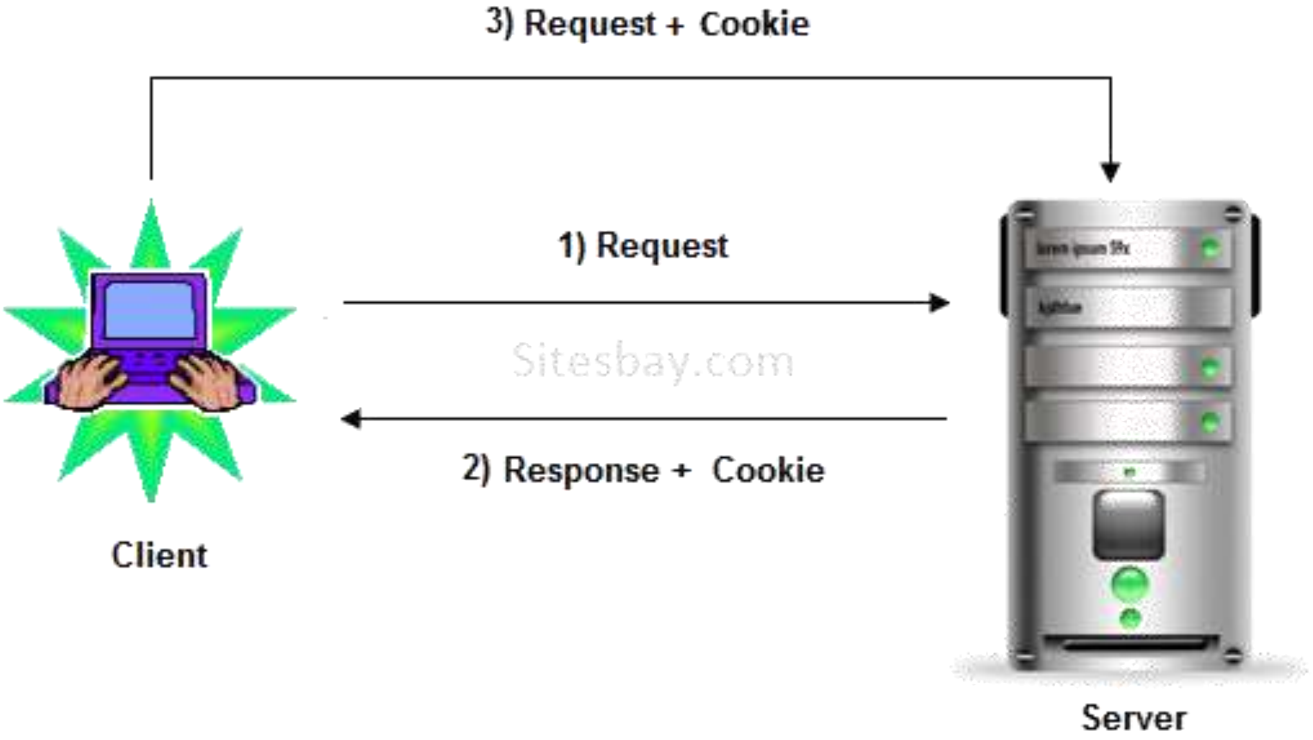
Cookies

Cookies

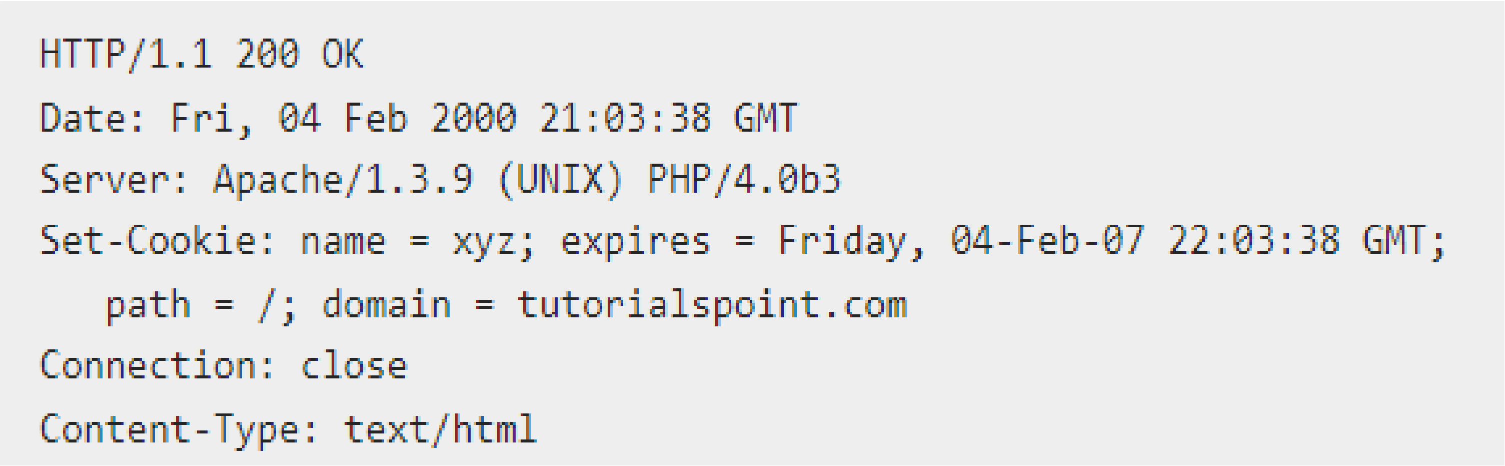
- Cookies are small piece of information sent by web server in response header and gets stored in browser side. A web server can assign a unique session ID to each web client. The cookies are used to maintain the session. The client can disable the cookies.



Introduction

- **Cookies** are small piece of data on the client computer that send response from the web server to client.
- They are used to store the client state.
- The information which is stored on the client machine is called **cookie.**
- A Servlet container sends small information to the web browser. This data is saved by the browser and later back to the server.
- The servlet sends cookies to the browser using **HttpServletResponse.addCookie(javax.servlet.http.Cookie)** method.

- Cookies are usually set in an HTTP header (although JavaScript can also set a cookie directly on a browser). A servlet that sets a cookie might send headers that look something like this –



```
HTTP/1.1 200 OK
Date: Fri, 04 Feb 2000 21:03:38 GMT
Server: Apache/1.3.9 (UNIX) PHP/4.0b3
Set-Cookie: name = xyz; expires = Friday, 04-Feb-07 22:03:38 GMT;
   path = /; domain = tutorialspoint.com
Connection: close
Content-Type: text/html
```

Types of Cookie

There are 2 types of cookies in servlets.

1. Non-persistent cookie
2. Persistent cookie

1. Non-persistent cookie

It is **valid for single session** only. It is removed each time when user closes the browser.

2. Persistent cookie

It is **valid for multiple session** . It is not removed each time when user closes the browser. It is removed only if user logout or signout.

Advantage of Cookies

- Simplest technique of maintaining the state.
- Cookies are maintained at client side.

Disadvantage of Cookies

- It will not work if cookie is disabled from the browser.
- Only textual information can be set in Cookie object.

Note: Gmail uses cookie technique for login. If you disable the cookie, gmail won't work.

Cookie class

javax.servlet.http.Cookie class provides the functionality of using cookies. It provides a lot of useful methods for cookies.

Constructor of Cookie class:

Constructor	Description
Cookie()	Used for constructs a cookie.
Cookie(String name, String value)	Used for constructs a cookie with a specified name and value.

Commonly used methods of the Cookie class

Methods	Description
public void setMaxAge(int expiry)	It is used for Sets the maximum age of the cookie in seconds.
public String getName()	It is used for Returns the name of the cookie. The name cannot be changed after creation.
public String getValue()	It is used for Returns the value of the cookie.
public void setName(String name)	It is used for changes the name of the cookie.
public void setValue(String value)	It is used for changes the value of the cookie.
public void addCookie(Cookie ck)	It is method of HttpServletResponse interface which is used to add cookie in response object.
public Cookie[] getCookies()	It is method of HttpServletRequest interface which is used to return all the cookies from the browser.

- **Create Cookies**

  To create cookies you need to use Cookie class of javax.servlet.http package.

  Cookie ck=new Cookie(name, value);

  //creating cookie object here name and value are string type

- **Add Cookies**

  To add a cookie to the response object, we use addCookie() mehtod.

  response.addCookie(ck); //adding cookie in the response

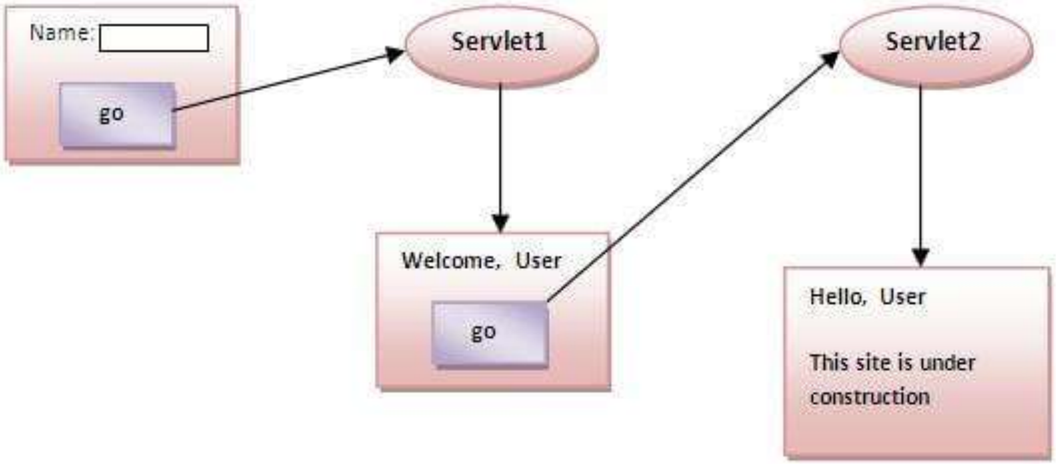
- **Read Cookies for browser**

  To read Cookies from browser to a servlet, we need to call getCookies methods given by request object and it returns an array type of cookie class.

  Cookie c[]=request.getCookie();

Servlet Cookie example

- we are storing the name of the user in the cookie object and accessing it in another servlet. As we know well that session corresponds to the particular user. So if you access it from too many browsers with different values, you will get the different value.



Index.html:

```
<form action="servlet1" method="post">
Name:<input type="text" name="userName"/><br/>
<input type="submit" value="go"/>
</form>
```

FirstServlet.java

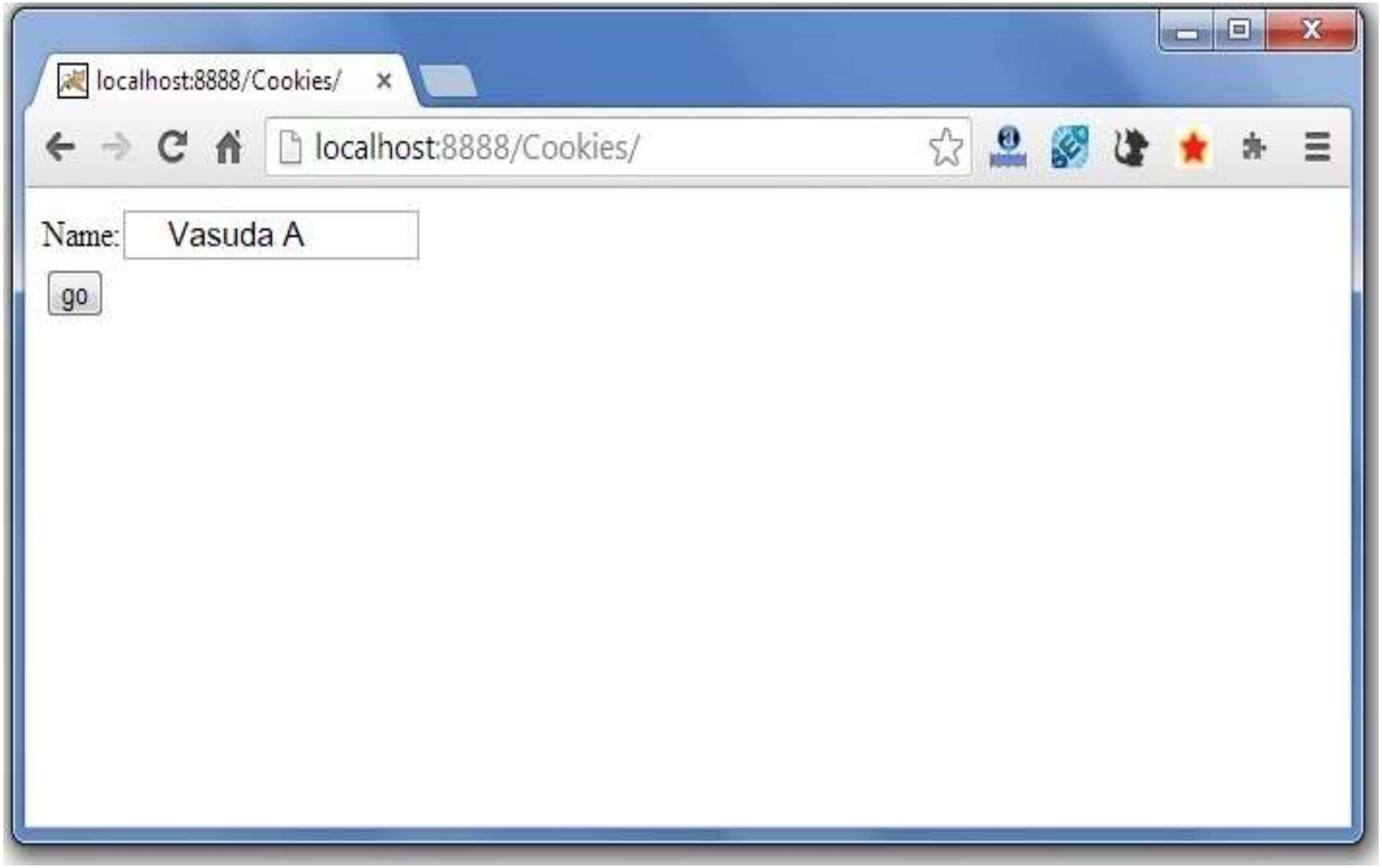
```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;


public class FirstServlet extends HttpServlet {

 public void doPost(HttpServletRequest request, HttpServletResponse response){
  try{

  response.setContentType("text/html");
  PrintWriter out = response.getWriter();

  String n=request.getParameter("userName");
  out.print("Welcome "+n);

  Cookie ck=new Cookie("uname",n);//creating cookie object
  response.addCookie(ck);//adding cookie in the response

  //creating submit button
  out.print("<form action='servlet2'>");
  out.print("<input type='submit' value='go'>");
  out.print("</form>");

  out.close();

    }catch(Exception e){System.out.println(e);}
 }
}
```

SecondServlet.java

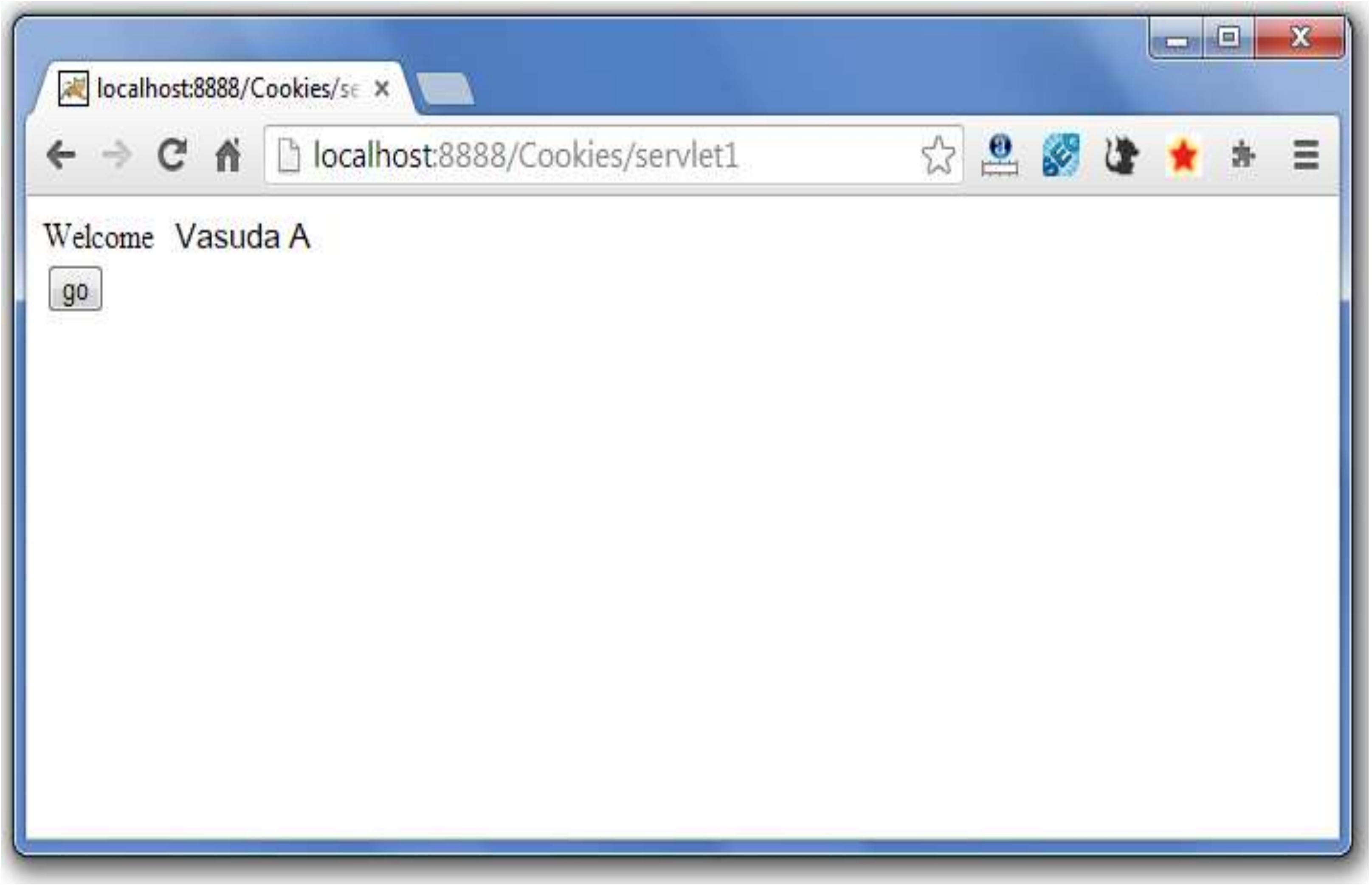
```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class SecondServlet extends HttpServlet {

public void doPost(HttpServletRequest request, HttpServletResponse response){
    try{

    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    Cookie ck[]=request.getCookies();
    out.print("Hello "+ck[0].getValue());

    out.close();

         }catch(Exception e){System.out.println(e);}
    }


}
```

web.xml

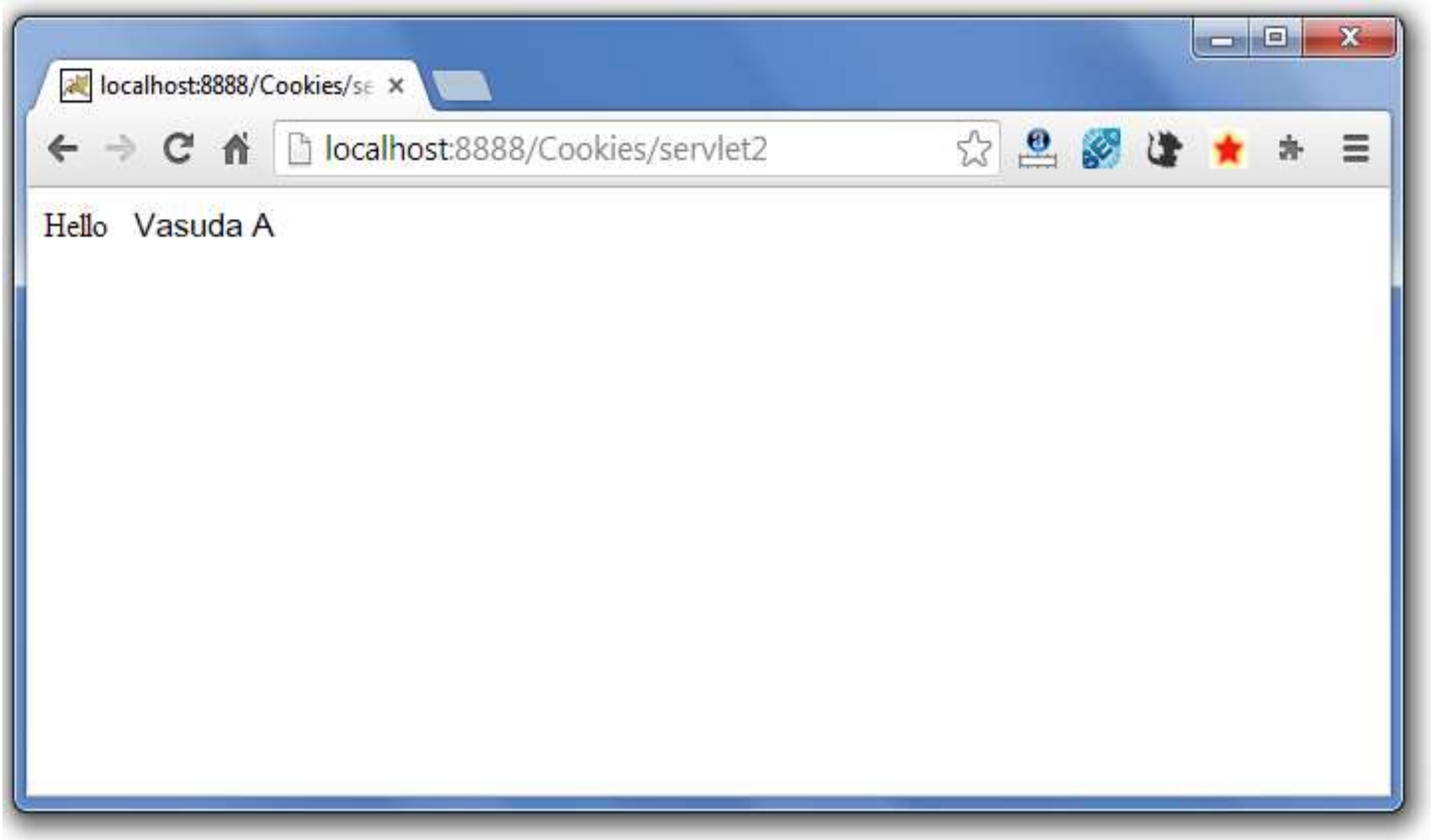
```
<web-app>

<servlet>
<servlet-name>s1</servlet-name>
<servlet-class>FirstServlet</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>SecondServlet</servlet-class>
</servlet>

<servlet-mapping>
<servlet-name>s2</servlet-name>
<url-pattern>/servlet2</url-pattern>
</servlet-mapping>

</web-app>
```



Name: Vasuda A



Welcome Vasuda A



Hello Vasuda A