**EXPERIMENT:** **1**

**Student Registration form**

**<html>**

**<head><title>Student Registration form</title>**

**<center>**

**<table bg color="green" text="while"><h1>Student Registration form</h1>**

**<form>**

**<table bg color="green" border="10" border color="red">**

**<tr>**

**<td>student name<input type="text" name="pname"></tb>**

**</tr>**

**<tr>**

**<td>address:<hr>**

**<textarea rows="6" cols="38"></textarea></td>**

**<tr>**

**<td>mobile number:<input type="text" name="sname"></td>**

**</tr>**

**<td>Email/id:<input type="text" name="sname"></td>**

**</tr>**

**<td>course**

**<select>**

**<option>select ....</option>**

**<option>b.tech</option>**

**<option>m.tech</option>**

**</select></td></tr>**

**<td>branch**

**<select>**

**<option>select...</option>**

**<option>cse</option>**

**<option>ece</option>**

**<option>eee</option>**

**</select></td></tr>**

**<td>city**

**<select>**

**<option>select...</option>**

**<option>Hyderabad</option>**

**<option>vizag</option>**

**</select></td>**

**</tr>**

**<td>state**

**<select>**

**<option>select...</option>**

**<option>telangana</option>**

**<option>andrapradesh</option>**

**</select></td>**

**</tr>**

**<select></td>**

**</tr>**

**<select></td>**

**</tr>**

**<tr>**

**<td><input type ="Button" name="submit" value="submit">**

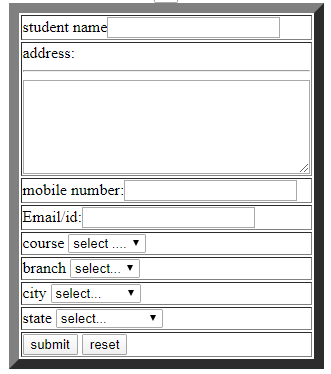
**<input type="Button" name="reset" value="reset"></td>**

**</tr></form></table></body></head>**

**</html>**

**OutPut:**

****

****

**EXPERIMENT:** **2**

**TIME TABLE**

**<html>**

**<body bgcolor="black" text="red">**

**<center>**

**<table bgcolor="yellow" border="20" border color="violet">**

**<tr>**

**<center><h1><i>Timetable</h1></center>**

**</tr>**

**<tr>**

**<th>Day</th>**

**<th>9:15 - 10:05</th>**

**<th>10:05 - 10:55</th>**

**<th>10:55 - 11:45</th>**

**<th>11:45 - 12:35</th>**

**<th>1:30 - 2:20</th>**

**<th>2:20 - 3:10</th>**

**<th>3:10 - 4:00</th>**

**</tr>**

**<tr>**

**<td>Monday</td>**

**<td>CNS</td>**

**<td>DP</td>**

**<td rowspan ="6"> Break</td>**

**<td>CD</td>**

**<td colspan ="3" align="center">CNS/WT LAB</td>**

**</tr>**

**<tr>**

**<td>Tuesday</td>**

**<td>DCCN/EDB</td>**

**<td colspan ="3" align="center">AEC LAB</td>**

**<td>CNS</td>**

**<td>DP</td>**

**<td>CD</td>**

**</tr>**

**<tr>**

**<td>Wednesday</td>**

**<td>WT</td>**

**<td>DCCN/EDB</td>**

**<td>CD</td>**

**<td>DP</td>**

**<td colspan ="3" align="center">CNS/WT LAB</td>**

**</tr>**

**<tr>**

**<td>Thorsday</td>**

**<td>DP</td>**

**<td>WT</td>**

**<td>DCCN/EDB</td>**

**<td>WT</td>**

**<td>CD</td>**

**<td>DCCN/EBD</td>**

**</tr>**

**<tr>**

**<td>Friday</td>**

**<td>WT</td>**

**<td>CNS</td>**

**<td>WT</td>**

**<td>CD</td>**

**<td colspan ="3" align="center">PYTHON/GATE</td>**

**</tr>**

**<tr>**

**<td>Satuday</td>**

**<td colspan ="3" align="center">EPICS/SSDC</td>**

**<td>MENTOR</td>**

**<td>SPORTS/LIBARY</td>**

**<td colspan ="2" align="center">AFFINITY</td>**

**<center>**

**</table>**

**</body>**

**</html>**

**TIME TABLE OUTPUT:**

****

**EXPERIMENT:** **3**

**Aim:** Install Apache Web Server, MYSQL, PHP (XAMPP)

**Install XAMPP Web Server**

**Installing XAMPP on Ubuntu 13.10, 13.04, 12.10, 12.04 and Linux Mint 13/14/15/16**

Now let’s see how to installing XAMPP 1.8.3 with PHP 5.5 in Ubuntu and derivatives, but no other previous Xampp 1.8.2 with PHP 5.4 can also be installed by following the same instructions. XAMPP 1.8.3, Tested on Ubuntu 13.10 (64 bit) without problems.

Step 1. Open terminal and download XAMPP 1.8.3 package.

**for 32-bit:**

wget http://sourceforge.net/projects/xampp/files/XAMPP%20Linux/1.8.3/xampp-linux-1.8.3-2-installer.run/download

for 64-bit:

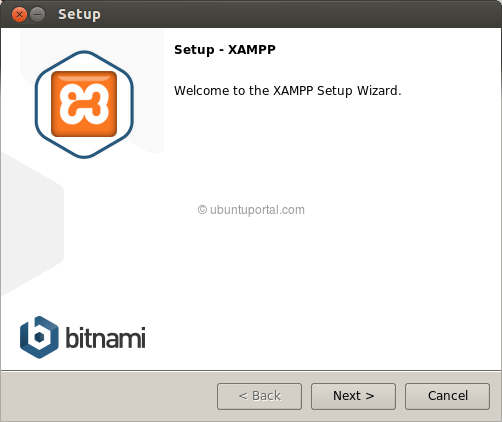
wget http://sourceforge.net/projects/xampp/files/XAMPP%20Linux/1.8.3/xampp-linux-x64-1.8.3-2-installer.run/download

Step 2. After that, Change xampp package installer to executable then run to installing with following command:

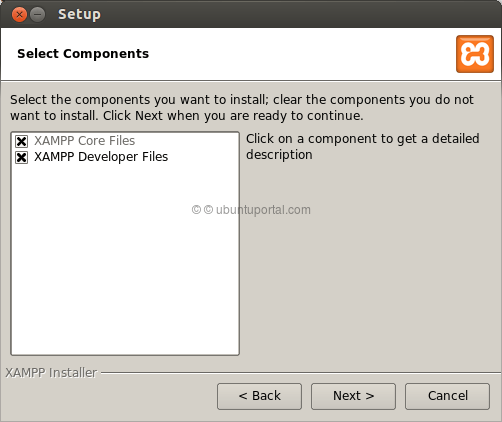
sudo chmod +x xampp-linux-x64-1.8.3-2-installer.run

sudo ./xampp-linux-x64-1.8.3-2-installer.run

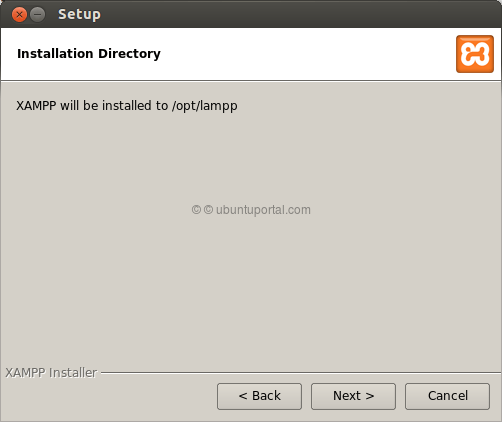
The window installation wizard will appear:



Click “Next”. Another appears:



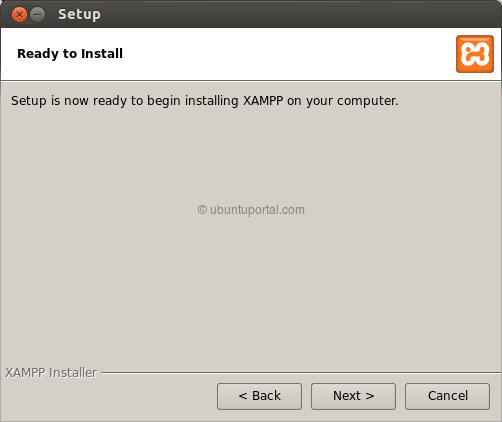
Click “Next” again. And you will go to the next window:



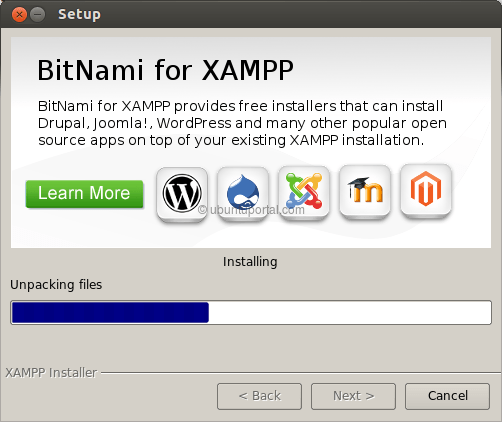
Click “Next” again. And will to the following:



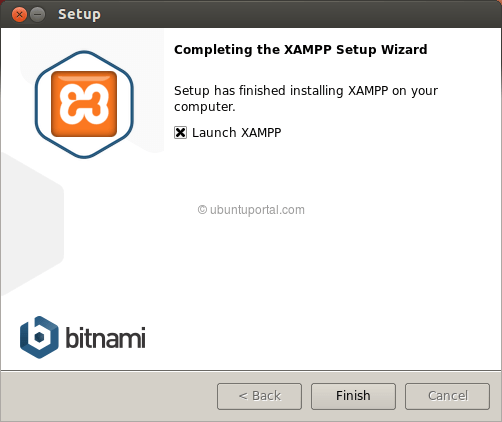
Uncheck: “Learn more about BitNami for XAMPP”  
And click “Next”



Click Next again. Wait for the installation:



When finished, this will be the last window:



When you checked “launch XAMPP” before clicking finish A page will open in your browser,  If the page does not open automatically, go into your browser’s address and type http://localhost/xampp

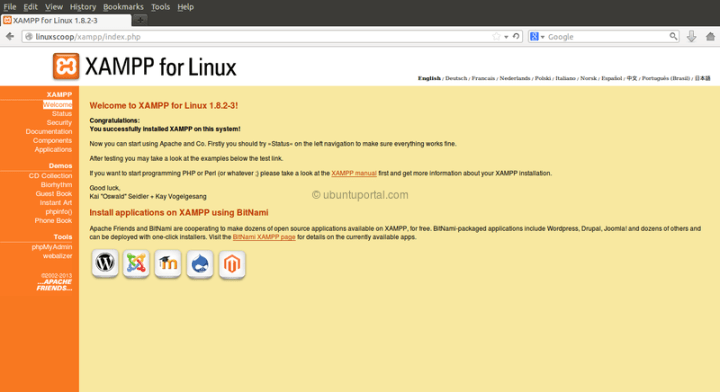
To stop the XAMPP service:

sudo /opt/lampp/lampp stop

To start the XAMPP service:

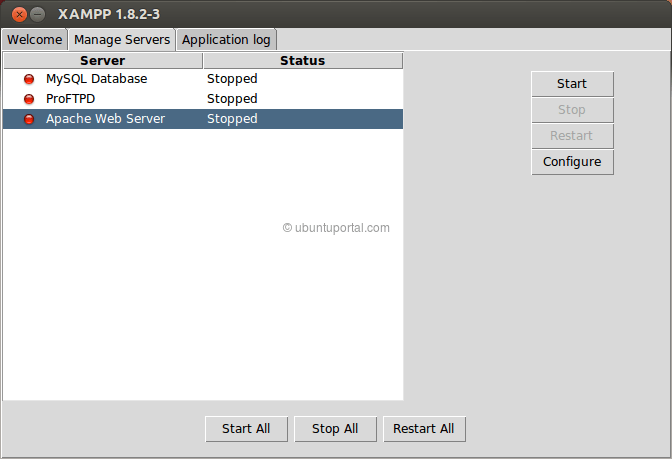
sudo /opt/lampp/lampp start

To open the page of XAMPP, whenever you want, type in the address bar of your browser: http://localhost/xampp/



sudo /opt/lampp/lampp restart

simply use the graphical interface of the program to start and stop the web server



**EXPERIMENT:** **4**

**TOMCAT INSTALLATION on Ubuntu**

**There are two basic ways to install Tomcat on Ubuntu:**

* Install through apt-get. This is the simplest method.
* Download the binary distribution from the Apache Tomcat [site](http://tomcat.apache.org/download-70.cgi). This guide does not cover this method; refer to [Apache Tomcat Documentation](http://tomcat.apache.org/tomcat-7.0-doc/index.html) for instructions.

For this tutorial, we will use the simplest method: apt-get. Please note that this will install the latest release of Tomcat that is in the official Ubuntu repositories, which may or may not be the latest release of Tomcat. If you want to guarantee that you are installing the latest version of Tomcat, you can always download the latest binary distribution.

## Step One — Prerequisites

Before you begin with this guide, you should have a separate, non-root user account set up on your server. You can learn how to do this by completing steps 1-4 in the [initial server setup](https://www.digitalocean.com/community/articles/initial-server-setup-with-ubuntu-14-04) for Ubuntu 14.04. We will be using the demo user created here for the rest of this tutorial.

## Step Two - Install Tomcat

The first thing you will want to do is update your apt-get package lists:

* sudo apt-get update

Now you are ready to install Tomcat. Run the following command to start the installation:

* sudo apt-get install tomcat7

Answer yes at the prompt to install tomcat. This will install Tomcat and its dependencies, such as Java, and it will also create the tomcat7 user. It also starts Tomcat with its default settings.

Let's make a quick change to the Java options that Tomcat uses when it starts. Open the Tomcat7 parameters file:

* sudo nano /etc/default/tomcat7

Find the JAVA\_OPTS line and replace it with the following. Feel free to change the Xmx and MaxPermSizevalues—these settings affect how much memory Tomcat will use:

/etc/default/tomcat7 — JAVA\_OPTS

JAVA\_OPTS="-Djava.security.egd=file:/dev/./urandom -Djava.awt.headless=true -Xmx512m -XX:MaxPermSize=256m -XX:+UseConcMarkSweepGC"

Save and exit.

Now restart Tomcat with this command:

* sudo service tomcat7 restart

Tomcat is not completely set up yet, but you can access the default splash page by going to your domain or IP address followed by :8080 in a web browser:

Open in web browser:

http://server\_IP\_address:8080

You will see a splash page that says "It works!", in addition to other information. Now we will go deeper into the installation of Tomcat.

## Step Three - Installing Additional Packages

*Note:* This section is not necessary if you are already familiar with Tomcat and you do not need to use the web management interface, documentation, or examples. If you are just getting into Tomcat for the first time, please continue.

With the following command, we will install the Tomcat online documentation, the web interface (manager webapp), and a few example webapps:

* sudo apt-get install tomcat7-docs tomcat7-admin tomcat7-examples

Answer yes at the prompt to install these packages. We will get into the usage and configuration of these tools in a later section. Next, we will install the Java Development Kit.

## Step Four - Install Java Development Kit (Optional)

If you are planning on developing apps on this server, you will want to be sure to install the software in this section.

The Java Development Kit (JDK) enables us to develop Java applications to run in our Tomcat server. Running the following command will install openjdk-7-jdk:

* sudo apt-get install default-jdk

In addition to JDK, the Tomcat documentation suggests also installing Apache Ant, which is used to build Java applications, and a source control system, such as git. Let's install both of those with the following command:

* sudo apt-get install ant git

For more information about Apache Ant, refer to [the official manual](http://ant.apache.org/manual/index.html). For a tutorial on using git, refer to [DigitalCloud's Git Tutorial](https://www.digitalocean.com/community/articles/how-to-use-git-effectively).

## Step 5 - Configure Tomcat Web Management Interface

In order to use the manager webapp installed in Step 3, we must add a login to our Tomcat server. We will do this by editing the tomcat-users.xml file:

* sudo nano /etc/tomcat7/tomcat-users.xml

This file is filled with comments which describe how to configure the file. You may want to delete all the comments between the following two lines, or you may leave them if you want to reference the examples:

tomcat-users.xml excerpt

<tomcat-users>

...

</tomcat-users>

You will want to add a user who can access the manager-gui and admin-gui (the management interface that we installed in Step Three). You can do so by defining a user similar to the example below. Be sure to change the username and password to something secure:

tomcat-users.xml — Admin User

<tomcat-users>

<user username="admin" password="password" roles="manager-gui,admin-gui"/>

</tomcat-users>

Save and quit the tomcat-users.xml file. To put our changes into effect, restart the Tomcat service:

* sudo service tomcat7 restart

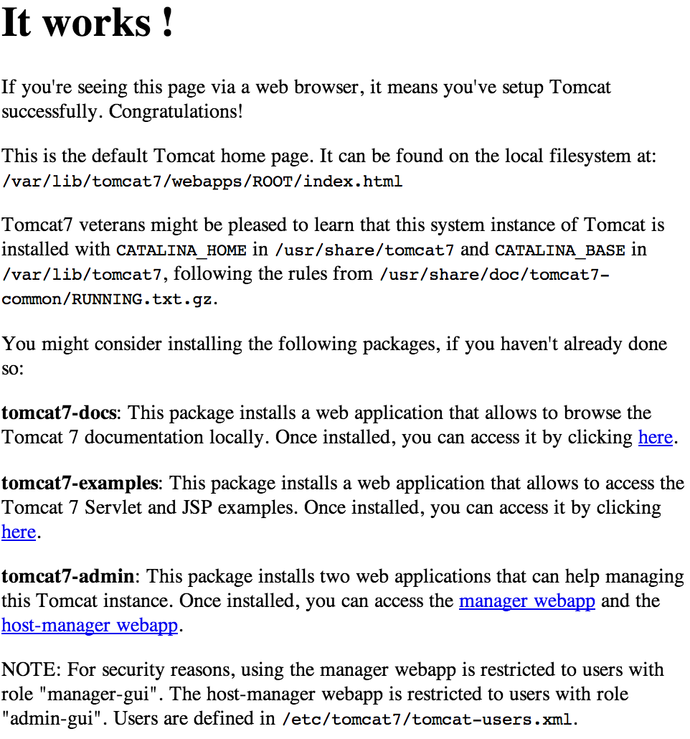
## Step 6 - Access the Web Interface

Now that we've configured an admin user, let's access the web management interface in a web browser:

Open in web browser:

http://server\_IP\_address:8080

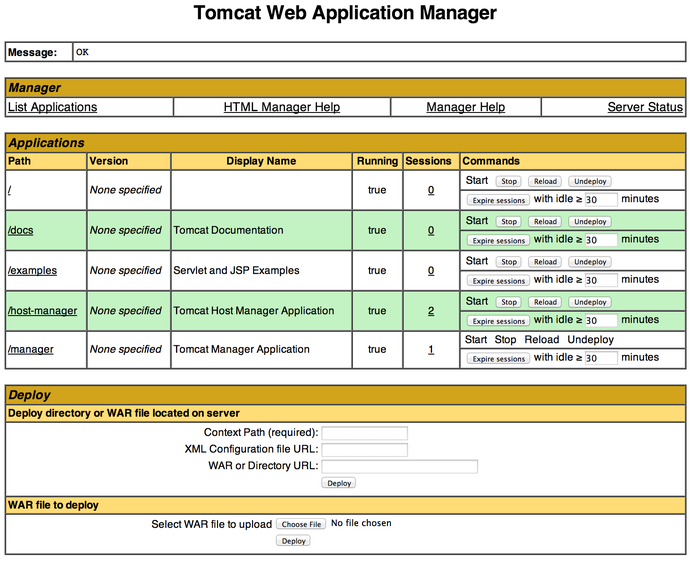
You will see something like the following image:



As you can see, there are four links to packages you installed in Step Three:

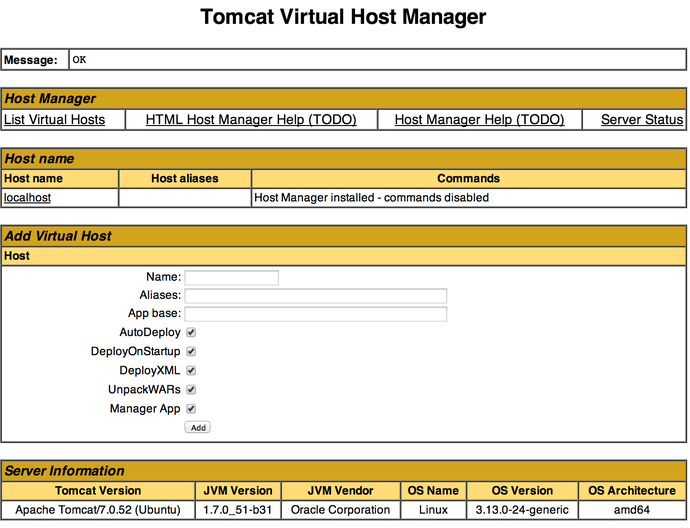
* tomcat7-docs: Online documentation for Tomcat. Accessible via http://server\_IP\_address:8080/docs/
* tomcat7-examples: Tomcat 7 Servlet and JSP examples. You can click through the example webapps to get a basic idea of how they work (and also look at the source code to see how they were implemented). Accessible via http://server\_IP\_address:8080/examples/
* tomcat7-admin (manager-webapp): Tomcat Web Application Manager. This will allow you to manage and your Java applications.
* tomcat7-admin (host-manager): Tomcat Virtual Host Manager.

Let's take a look at the Web Application Manager, accessible via the link or http://server\_IP\_address:8080/manager/html:



The Web Application Manager is used to manage your Java applications. You can Start, Stop, Reload, Deploy, and Undeploy here. You can also run some diagnostics on your apps (i.e. find memory leaks). Lastly, information about your server is available at the very bottom of this page.

Now let's take a look at the Virtual Host Manager, accessible via the link or http://server\_IP\_address:8080/host-manager/html/:



From the Virtual Host Manager page, you can add virtual hosts to serve your applications in.

## Finished!

Your installation of Tomcat is complete! Your are now free to deploy your own webapps!

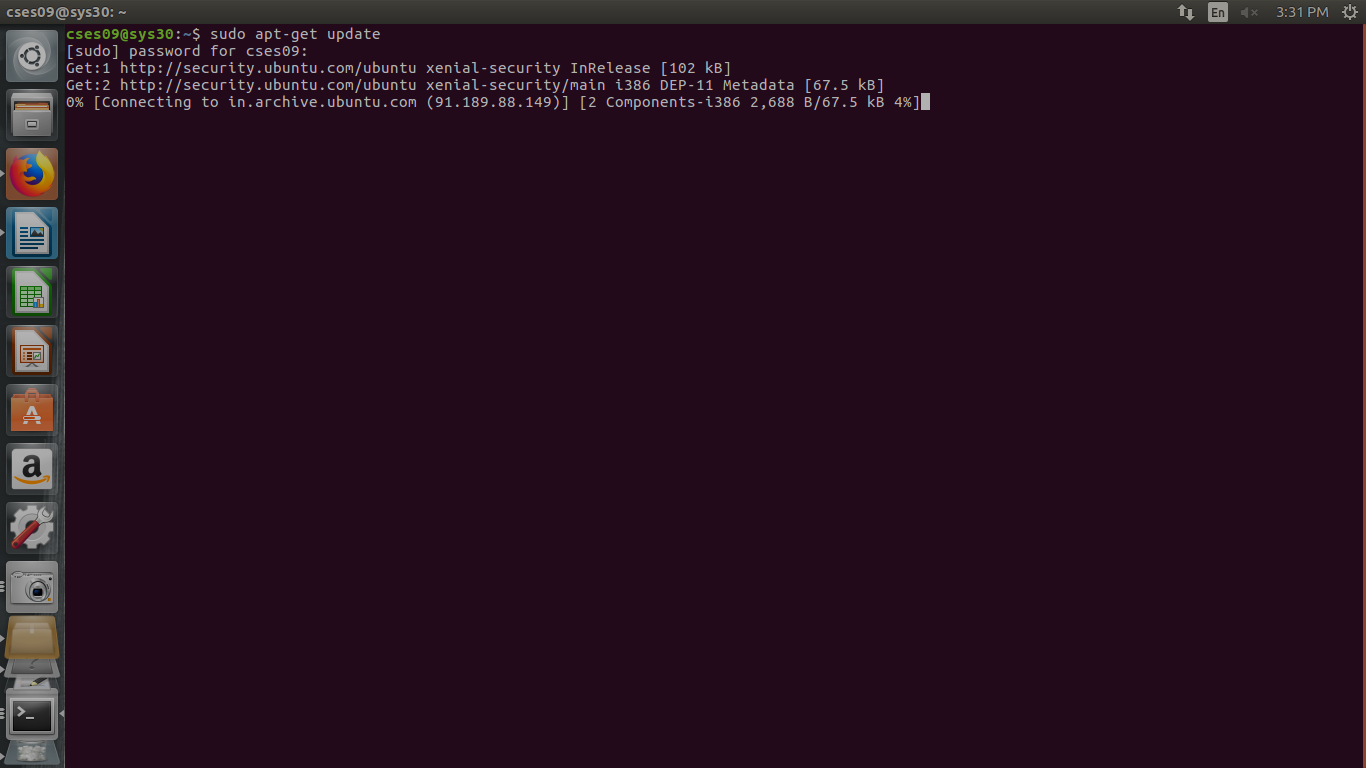
**EXPERIMENT:** **5**

**SQL Installation process:**

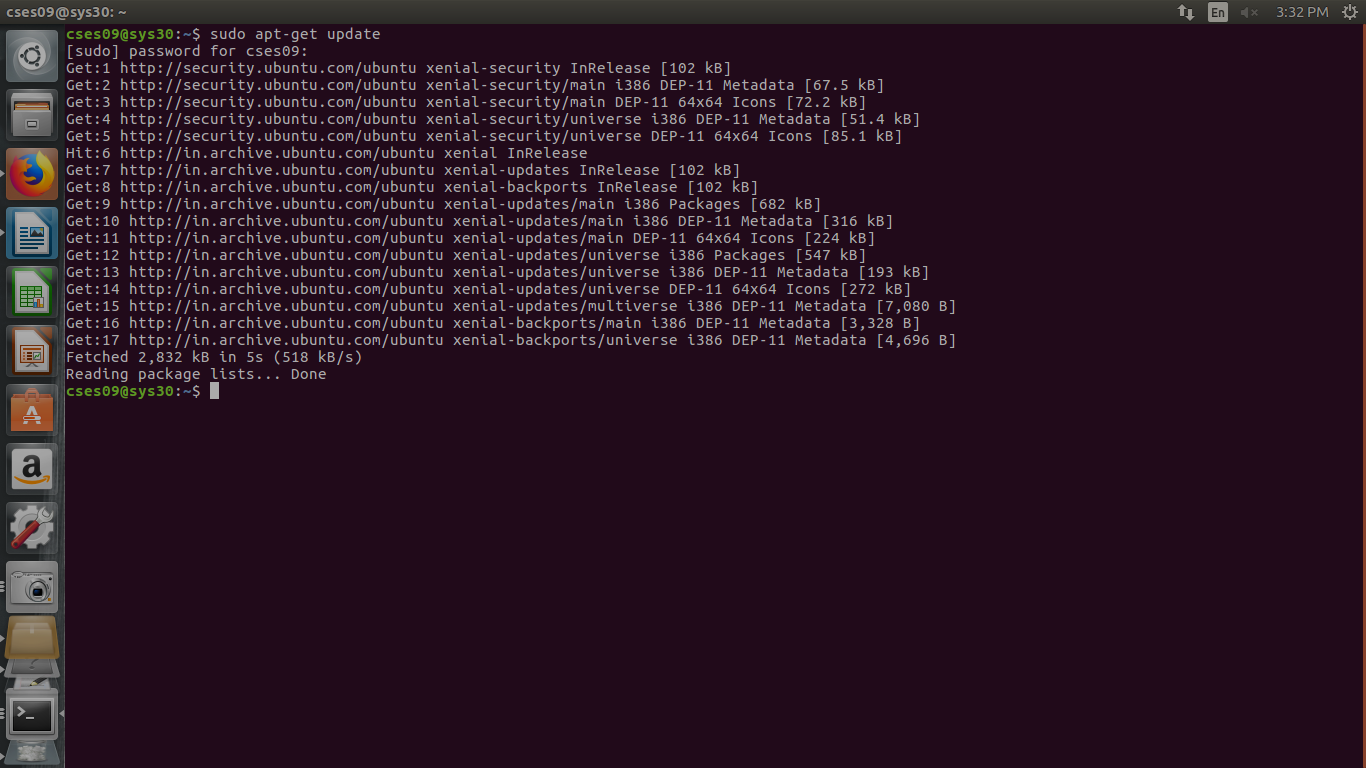
**As shown in the below screenshot we are getting an update from Ubuntu server using apt get update query.**

**And the below given URLs:**

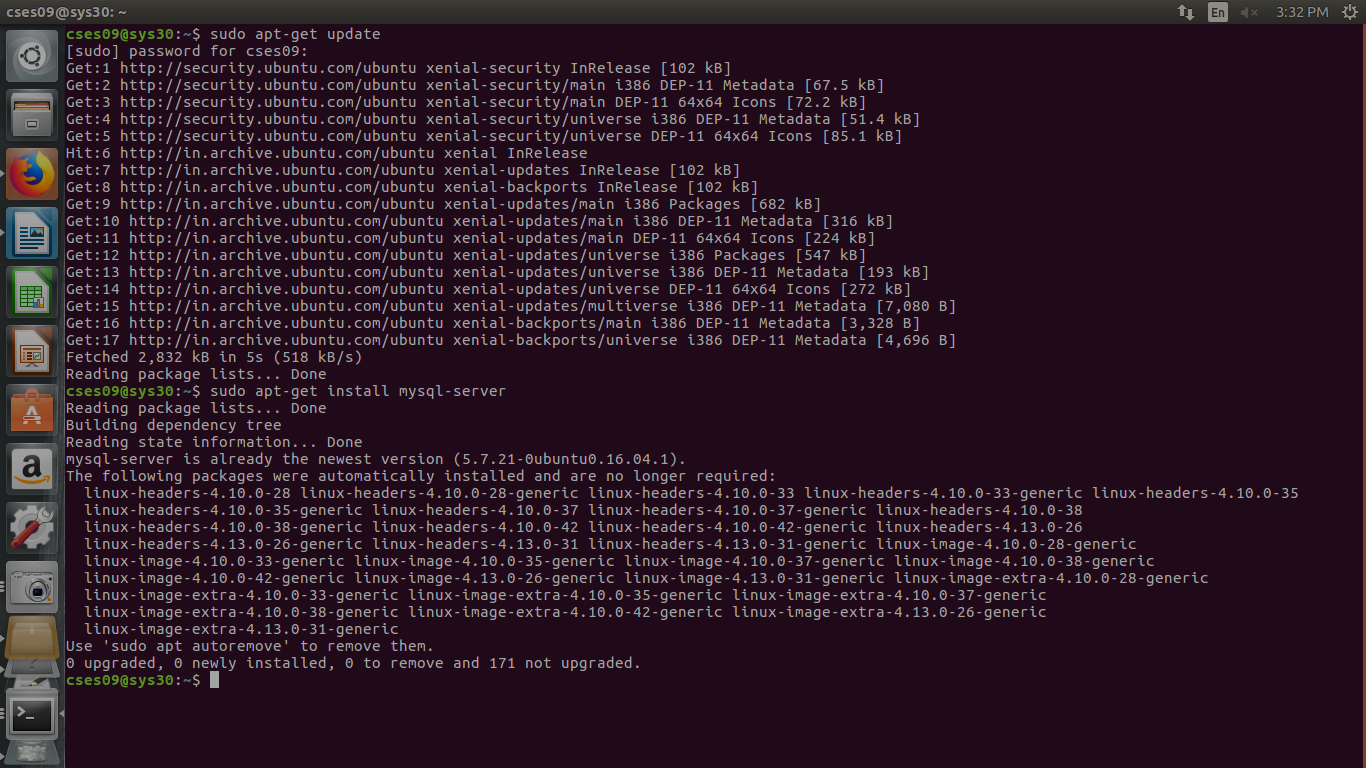
**Once we fetch the installation files from the Ubuntu server .we r going to install the package**

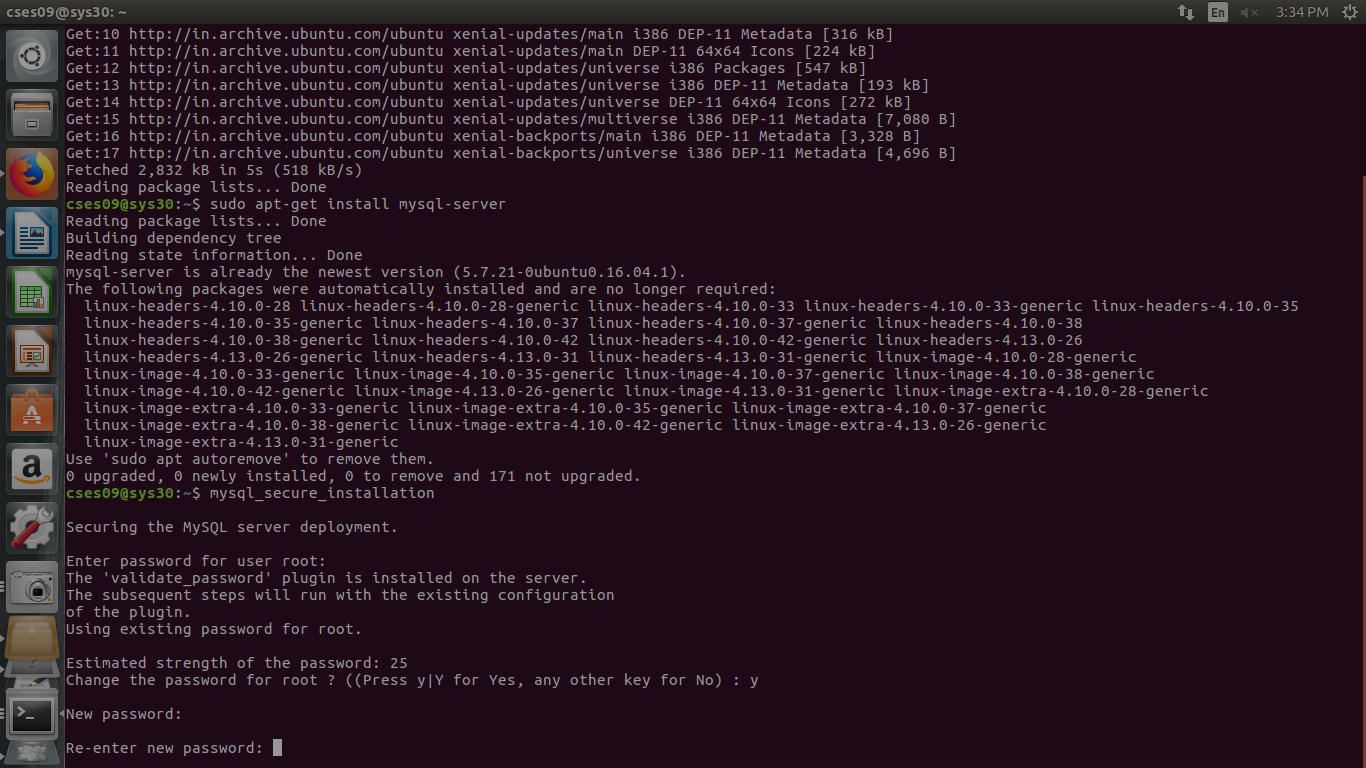


**Packages Installed:**



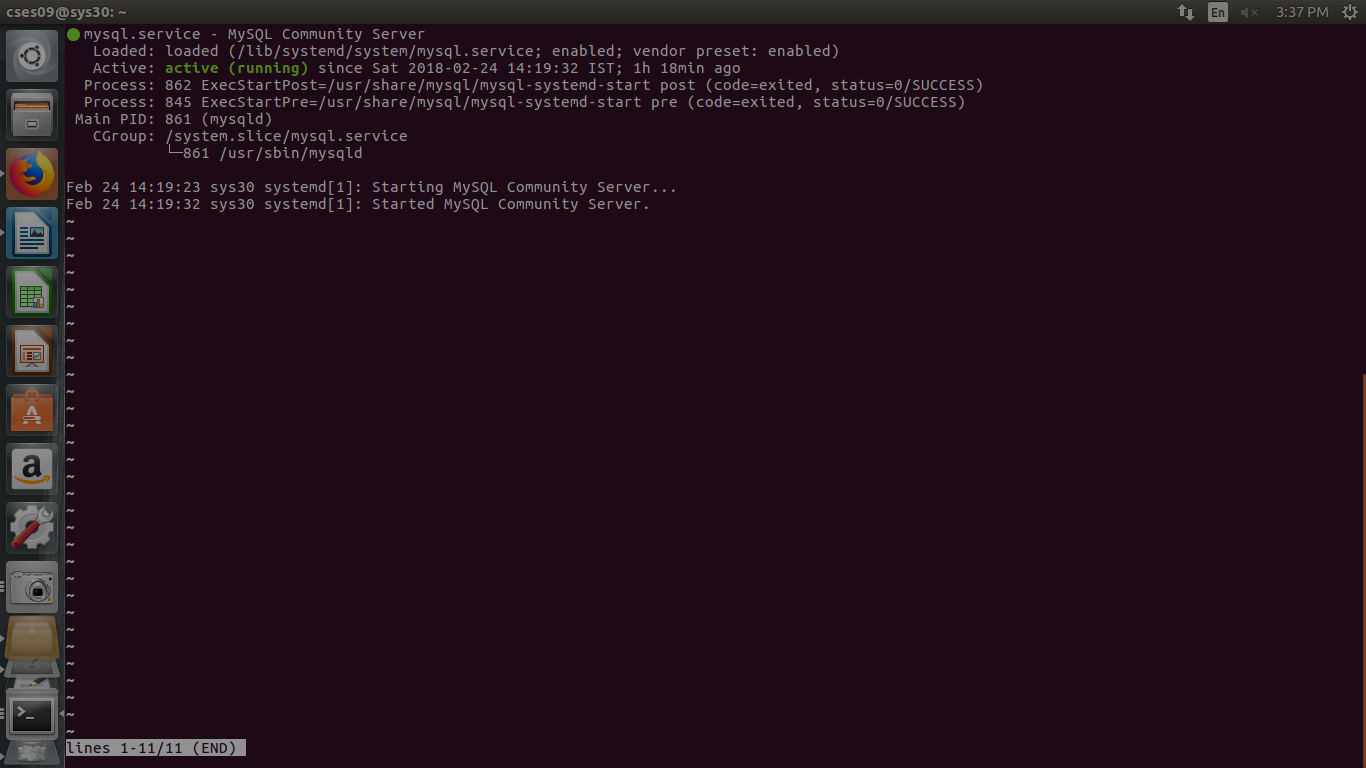
**Once we load /get an update from ubuntu server we r going to install the sql server in the system with the below specified command apt -get sql installation** ...



**Once we install the mysql server packages ensuring a security by providing password**

**mysql\_secure\_installation**

**Below screen shot depicts the my sql server is installed securely, initiating/activating the service for use.**



**EXPERIMENT:** **6**

**Aim: Write an HTML page including any required Javascript that takes a number from one text field in the range of 0 to 999 and shows it in another text field in words.**

* If the number is out of range, it should show “out of range” and
* If it is not a number, it should show “not a number” message in the result box.

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>Untitled Document</title>

</head>

<body>

<input type="text" onKeyDown="numberFunction(this.value)">

<input type="text" id="inwords">

</body>

</html>

<script>

function numberFunction(num)

{

if(isNumber(num) == true)

{

if(num>0 && num<=999 ){

document.getElementById("inwords").value="Valid Number";

}

else{

document.getElementById("inwords").value="Out of range";

}

}

else if(isNumber(num) == false)

{

document.getElementById("inwords").value="Not a number";

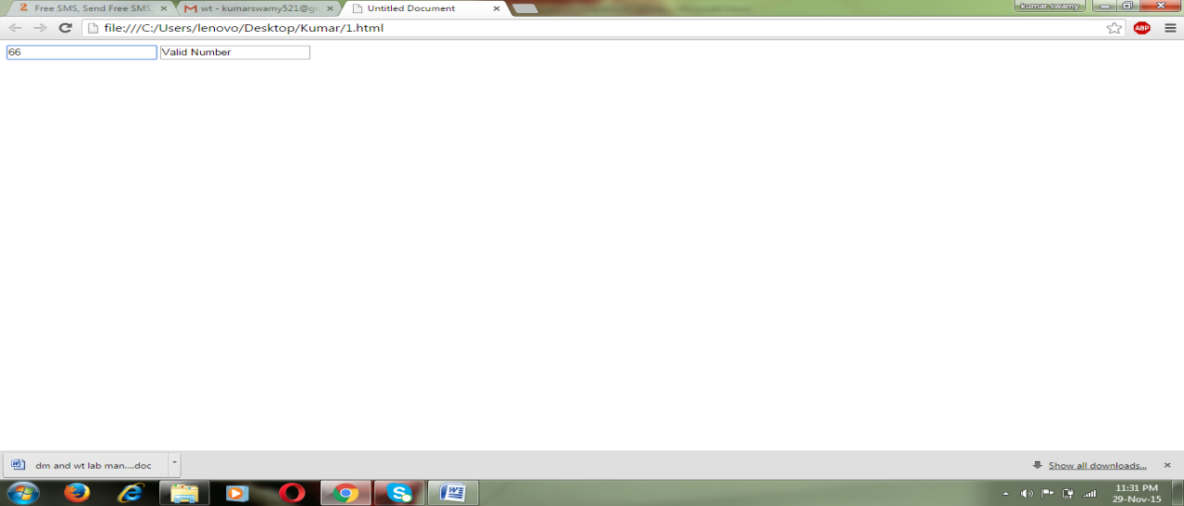
}

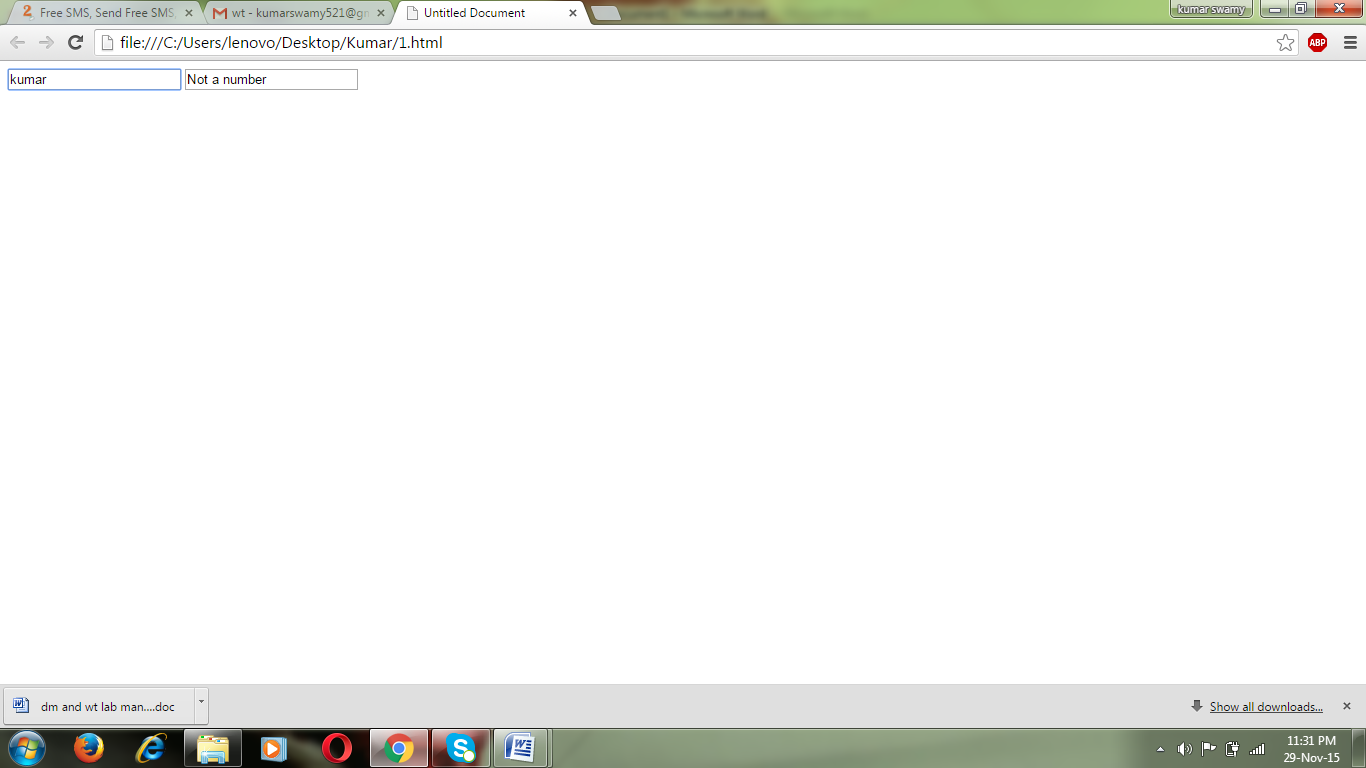
}

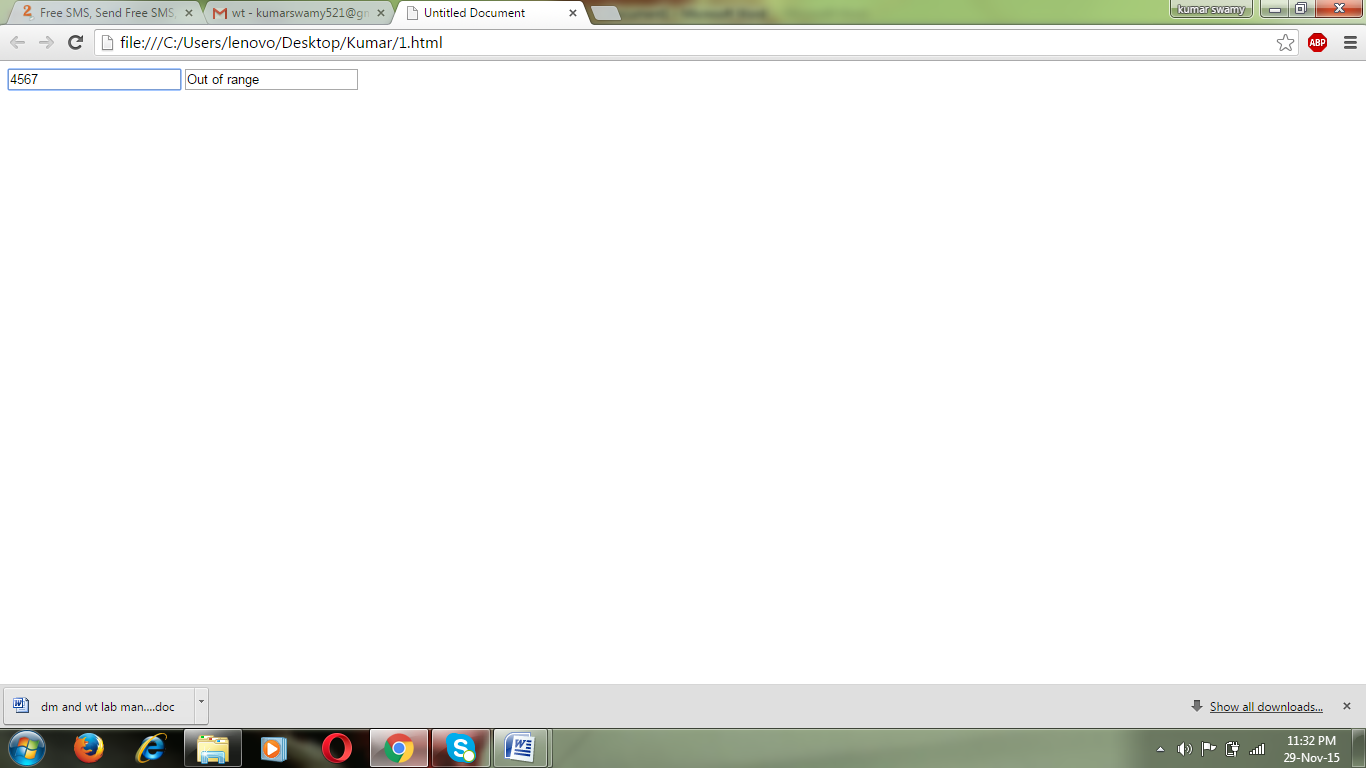
function isNumber(n) { return /^-?[\d.]+(?:e-?\d+)?$/.test(n); }

</script>

**OUTPUT:**







**EXPERIMENT:** **7**

**Aim: Write an HTML page that has one input, which can take multi-line text and a submit button. Once the user clicks the submit button, it should show the number of characters, words and the lines in the text entered using an alert message. Words are separated with white space and lines are separated with new line character.**

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>Untitled Document</title>

</head>

<body>

<textarea name="Text1" cols="40" rows="5" id="inputbox"></textarea><br/>

<button onClick="calc()">Submit</button>

</body>

</html>

<script>

function calc()

{

var inputboxval = document.getElementById("inputbox").value;

//var withoutSpace = inputboxval.replace(/ /g,"");

var Charslength = inputboxval.length;

var wordcount = countWords(inputboxval);

var lines = inputboxval.split("\n").length;

alert("NUmber of characters"+Charslength+"\nnumber of words : "+wordcount+"\nnumber of lines"+lines)

}

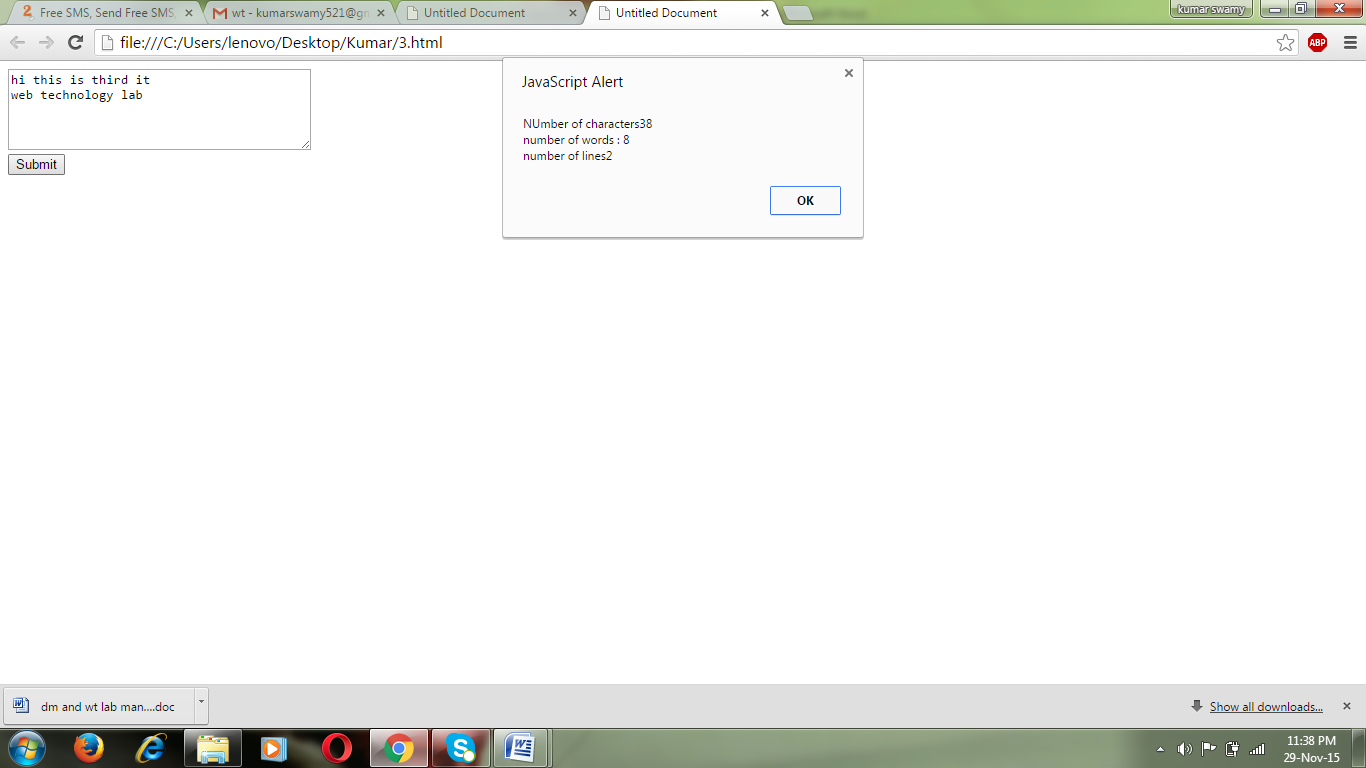
function countWords(str) {

return str.split(/\s+/).length;

}

</script>

**OUTPUT:**

****

**EXPERIMENT:** **8**

**Aim: Write an HTML page that contains a selection box with a list of five countries. When the user selects a country, its capital should be printed next to the list. Add css to customize the properties of the font of the capital (color, bold and font size).**

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>Untitled Document</title>

</head>

<body>

<select id="countries" onChange="capital()">

<option value="Delhi">India</option>

<option value="Canberra">Australia</option>

<option value="Washington-DC">US</option>

<option value="London">UK</option>

<option value="Singapore">Singapore</option>

</select>

<div>

<ul id="list">

</ul>

</div>

</body>

</html>

<script>

function capital()

{

var country = document.getElementById("countries").text;

var capital = document.getElementById("countries").value;

document.getElementById("list").innerHTML=""; // first empty UL

var ul = document.getElementById("list"); // next get element by id

var li = document.createElement("li");//create elemnt to UL.

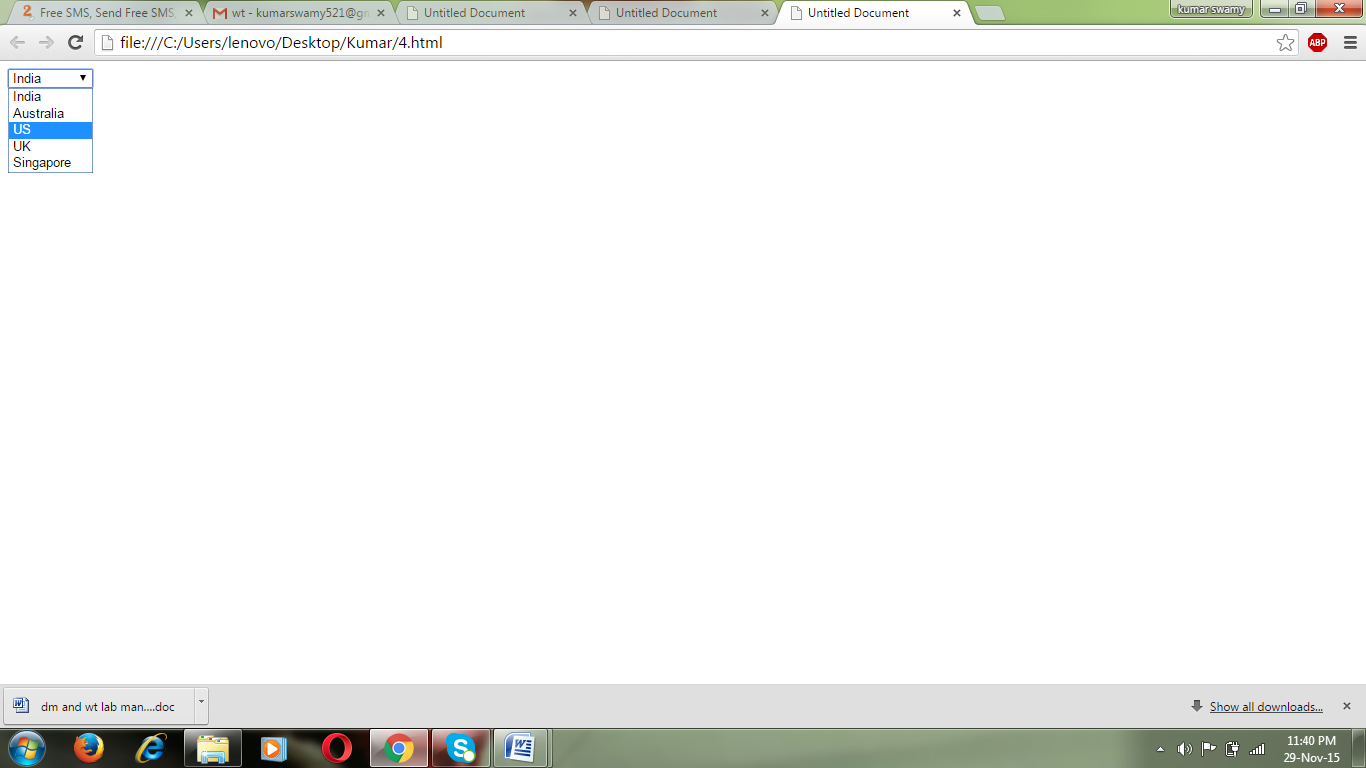
var LI = ul.appendChild(li);// next append li to that element

LI.innerHTML="<p style='color:red;font-size:20px;'>"+capital+"</p>";

}

</script>

**OUTPUT:**

****

**EXPERIMENT:** **9**

**Aim:** A simple calculator web application that takes two numbers and an operator (+, -, /,\* and %) from an HTML page and returns the result page with the operation performed on the operands.

**cal.html**

<html>

<head>

<title>HTML Calculator</title>

</head>

<body bgcolor= "#000000" text= "gold">

<form name="calculator" >

<input type="button" value="1" onClick="document.calculator.ans.value+='1'">

<input type="button" value="2" onClick="document.calculator.ans.value+='2'">

<input type="button" value="3" onClick="document.calculator.ans.value+='3'">

<input type="button" value="+" onClick="document.calculator.ans.value+='+'">

<br>

<input type="button" value="4" onClick="document.calculator.ans.value+='4'">

<input type="button" value="5" onClick="document.calculator.ans.value+='5'">

<input type="button" value="6" onClick="document.calculator.ans.value+='6'">

<input type="button" value="-" onClick="document.calculator.ans.value+='-'">

<br>

<input type="button" value="7" onClick="document.calculator.ans.value+='7'">

<input type="button" value="8" onClick="document.calculator.ans.value+='8'">

<input type="button" value="9" onClick="document.calculator.ans.value+='9'">

<input type="button" value="\*" onClick="document.calculator.ans.value+='\*'">

<br>

<input type="button" value="/" onClick="document.calculator.ans.value+='/'">

<input type="button" value="0" onClick="document.calculator.ans.value+='0'">

<input type="button" value="%" onClick="document.calculator.ans.value+='%'">

<input type="button" value="=" onClick="document.calculator.ans.value=eval(document.calculator.ans.value)">

<br>

<input type="reset" value="Reset">

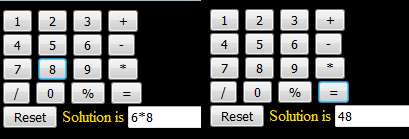
Solution is <input type="textfield" name="ans" value="">

</form>

</body>

</html>

**OUTPUT:**



**EXPERIMENT:** **10**

**Aim: Create an XML document that contains 10 users information. Write a Java program, which takes User Id as input and returns the user details by taking the user information from the XML document using DOM parser.**

**employees.xml**

<employees>

<employee id="111">

<firstName>Naresh</firstName>

<lastName>Gupta</lastName>

<location>India</location>

</employee>

<employee id="222">

<firstName>Kumar</firstName>

<lastName>Gussin</lastName>

<location>Russia</location>

</employee>

<employee id="333">

<firstName>David</firstName>

<lastName>Feezor</lastName>

<location>USA</location>

</employee>

<employee id="444">

<firstName>Lokesh</firstName>

<lastName>Gupta</lastName>

<location>India</location>

</employee>

<employee id="555">

<firstName>Vishnu</firstName>

<lastName>Gussin</lastName>

<location>Russia</location>

</employee>

<employee id="666">

<firstName>Veeru</firstName>

<lastName>Feezor</lastName>

<location>USA</location>

</employee>

<employee id="777">

<firstName>Pavan</firstName>

<lastName>Feezor</lastName>

<location>USA</location>

</employee>

<employee id="888">

<firstName>Narayana</firstName>

<lastName>Gussin</lastName>

<location>Russia</location>

</employee>

<employee id="999">

<firstName>David</firstName>

<lastName>Feezor</lastName>

<location>USA</location>

</employee>

<employee id="1000">

<firstName>Sunder</firstName>

<lastName>Feezor</lastName>

<location>USA</location>

</employee>

</employees>

**ReadXML.java:**

**import** org.w3c.dom.\*;

**import** javax.xml.parsers.\*;

**import** java.io.\*;

**import** java.util.Scanner;

**public** **class** ReadXML {

**public** **static** **void** main(String a[]) **throws** Exception{

DocumentBuilderFactory factory = DocumentBuilderFactory.*newInstance*();

DocumentBuilder builder = factory.newDocumentBuilder();

//Build Document

Document document = builder.parse(**new** File("C:\\Users\\Naresh\\Desktop\\employees.xml"));

//Normalize the XML Structure; It's just too important !!

document.getDocumentElement().normalize();

//Here comes the root node

Element root = document.getDocumentElement();

//Get all employees

NodeList nList = document.getElementsByTagName("employee");

System.*out*.println("enter employee id:");

Scanner s=**new** Scanner(System.*in*);

String id=s.next();

**for** (**int** temp = 0; temp < nList.getLength(); temp++)

{

Node node = nList.item(temp);

**if** (node.getNodeType() == Node.*ELEMENT\_NODE*)

{

Element eElement = (Element) node;

**if**(eElement.getAttribute("id").equals(id)){

System.*out*.println("First Name : " + eElement.getElementsByTagName("firstName").item(0).getTextContent());

System.*out*.println("Last Name : " + eElement.getElementsByTagName("lastName").item(0).getTextContent());

System.*out*.println("Location : " + eElement.getElementsByTagName("location").item(0).getTextContent());

}

}

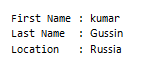
}

}

}

**OUTPUT:-**

Enter Employee Id: 222



**EXPERIMENT:** **11**

HelloWorldServlet.java

import javax.servlet.GenericServlet;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.ServletException;

import java.io.IOException;

import java.io.PrintWriter;

public class HelloWorldServlet extends GenericServlet

{

public void service(ServletRequest request,ServletResponse response)throws ServletException,IOException

{

PrintWriter out=response.getWriter();

out.println("<h1>Welcome to Servlet</h1>");

out.close();

}

}

WEB.XML

<web-app>

<servlet>

<servlet-name>Hello</servlet-name>

<servlet-class>HelloWorldServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>Hello</servlet-name>

<url-pattern>/srv1</url-pattern>

</servlet-mapping>

</web-app>

**EXPERIMENT:** **12**

**Aim: “Implementing the following web application using JSP.”**

**A web application takes a name as input and on submit it shows a hello <name> page where <name> is taken from the request. It shows the start time at the right top corner of the page and provides a logout button. On clicking this button, it should show a logout page with Thank you <name > message with the duration of usage (hint: Use session to store name and time).**

**Login.html:**

<form name = *"login"* method=*"post"* action=*"hello.jsp"*>

<table width=*"370"* border=*"1"* align=*"center"* style="background: *silver*; ">

<tr height=*25* bgcolor=*""*>

<th colspan=*"4"* align=*"center"*>User Login &nbsp;

</th>

</tr>

<tr>

<td>User Name</td>

<td><input type=*"text"* name=*"uname"* ></td>

</tr>

<tr align=*"center"*>

<td align=*"center"* colspan=*"2"*>

<input type=*"submit"* name=*"ok"* value=*"Submit"*>

</tr>

</table>

</form>

**hello.jsp:**

<%@page import=*"java.util.Date"* %>

<%

String uname=request.getParameter("uname");

session.setAttribute("uname",uname);

Date d=**new** Date();

session.setAttribute("time",d.getSeconds());

%>

<h2 align=*"center"*>hello <%=uname %></h2>

<h4 align=*"right"*><%=d %></h4><br>

<a href=*"thanku.jsp"* align=*"center"*>Logout</a>

**Thanku.jsp:**

<%@page import=*"java.util.Date"* %>

<%

String uname=(String)session.getAttribute("uname");

**int** time=(Integer)session.getAttribute("time");

Date d=**new** Date();

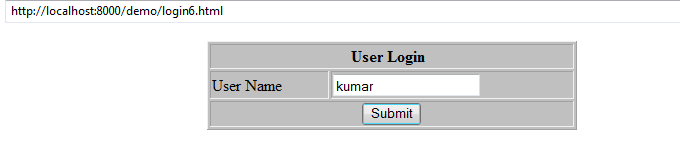
**int** current=d.getSeconds()-time;

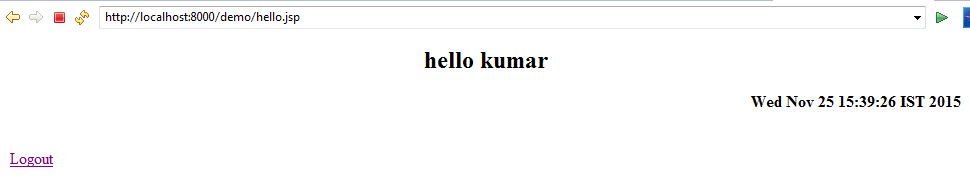
%>

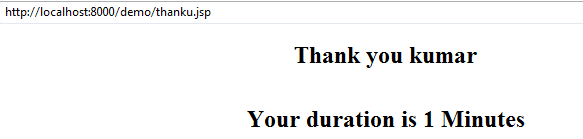
<h2 align=*"center"*>Thank you <%=uname %></h2><br>

<h2 align=*"center"*>Your duration is <%=current %> Seconds</h2>

**OUTPUT:**

****

****

****

**EXPERIMENT:** **13**

**Aim:** A web application that takes name and age from an HTML page. If the age is less than 18, it should send a page with “Hello <name> you are not authorized to visit this site” message, where <name> should be replaced with the entered name. Otherwise it should send “Welcome <name> to this site” message.

**login.html**

<form name = *"login"* method=*"post"* action=*"hello.jsp"*>

<table width=*"370"* border=*"1"* align=*"center"* style="background: *silver*; ">

<tr height=*25* bgcolor=*""*>

<th colspan=*"4"* align=*"center"*>User Login &nbsp;

</th>denial prabhak

</tr>

<tr>

<td>User Name</td>

<td><input type=*"text"* name=*"uname"* ></td>

</tr>

<tr>

<td>Age</td>

<td><input type=*"text"* name=*"age"* ></td>

</tr>

<tr align=*"center"*>

<td align=*"center"* colspan=*"2"*>

<input type=*"submit"* name=*"ok"* value=*"Submit"*>

</tr>

</table>

</form>

**hello.jsp:**

<%

String uname=request.getParameter("uname");

session.setAttribute("uname",uname);

**int** age=Integer.parseInt(request.getParameter("age"));

**if**(age<18){

out.println("<h4 align='center'>Hello\t" +uname+", \t you are not authorised to visit this site</h4>");

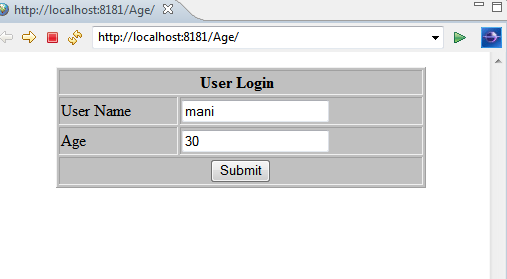
}**else**{

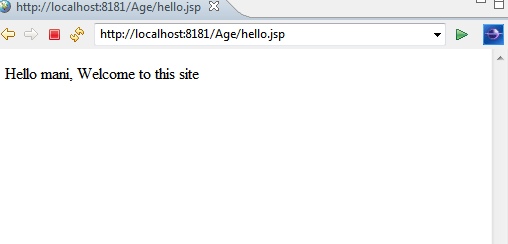
out.println("Hello\t" +uname+", \t Welcome to this site");

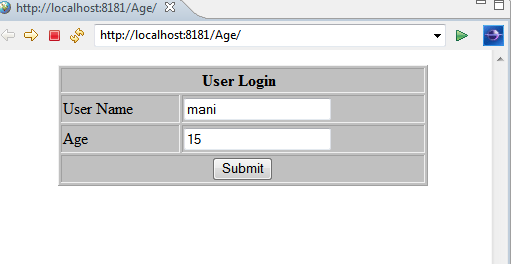
}

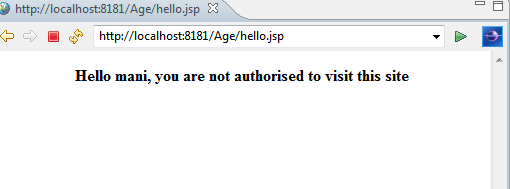
%>

**OUTPUT:**

****

****

****

****

**EXPERIMENT:** **14**

**Aim:** A web application that lists all cookies stored in the browser on clicking “List Cookies” button. Add cookies if necessary.

**listcookie.jsp:**

<%

// Create cookies for first and last names.

Cookie firstName = **new** Cookie("first\_name", "naresh");

Cookie lastName = **new** Cookie("last\_name", "koenni");

// Set expiry date after 24 Hrs for both the cookies.

firstName.setMaxAge(60\*60\*24);

lastName.setMaxAge(60\*60\*24);

// Add both the cookies in the response header.

response.addCookie( firstName );

response.addCookie( lastName );

//read cookies

Cookie cookie = **null**;

Cookie[] cookies = **null**;

// Get an array of Cookies associated with this domain

cookies = request.getCookies();

**if**( cookies != **null** ){

out.println("<h2> Found Cookies Name and Value</h2>");

**f**or (**int** i = 0; i < cookies.length; i++){

cookie = cookies[i];

out.print("Name : " + cookie.getName( ) + ", ");

out.print("Value: " + cookie.getValue( )+" <br/>");

}

}**else**{

out.println("<h2>No cookies founds</h2>");

}

%>

**OUTPUT:**



**EXPERIMENT:** **15**

**Aim: “Implementing the following web application using Servlets.”**

A user validation web application, where the user submits the login name and password to the server. The name and password are checked against the data already available in Database and if the data matches, a successful login page is returned. Otherwise a failure message is shown to the user.

**login.html**

<form name = *"login"* method=*"post"* action=*"sls"*>

<table width=*"370"* border=*"1"* align=*"center"* style="background: *silver*; ">

<tr height=*25* >

<th colspan=*"4"* align=*"center"*>User Login &nbsp;

</th>

</tr>

<tr>

<td>User Name</td>

<td><input type=*"text"* name=*"uname"* ></td>

</tr>

<tr>

<td>Password</td>

<td><input type=*"password"* name=*"pwd"* ></td>

</tr>

<tr align=*"center"*>

<td align=*"center"* colspan=*"2"*>

<input type=*"submit"* name=*"ok"* value=*"Submit"*>

</tr>

</table>

</form>

**StudentLoginSevlet.java:**

**package** pres;

**import** java.io.IOException;

**import** java.io.PrintWriter;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.sql.Statement;

**import** javax.servlet.\*;

**import** javax.servlet.http.HttpServlet;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**public** **class** StudentLoginServlet **extends** HttpServlet {

**private** Connection con=**null**;

**public** **void** init() **throws** ServletException {

**try** {

DriverManager.*registerDriver*(**new** com.mysql.jdbc.Driver());

con = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/mrecw","root","naresh");

System.*out*.println("got database connection");

} **catch** (Exception e) {

System.*out*.println("not able to database connection");

e.printStackTrace();

}

}

**public** **void** service(HttpServletRequest req, HttpServletResponse resp)

**throws** ServletException, IOException {

resp.setContentType("text/html");

Statement statement=**null**;

ResultSet resultSet = **null**;

String uname=req.getParameter("uname");

String pwd=req.getParameter("pwd");

**try** {

PrintWriter out=resp.getWriter();

statement = con.createStatement();

ResultSet rs=statement.executeQuery("select \* from login where uname='"+uname+"' and pwd='"+pwd+"' ");

**if**(rs.next()){

RequestDispatcher rd=req.getRequestDispatcher("home.jsp");

rd.forward(req,resp);

}**else**{

RequestDispatcher rd=req.getRequestDispatcher("login.html");

rd.include(req,resp);

out.println("<h3 align='center' style=\"color:red;\">wrong credentials</h3>");

}

} **catch** (SQLException e) {

e.printStackTrace();

System.*out*.println("not able to get the connection");

}**finally**{

**try**{

//con.close();

System.*out*.println("connection closed");

}

**catch**(Exception e){}

} }

}

**home.jsp:**

<h2>welcome to home page- You are successfully logged in</h2><br/>

<a href=*"login.html"*>Back</a>

**Web.xml**

<web-app>

<welcome-file-list>

<welcome-file>index.html</welcome-file>

</welcome-file-list>

<servlet>

<servlet-name>sls</servlet-name>

<servlet-class>pres.StudentLoginServlet</servlet-class>

</servlet>

<servlet-mapping>

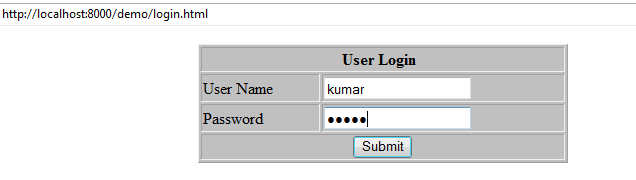
<servlet-name>sls</servlet-name>

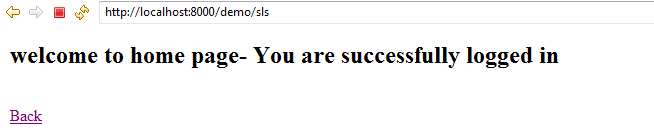
<url-pattern>/sls</url-pattern>

</servlet-mapping>

</web-app>

**OUTPUT:-**





**EXPERIMENT:** **16**

**Aim: Modify the above program to use an xml file instead of database.**

**login.html**

<form name = *"login"* method=*"post"* action=*"sls"*>

<table width=*"370"* border=*"1"* align=*"center"* style="background: *silver*; ">

<tr height=*25* bgcolor=*""*>

<th colspan=*"4"* align=*"center"*>User Login &nbsp;

</th>

</tr>

<tr>

<td>User Name</td>

<td><input type=*"text"* name=*"uname"* ></td>

</tr>

<tr>

<td>Password</td>

<td><input type=*"password"* name=*"pwd"* ></td>

</tr>

<tr align=*"center"*>

<td align=*"center"* colspan=*"2"*>

<input type=*"submit"* name=*"ok"* value=*"Submit"*>

</tr>

</table>

</form>

**StudentLoginSevlet.java:**

**package** pres;

**import** java.io.IOException;

**import** java.io.PrintWriter;

**import** javax.servlet.\*;

**import** javax.servlet.http.HttpServlet;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**public** **class** StudentLoginServlet **extends** HttpServlet {

**public** **void** service(HttpServletRequest req, HttpServletResponse resp)

**throws** ServletException, IOException {

resp.setContentType("text/html");

PrintWriter out=resp.getWriter();

ServletConfig config=getServletConfig();

String un=req.getParameter("uname");

String pw=req.getParameter("pwd");

String uname=config.getInitParameter("uname");

String pwd=config.getInitParameter("pwd");

**if**(uname.equals(uname) && pw.equals(pwd)){

RequestDispatcher rd=req.getRequestDispatcher("home.jsp");

rd.forward(req,resp);

}**else**{

RequestDispatcher rd1=req.getRequestDispatcher("login.html");

rd1.include(req,resp);

out.println("<h3 align='center' style=\"color:red;\">wrong credentials</h3>");

}

}

}

**home.jsp:**

<h2>welcome to home page- You are successfully logged in</h2><br/>

<a href=*"login.html"*>Back</a>

**Web.xml**

<web-app>

<servlet>

<servlet-name>sls</servlet-name>

<servlet-class>pres.StudentLoginServlet</servlet-class>

<init-param>

<param-name>uname</param-name>

<param-value>kumar</param-value>

</init-param>

<init-param>

<param-name>pwd</param-name>

<param-value>kumar</param-value>

</init-param>

</servlet>

<servlet-mapping>

<servlet-name>sls</servlet-name>

<url-pattern>/sls</url-pattern>

</servlet-mapping>

</web-app>

**OUTPUT:-**

