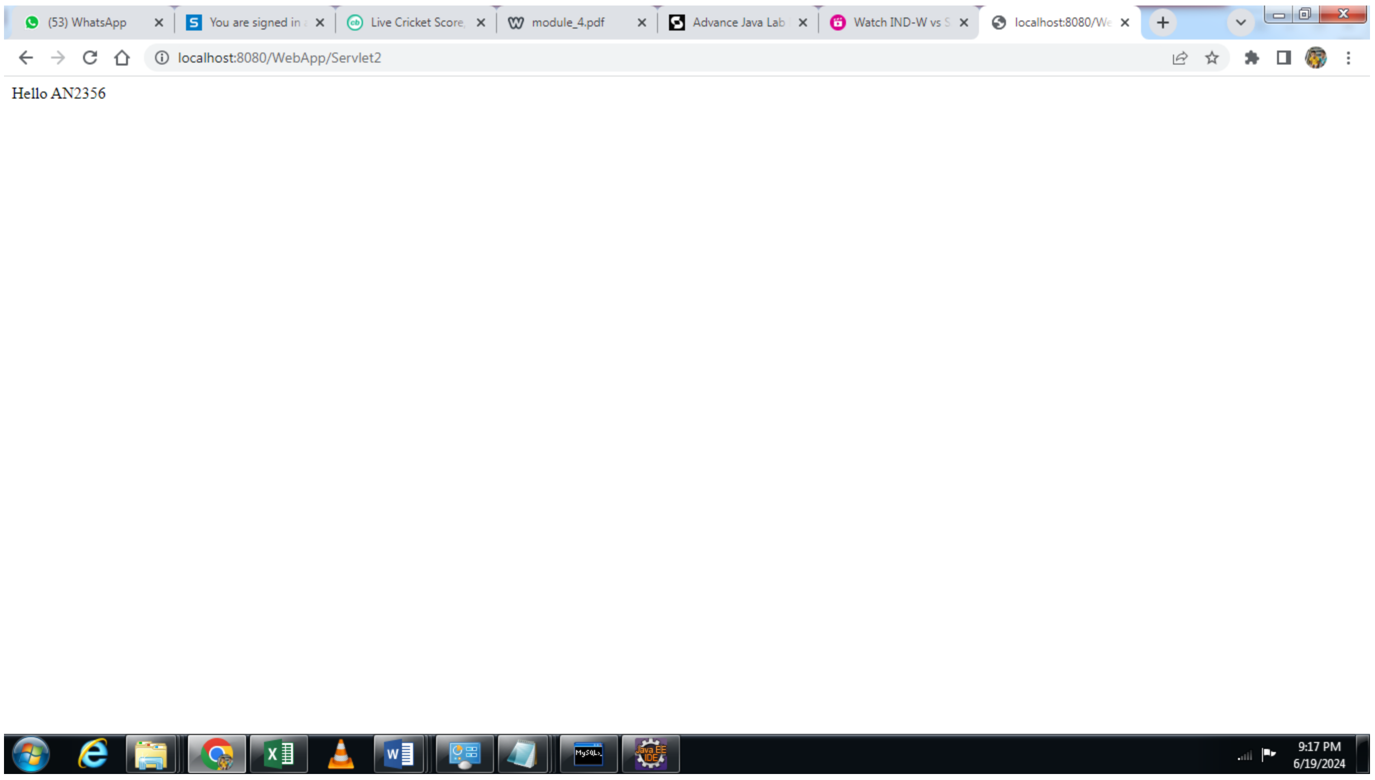
Enter Employee ID: AN2356 go

Output screen2



Welcome AN2356

Output screen3



Hello AN2356

11. Write a JAVA Program to insert data into Student DATA BASE and retrieve info based on particular queries(For example update, delete, search etc…).

Program

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Scanner;
public class StudentDatabase {
        public static void main(String[] args) {
                Scanner sc=new  Scanner(System.in);
                int choice=0,age,status,flag=0;
                String usn,name,sql;
                Statement st=null;
                Connection con=null;
                ResultSet rs=null;
                try {
                do {
                System.out.println("Enter your choice ");
                Class.forName("com.mysql.cj.jdbc.Driver");
                con=DriverManager.getConnection("jdbc:mysql://localhost:3306/db", "root", "");
                System.out.println("1 for inserting data into a Table");
                System.out.println("2 for Searching  a record from the Table");
                System.out.println("3 for Updating  a record into the Table");
                System.out.println("4 for deleting a record");
                System.out.println("5 for Exit");
                choice=sc.nextInt();
                switch(choice) {
                case 1: System.out.println("Enter Student USN");
                        usn=sc.next();
                        System.out.println("Enter name of the Student");
                        name=sc.next();
                        System.out.println("Enter Student age:");
                        age=sc.nextInt();
                        st=con.createStatement();
                        sql="insert into student values('"+usn+"','"+name+"',"+age+")";
                        System.out.println(sql);
                        status=st.executeUpdate(sql);
                        if(status>0) {
                                System.out.println("Record inserted successfully");
                         }else {
                                System.out.println("REcord not inserted");
                         }
                             break;
                case 2: System.out.println("Enter usn");
                         usn=sc.next();
                         sql="select name,age from student where usn='"+usn+"'";
                         st=con.createStatement();
                         rs=st.executeQuery(sql);
                         System.out.println(rs);
                         while(rs.next())
                         flag=1;
                        System.out.println("Name of the Student:="+rs.getString(1)+" Age of the
                        Student:="+rs.getInt(2));
                         }
                         if(flag==0){
                                System.out.println("Usn Doesnot exists");
                         }
                         break;
                case 3: System.out.println("Enter Usn");
                         usn=sc.next();
                         System.out.println("Enter name of the Student to be updated");
                         name=sc.next();
                         sql="update student set name='"+name+"' where usn='"+usn+"'";
                         st=con.createStatement();
                         status=st.executeUpdate(sql);
                         if(status>0) {
                              System.out.println("Record Updated successfully");
                         }else {
                              System.out.println("Record not updated");
                         }
                          break;
              case 4: System.out.println("Enter the usn");
                          usn=sc.next();
                           sql="delete from student where usn='"+usn+"'";
                           st=con.createStatement();
                           status=st.executeUpdate(sql);
                           if(status>0) {
                              System.out.println("record deleted successfully");
                              }else {
                                System.out.println("Record doesnot exists");
                              }
                             break;
                case 5:
                                   System.out.println("Invalid choice");
                                   System.exit(0);
                        }
                }while(choice>0 || choice<=5);
                }catch(Exception e) {
                        e.printStackTrace();
```

```
                }


        }
}
```

Output:

```
Enter your choice
1 for inserting data into a Table
2 for Searching  a record from the Table
3 for Updating  a record into the Table
4 for deleting a record
5 for Exit
1
Enter Student USN
4RA22IS061
Enter name of the Student
Vinay
Enter Student age:
20
insert into student values('4RA22IS061','Vinay',20)
Record inserted successfully
Enter your choice
1 for inserting data into a Table
2 for Searching  a record from the Table
3 for Updating  a record into the Table
4 for deleting a record
5 for Exit
2
Enter usn
4RA22IS061
com.mysql.cj.jdbc.result.ResultSetImpl@72035809
Name of the Student:=Vinay Age of the Student:=20
Enter your choice
1 for inserting data into a Table
2 for Searching  a record from the Table
3 for Updating  a record into the Table
4 for deleting a record
5 for Exit
3
Enter Usn
4RA22IS061
Enter name of the Student to be updated
VinayK
Record Updated successfully
Enter your choice
1 for inserting data into a Table
```

```
2 for Searching  a record from the Table
3 for Updating  a record into the Table
4 for deleting a record
5 for Exit
2
Enter usn
4RA22IS061
com.mysql.cj.jdbc.result.ResultSetImpl@6fa51cd4
Name of the Student:=VinayK Age of the Student:=20
Enter your choice
1 for inserting data into a Table
2 for Searching  a record from the Table
3 for Updating  a record into the Table
4 for deleting a record
5 for Exit
4
Enter the usn
4RA22IS061
record deleted successfully
Enter your choice
1 for inserting data into a Table
2 for Searching  a record from the Table
3 for Updating  a record into the Table
4 for deleting a record
5 for Exit
5
Invalid choice
```

12. A program to design the Login page and validating the USER_ID and PASSWORD using JSP and DataBase).

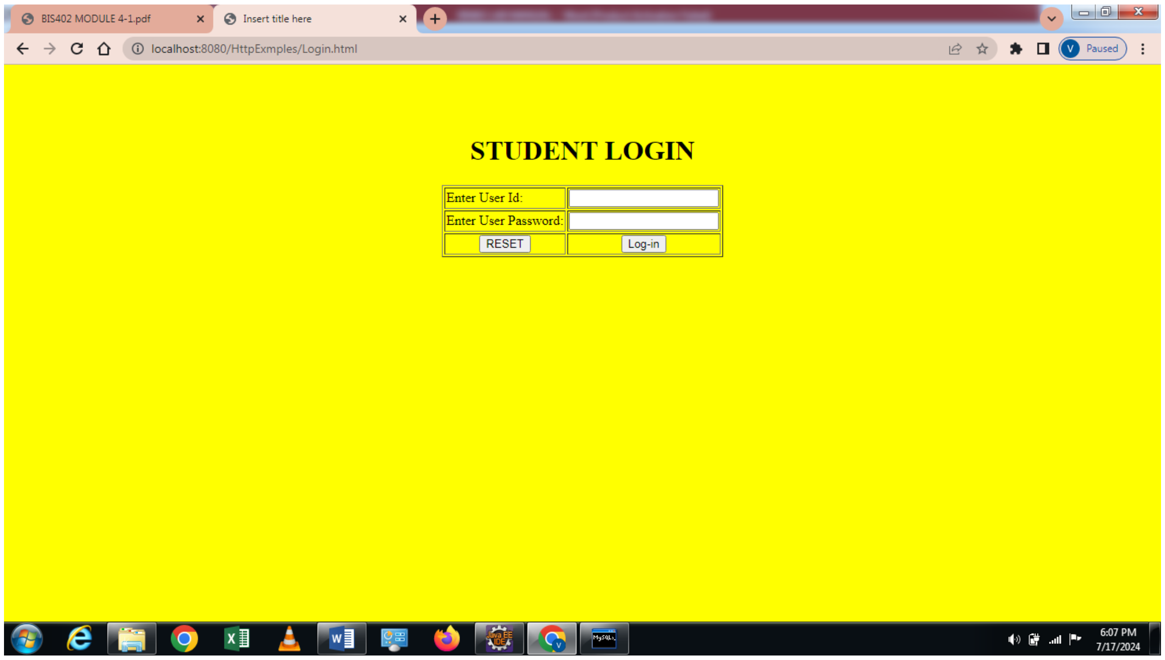
Login page(Login.html>:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor='yellow'>
<form action='authenticate.jsp' method="get">
<br><br><br>
         <h1><center>STUDENT LOGIN</center></h1>
       <table border='1' align='center'>
       <tr>
            <td>Enter User Id:</td>
            <td><input type='text' name='uname'></td>
       </tr>
       <tr>
            <td>Enter User Password:</td>
            <td><input type='password' name='upass'></td>
       </tr>
       <tr>
            <td>
                 <center>
                 <input type='reset' value='RESET'>
                 </center>
            </td>
            <td>
                 <center>
                 <input type='submit' value='Log-in'>
                 </center>
             </td>
       </tr>
       </table>
</form>
</body>
</html>
```

authenticate.jsp(Validating user)

```
<%@ page  language="java" contentType="text/html; charset=ISO-8859-1"
import="java.sql.*" pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor='wheat'>
<%
String uname,pass;
boolean flag=false;
%>
<%
uname=request.getParameter("uname");
pass=request.getParameter("upass");
try{
Class.forName("com.mysql.cj.jdbc.Driver");
Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/db","root
","");
Statement st=con.createStatement();
ResultSet rs=st.executeQuery("select username,password from login");
while(rs.next()){
      String user1=rs.getString(1);
      String pwd=rs.getString(2);
      if(user1.equals(uname) && pass.equals(pwd)){
                        flag=true;
                        break;
      }
}
}catch(Exception e){
           e.printStackTrace();
}
if(flag){
      out.println("hai user"+uname+" you are welcome ");
}else{
      out.println("you are not welcome");
      out.println("<a href='Login.html'>Click here to Relogin</a>");
}
%>
</body>
</html>
```

Screen1(Login.html)

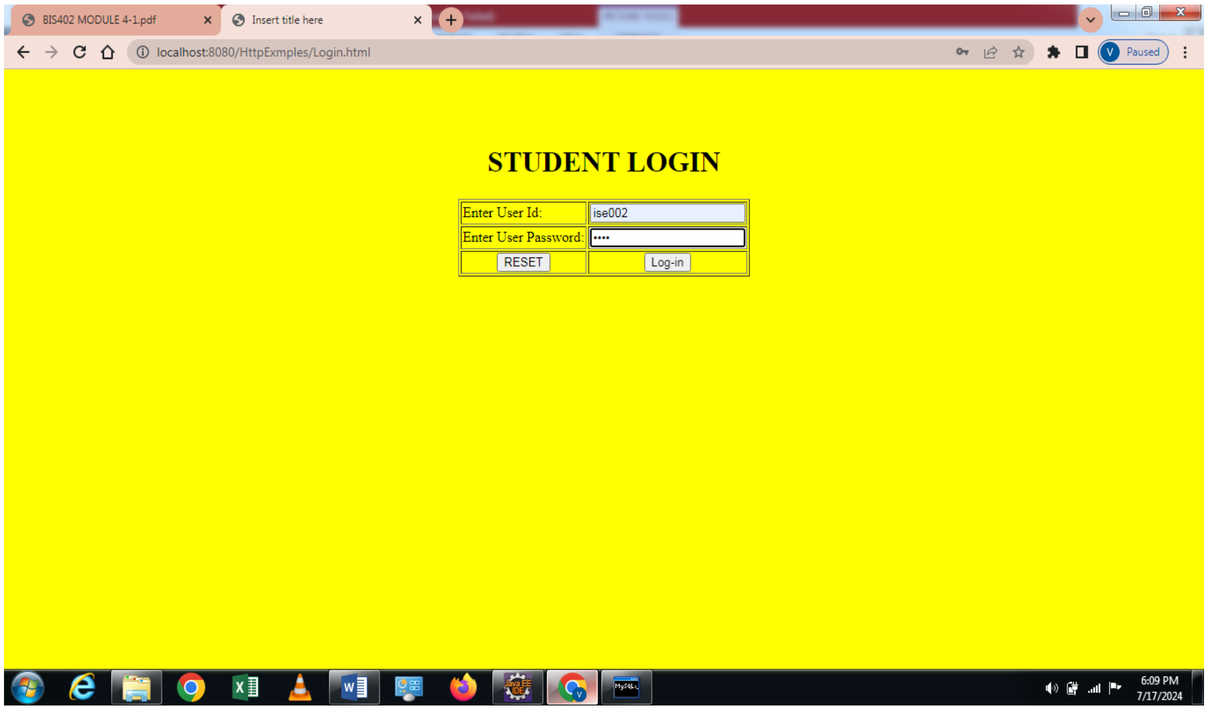

localhost:8080/HttpExmples/Login.html

STUDENT LOGIN

Enter User Id:	
Enter User Password:	
RESET	Log-in

.

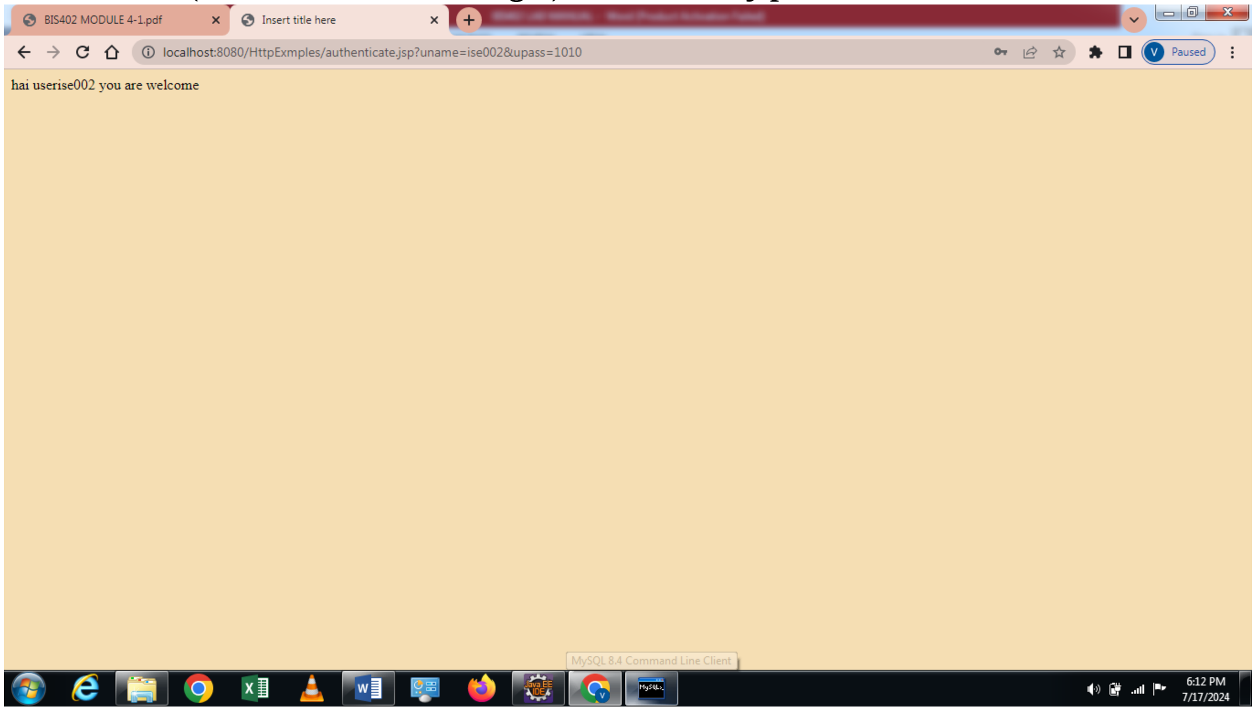
Screen-2


localhost:8080/HttpExmples/Login.html

STUDENT LOGIN

Enter User Id:	ise002
Enter User Password:	••••
RESET	Log-in

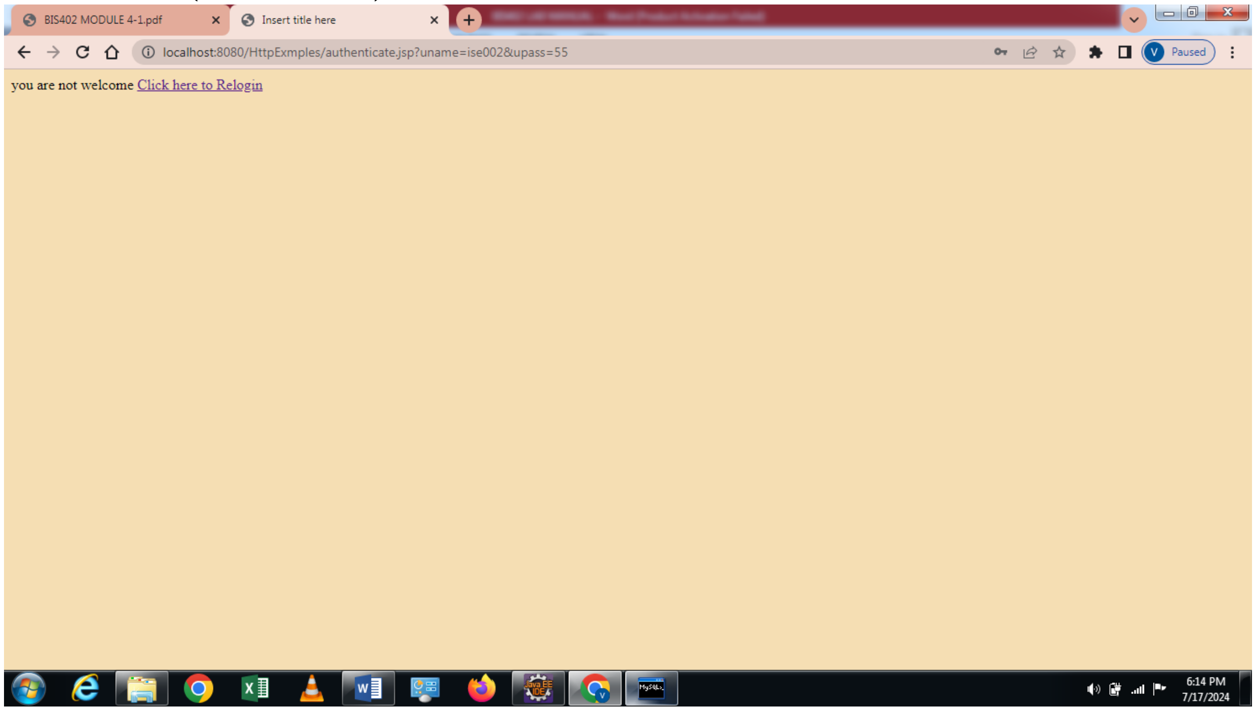
Screen shot-3(valid user successful login) authenticate.jsp



localhost:8080/HttpExmples/authenticate.jsp?uname=ise002&upass=1010

```
hai userise002 you are welcome
```

Screen shot-4(In-valid user)



localhost:8080/HttpExmples/authenticate.jsp?uname=ise002&upass=55

```
you are not welcome Click here to Relogin
```

VIVA QUESTIONS (ADVANCED JAVA-BIS402)

1. **Define Collection**.
   Collection is a framework to handle group of objects.

2. **What is an Iterator?**
   Iterator is an interface present in **java.util** package. Used for accessing the elements within a collection, one at a time.

3. **What is ListIterator?**
   ListIterator is an interface present in **java.util** package. Used for accessing the elements within a collection, one at a time in two forward as well as backward direction.

4. **What is Comparator in java?**
   Comparator is an interface in **java.util** package to compare to objects and sort the data either in ascending or descending or alphabetical order.

5. **What is Map?**
   Map is a interface present in **java.util** package. Used to store group of objects in association with keys and values in pair.

6. **Define String**.
   String is a final class present in **java.lang** package. Contains sequence of characters enclosed between pair of double quotes. String is immutable in nature.

7. **Define toString method in java**.
   toString method is called automatically without creating a object of a class in which it is defined. We want to override **toString( )** and provide our own string representations.This method returns a String which describes the object the syntax is as follows: public **String toString**( )

8. **What is the use of charAt() method of String class? List the exception thrown by this method**.
   This method is used to extract a character from the given string object. It throws **StringIndexOutOfBoundsException** is index is not present in the invoked object.

9. **Define StringBuffer**.
   StringBuffer represents growable and writeable character sequences. **StringBuffer** may have characters and substrings inserted in the middle or appended to the end. **StringBuffer** will automatically grow to make room for such additions and often has more characters preallocated than are actually needed, to allow room for growth.

10. **What is StringBuilder?**
    StringBuilder is a class present in java.lang package. StringBuilder is *non-synchronized* i.e. not thread safe. It means two threads can call the methods of StringBuilder simultaneously.

11. **Define Servlet.**
    It is a server side program used to read clients request and processes the request and generates response back to the client. It is a java file created with .java extension. To execute servlet we require server such as apache tomcat and jboss etc.

12. **List the life cycle methods of servlet.**
    Init() method, **service**() method **destroy**() method.

13. **What is web.xml in web application?**
    Deployment descriptor is a file located in the WEB-INF directory that controls the behavior of a java servlet and java server pages. The file is called the web.xml file and contains the header, DOCTYPE, and web app element. The web app element should contain a servlet element with three elements. These are servlet name, servlet class, and url-pattern. The servlet name elements contain the name used to access the java servlet. The servlet class is the name of the java servlet class.url pattern is associated with the path specified by the client side programming in action is matched with url pattern in web.xml.

14. **What is JSP?**
    A jsp is java server page is server side program that is similar in design and functionality to a java servlet.A JSP is called by a client to provide web services, the nature of which depends on client application.

15. **What are sessions and cookies?**
    Session: HTTP is a stateless protocol. Each request is independent of the previous one. However, in some applications, it is necessary to save state information so that information can be collected from several interactions between a browser and a server. Sessions provide such a mechanism. A session can be created via the **getSession( )** method **HttpServletRequest**. An **HttpSession** object is returned.

    Cookies: By default, each request is considered as a new request. In cookies technique, we add cookie with response from the servlet. So cookie is stored in the cache of the browser. After that if request is sent by the user, cookie is added with request by default. Thus, we recognize the user as the old user.

16. **What is JDBC?**
    JDBC Package contained in two packages. The first package is called java.sql and contains core java data objects of the JDBC API .These include java data objects that provide the basics for connecting to the DBMS and interacting with data stored in DBMS.

17. **List the types of Statement objects used in jDBC.**
    Statement,PreparedStatement and CallableStatement

18. **Define Stored Procedure.**
    A procedure (often called a stored procedure) is a collection of pre-compiled SQL statements stored inside the database. It is a subroutine or a subprogram in the regular computing language. A procedure always contains a name, parameter lists, and SQL statements.