AJAX

Index.html

```
<html>
<head>
<script>
var request;
function sendInfo()
{
   var v=document.vinform.t1.value;
   var url="index.jsp?val="+v;

   request=new XMLHttpRequest();
  try
   {
     request.onreadystatechange=getInfo;
     request.open("GET",url,true);
     request.send();
   }
   catch(e)
   {
     alert("Unable to connect to server");      } }
```

```
function getInfo(){
    if(request.readyState==4){
      var val=request.responseText;
      document.getElementById('fv').innerHTML=val;
     }
}

</script>
</head>
<body>
   <marquee><h1>This is an example of ajax</h1></marquee>
<form name="vinform">
<input type="text" name="t1">
<input type="button" value="ShowTable" onClick="sendInfo()">
</form>

<span id="fv"> </span>

</body>
</html>
```

```
<html>
<head>
<script>
var request;
function sendInfo()
{
   var v=document.vinform.t1.value;
   var url="index.jsp?val="+v;

request=new XMLHttpRequest();
try
    {
      request.onreadystatechange=getInfo;
      request.open("GET",url,true);
      request.send();
    }
   catch(e)
    {
      alert("Unable to connect to server");
    }
}
```

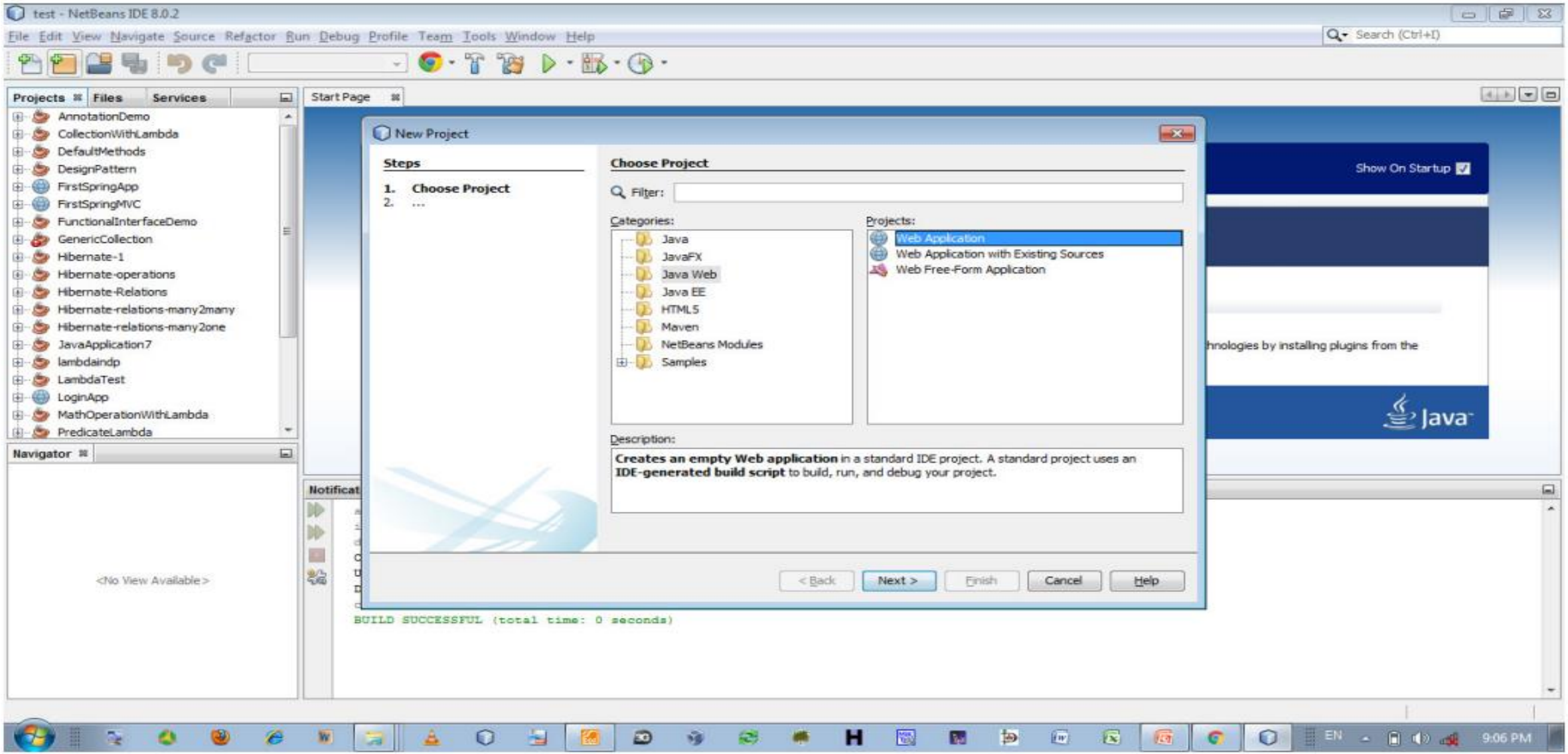
```
function getInfo(){
    if(request.readyState==4){
     var val=request.responseText;
     document.getElementById('fv').innerHTML=val;
     }
}

</script>
</head>
<body>
   <marquee><h1>This is an example of ajax</h1></marquee>
<form name="vinform">
<input type="text" name="t1">
<input type="button" value="ShowTable" onClick="sendInfo()">
</form>

<span id="fv"> </span>

</body>
</html>
```

Index.jsp

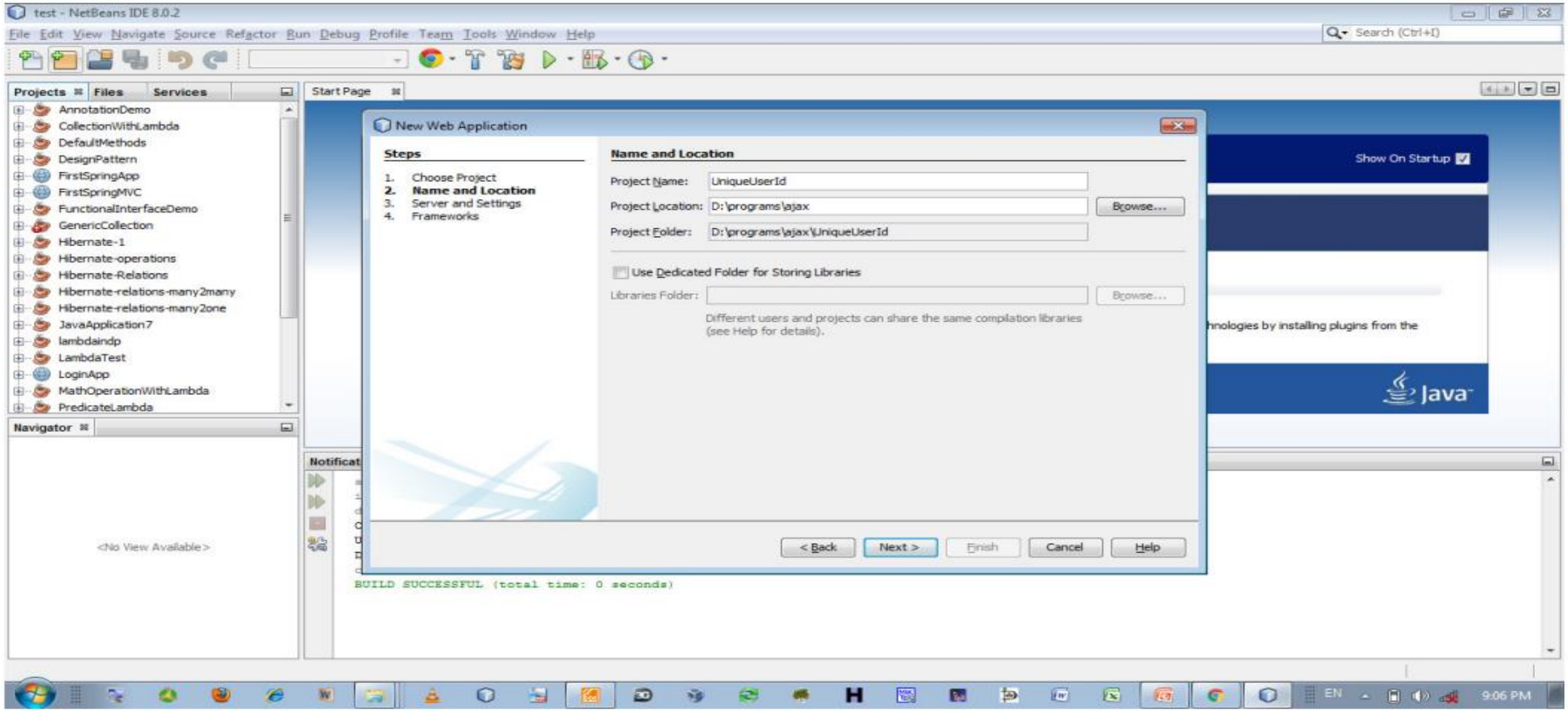
```
<%
int n=Integer.parseInt(request.getParameter("val"));

for(int i=1;i<=10;i++)
out.print(i*n+"<br>");

%>
```

AJAX Application: Check unique user id

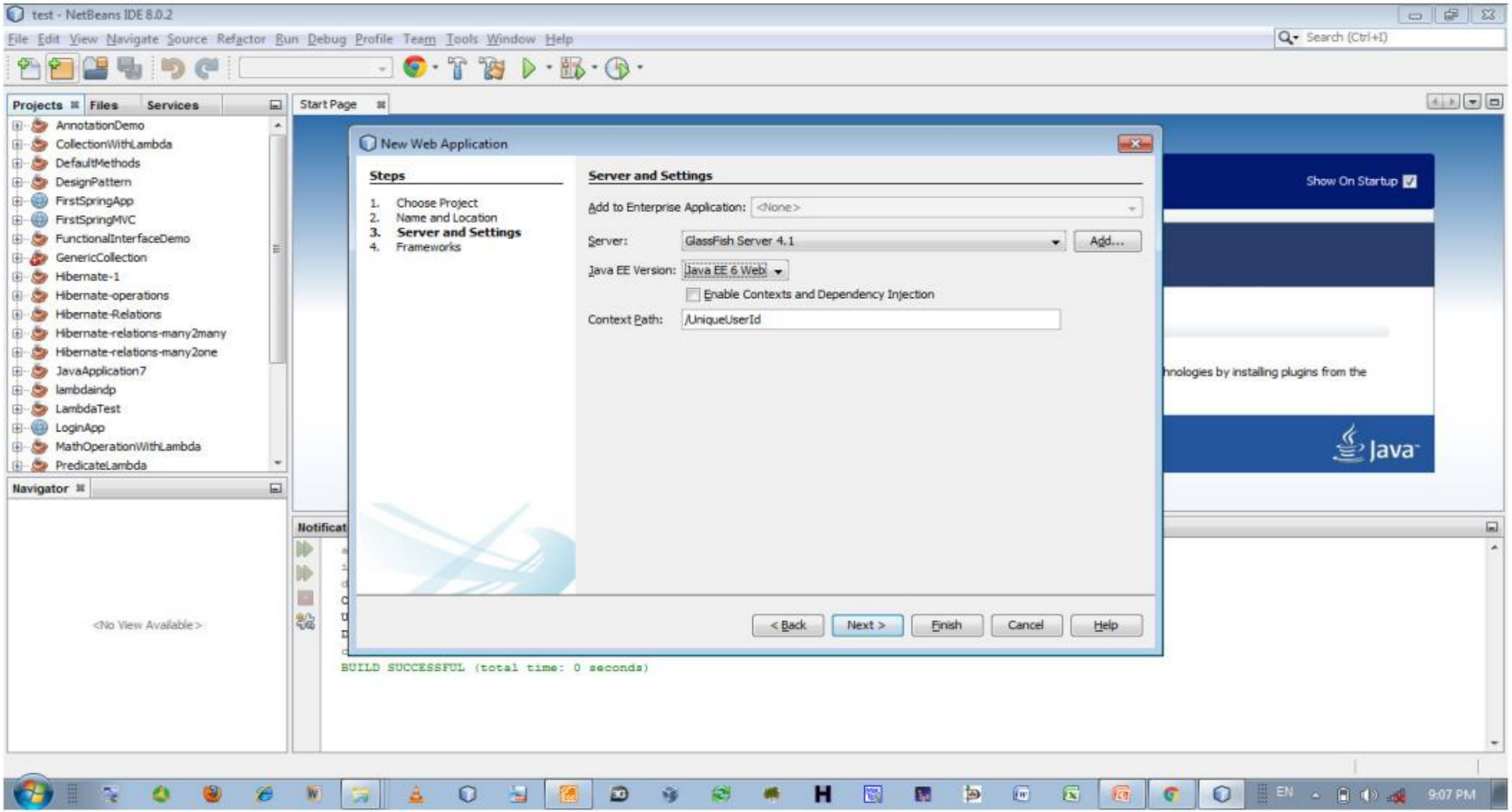
Create Java Web project



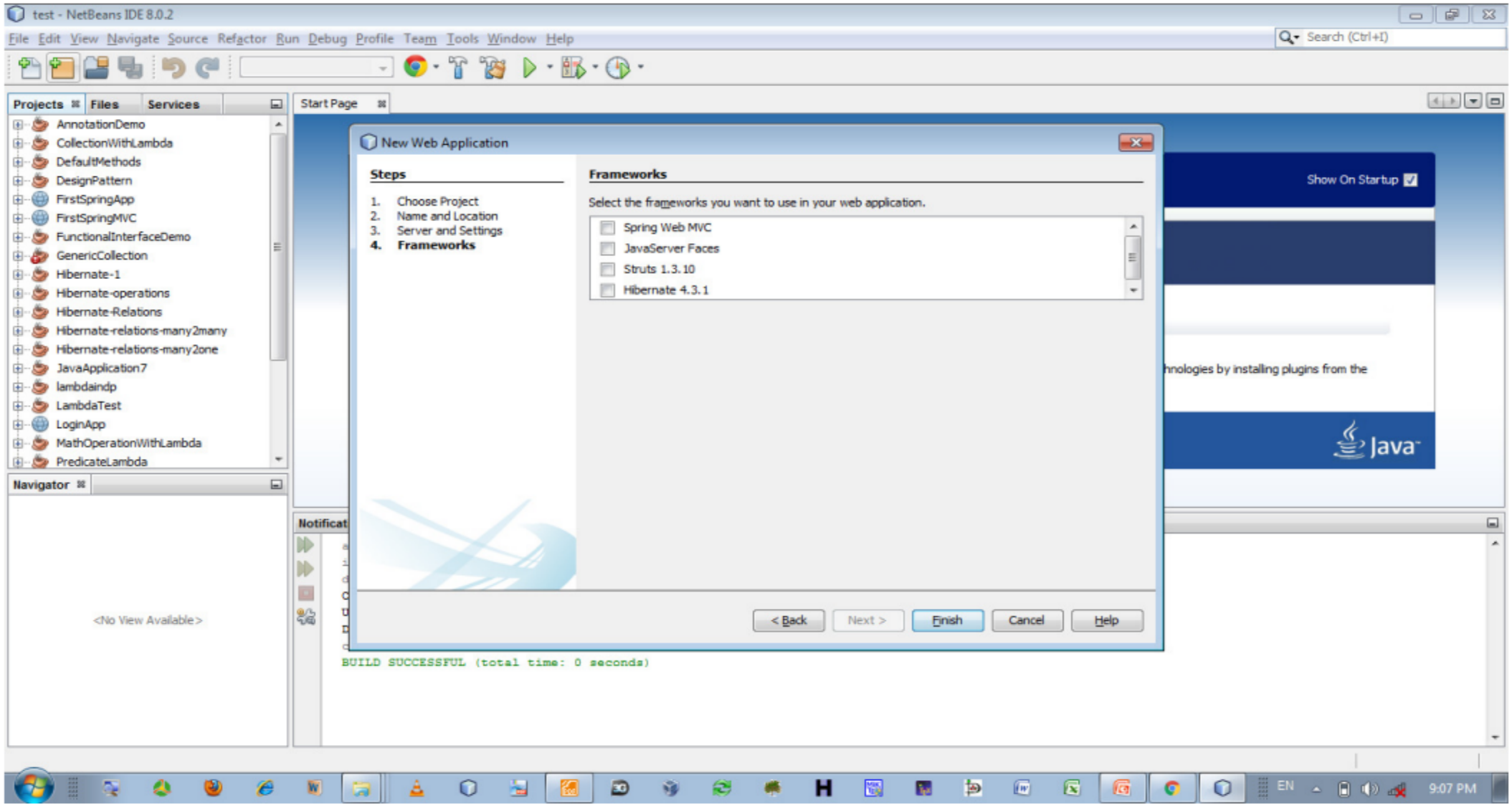
Provide name and path



Select server and code version



Click Finish



- We want to create the following form using HTML code

Programming for Client side

- Create html form
- Write JavaScript

index.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Registration Page</title>
   </head>
   <body>
      <h1>Kindly fill up details for your registration</h1>
      <form name="registrationForm" action="register">
         <table border="0" cellpadding="5">
            <tr>
              <td>User Id:</td>
              <td><input type="textfield" name="userid"/></td>
              <td><div id="status"></div></td>
            </tr>
```

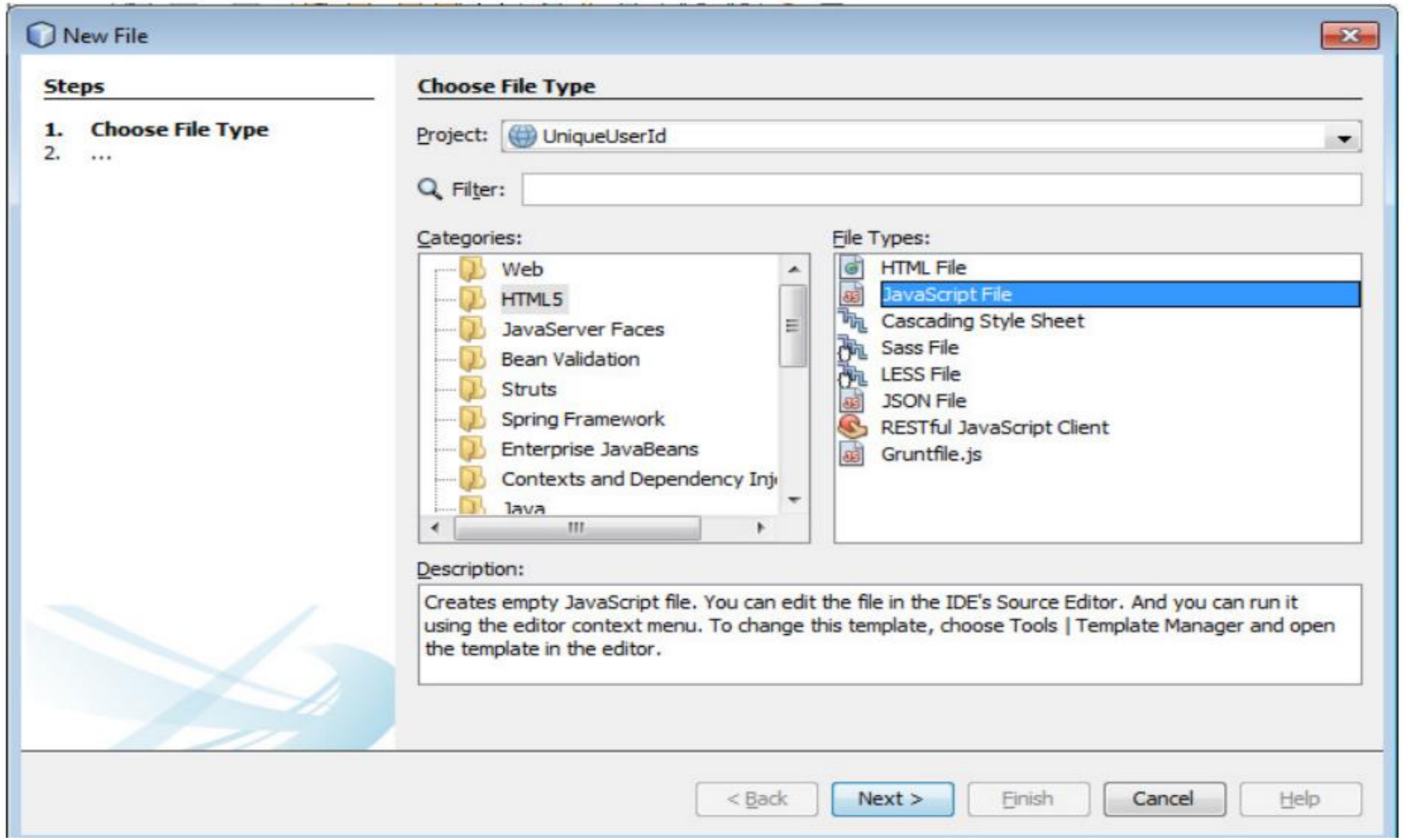
index.jsp

```
            <tr>
              <td>Name:</td>
              <td><input type="textfield" name="name"/></td>
              <td></td>
            </tr>
            <tr>
              <td>Semester:</td>
              <td><input type="textfield" name="name"/></td>
              <td></td>
            </tr>
            <tr>
              <td></td>
              <td><input type="submit"/></td>
              <td></td>
            </tr>
          </table>
        </form>
      </body>
</html>
```

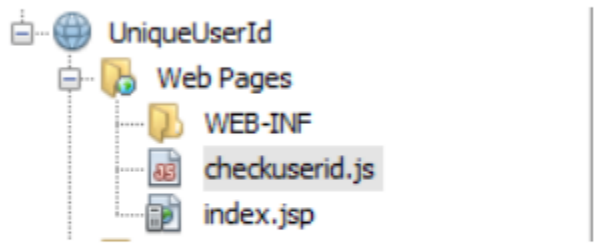
Call JavaScript method for userid field

```
<form name="registrationForm" action="register">
        <table border="0" cellpadding="5">
          <tr>
           <td>User Id:</td>
           <td><input type="textfield" name="userid" id="userid"
    onchange="doChecking();"/></td>
           <td><div id="status"></div></td>
          </tr>
```

Add JavaScript file



JavaScript file should be along with index.jsp



JavaScript file: checkuserid.js

```
var xhr;

function init() {
   useridField = document.getElementById("userid");
   statusField = document.getElementById("status");
}

function doChecking() {
      var url = "checkuserid?userid=" + escape(useridField.value);
      xhr = createXHRObject();
      xhr.open("GET", url, true);
      xhr.onreadystatechange = callback;
      xhr.send(null);
}
```

JavaScript file: checkuserid.js

```
function createXHRObject() {
    var xhrObject;
    try{
        // Opera 8.0+, Firefox, Safari
        xhrObject = new XMLHttpRequest();
    }catch (e){
        // Internet Explorer Browsers
        try{
          xhrObject = new ActiveXObject("Msxml2.XMLHTTP");
        }catch (e) {
```

JavaScript file: checkuserid.js

```
        try{
         xhrObject = new ActiveXObject("Microsoft.XMLHTTP");
         }catch (e){
            // Something went wrong
            alert("Your browser broke!");
            return false;
         }
      }
   }
}
```

Understanding of the code in JavaScript file

- The role of doChecking() is to:
  - create a URL that contains data that can be utilized by the server-side,
  - initialize an XMLHttpRequest object, and
  - prompt the XMLHttpRequest object to send an asynchronous request to the server.
- escape()
  - Replaces characters that are not legal in a URL
    - E.g., space is replaced by %20

Understanding of the code in JavaScript file

Request forming

- GET, signifies that HTTP interactions use the GET method
- true, signifying that the interaction is asynchronous:
  - req.open("GET", url, true);
- If the interaction is set as asynchronous, a callback function must be specified. The callback function for this interaction is set with the statement:
  - req.onreadystatechange = callback;
- The HTTP interaction begins when XMLHttpRequest.send() is called.

Add JavaScript in index.jsp

- Switch back to the index page and add a reference to the JavaScript file between the <head> tags.

```
<head>
    <script type="text/javascript" src="checkuserid.js"></script>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Registration Page</title>
</head>
```

- Insert a call to init() in the opening <body> tag.

```
<body onload="init()">
```

This ensures that init() is called each time the page is loaded.

Programming on server side

- We need to create
  - A Data Store
  - Servlet

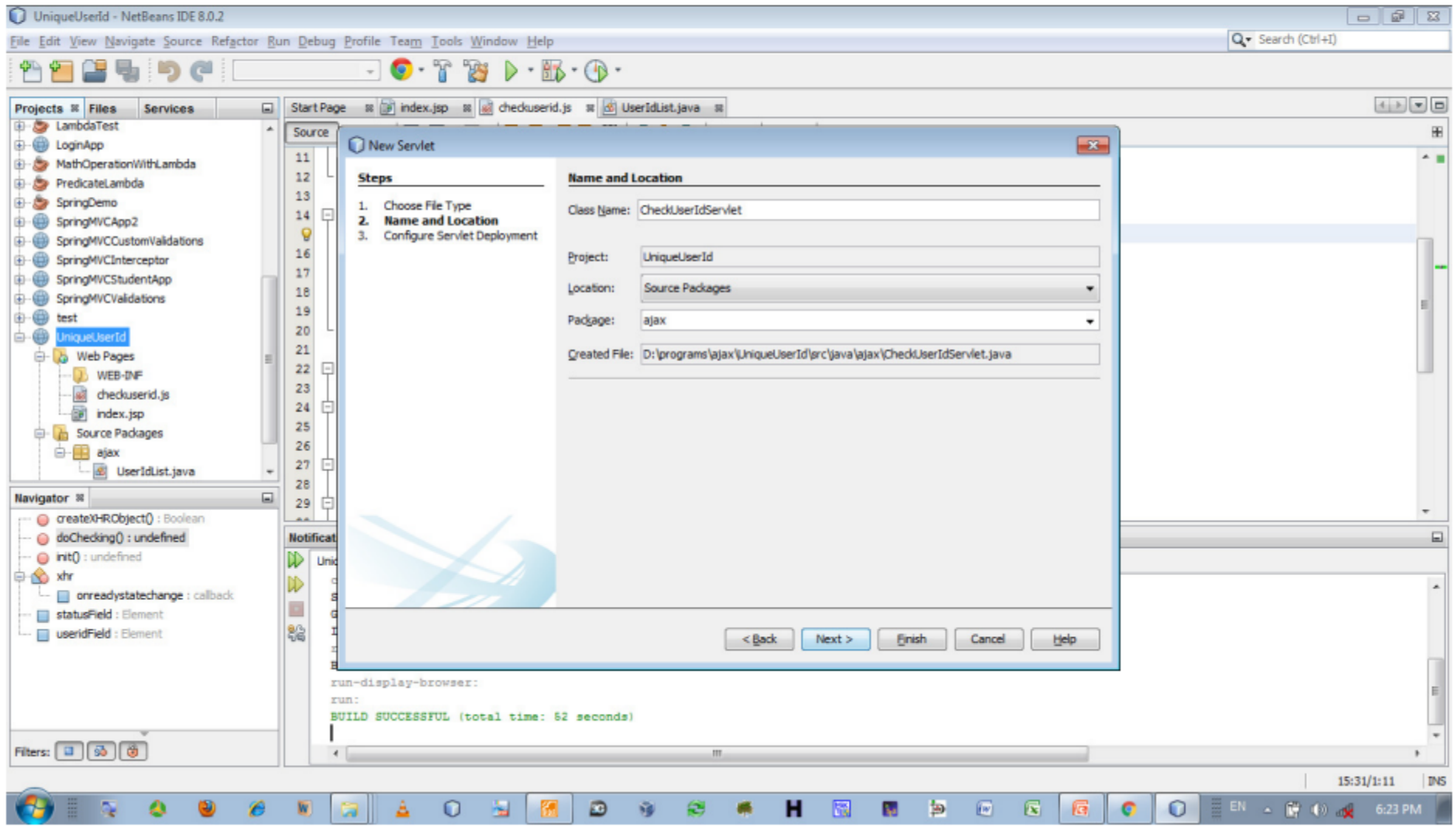
Programming on server side: Create a data store

```
package ajax;
import java.util.ArrayList;
public class UserIdList {
   private ArrayList<String> userIdList=new ArrayList<>();
   public ArrayList<String> getList(){
      return userIdList;
   }
   public UserIdList(){
      userIdList.add("a@xyz.com");
      userIdList.add("b@xyz.com");
      userIdList.add("c@xyz.com");
      userIdList.add("abc@xyz.com");
      userIdList.add("xyz@xyz.com");
      userIdList.add("def@xyz.com");
   }
```

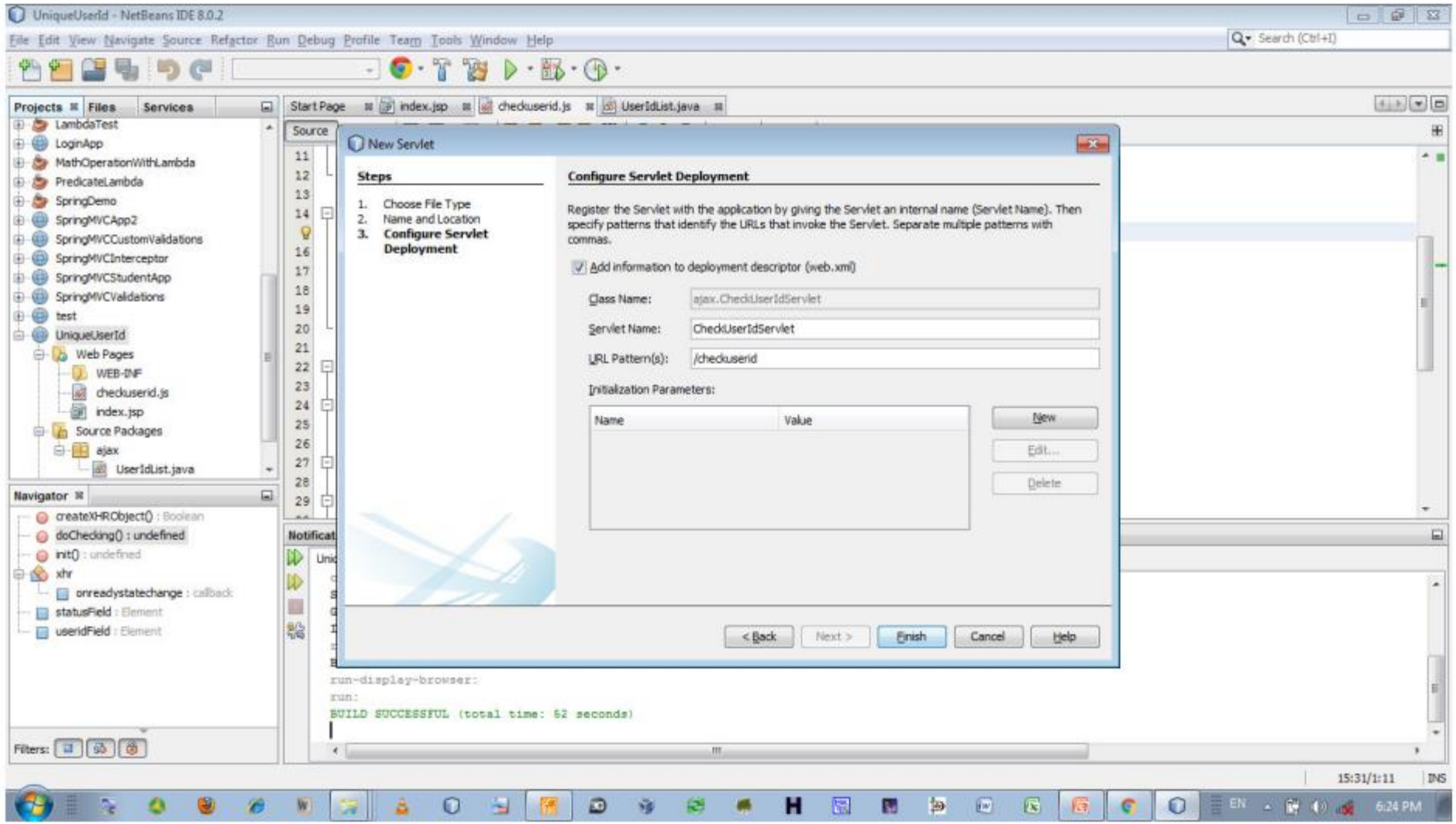
Programming on server side: Create a data store

```
public void addId(String userId){
    userIdList.add(userId);
}
public boolean isPresent(String userId){
    boolean status=false;
    for(String id:userIdList){
        if(id.compareTo(userId)==0){
            status=true;
            break;
        }
    }
    return status;
}
}
```

Programming on server side: Create a Servlet for Ajax operation



Programming on server side: Create a Servlet for Ajax operation



Programming on server side: Create a Servlet for Ajax operation

- We give servlet class name as
  - CheckUserIdServlet
- We give url pattern as
  - checkuserid

Programming on server side: Create a Servlet for Ajax operation

- Add the following data member in the servlet

```
private UserIdList userIdList=new UserIdList();
```

- Add a variable for a ServletContext object and add/modify init()

```
private ServletContext context;
@Override
  public void init(ServletConfig config) throws
  ServletException {
     this.context = config.getServletContext();
  }
```

Programming on server side: Create a Servlet for Ajax operation

- Add the following code in processRequest() method of the servlet

```
String submittedUserId=request.getParameter("userid");
  StringBuffer sBuff=new StringBuffer();
  if(submittedUserId!=null){
     submittedUserId=submittedUserId.trim().toLowerCase();
  }else{
     context.getRequestDispatcher("/error.jsp").forward(request,
response);
  }
```

Programming on server side: Create a Servlet for Ajax operation

```
if(userIdList.isPresent(submittedUserId)){
    sBuff.append("<status>");
    sBuff.append("NotAvailable");
    sBuff.append("</status>");
 }else{
    sBuff.append("<status>");
    sBuff.append("Available");
    sBuff.append("</status>");
 }
 response.setContentType("text/xml");
 response.getWriter().write(sBuff.toString());
```

Complete code of CheckUserIdServlet

```
package ajax;
import java.io.IOException;
import javax.servlet.ServletConfig;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class CheckUserIdServlet extends HttpServlet {
   private UserIdList userIdList=new UserIdList();
   private ServletContext context;
@Override
   public void init(ServletConfig config) throws ServletException {
      this.context = config.getServletContext();
   }
```

Complete code of CheckUserIdServlet

```
protected void processRequest(HttpServletRequest request,
   HttpServletResponse response)
        throws ServletException, IOException {
    String submittedUserId=request.getParameter("userid");
    StringBuffer sBuff=new StringBuffer();
    if(submittedUserId!=null){
        submittedUserId=submittedUserId.trim().toLowerCase();
    }else{
        context.getRequestDispatcher("/error.jsp").forward(request,
   response);
    }
```

Complete code of CheckUserIdServlet

```
if(userIdList.isPresent(submittedUserId)){
        sBuff.append("<status>");
        sBuff.append("NotAvailable");
        sBuff.append("</status>");
    }else{
        sBuff.append("<status>");
        sBuff.append("Available");
        sBuff.append("</status>");
    }
    response.setContentType("text/xml");
    response.getWriter().write(sBuff.toString());
  }
```

Complete code of CheckUserIdServlet

```
@Override
   protected void doGet(HttpServletRequest request,
    HttpServletResponse response)
         throws ServletException, IOException {
      processRequest(request, response);
   }
@Override
   protected void doPost(HttpServletRequest request,
    HttpServletResponse response)
         throws ServletException, IOException {
      processRequest(request, response);
   }
}
```

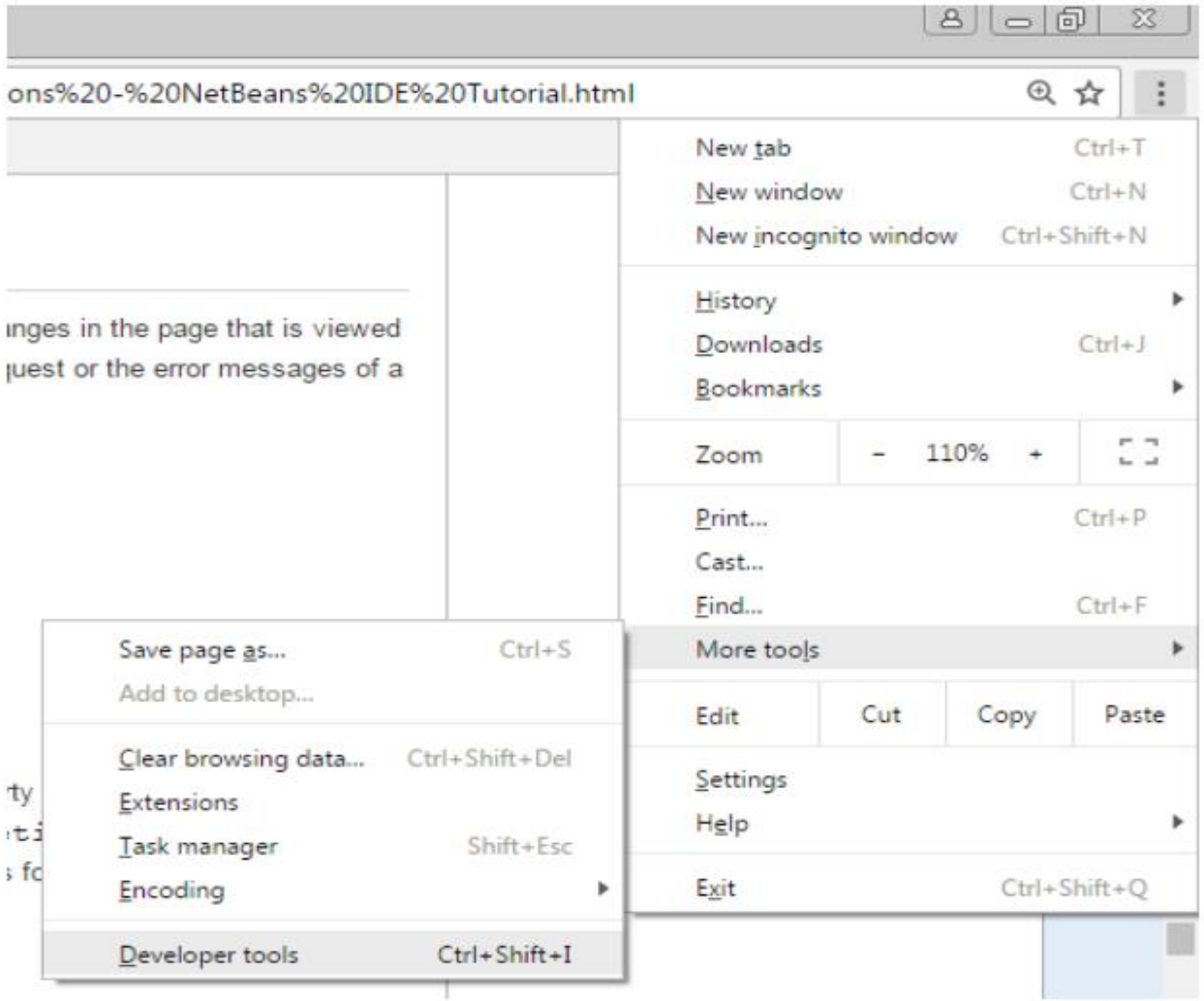
web.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd">
   <servlet>
      <servlet-name>CheckUserIdServlet</servlet-name>
      <servlet-class>ajax.CheckUserIdServlet</servlet-class>
   </servlet>
   <servlet-mapping>
      <servlet-name>CheckUserIdServlet</servlet-name>
      <url-pattern>/checkuserid</url-pattern>
   </servlet-mapping>
```

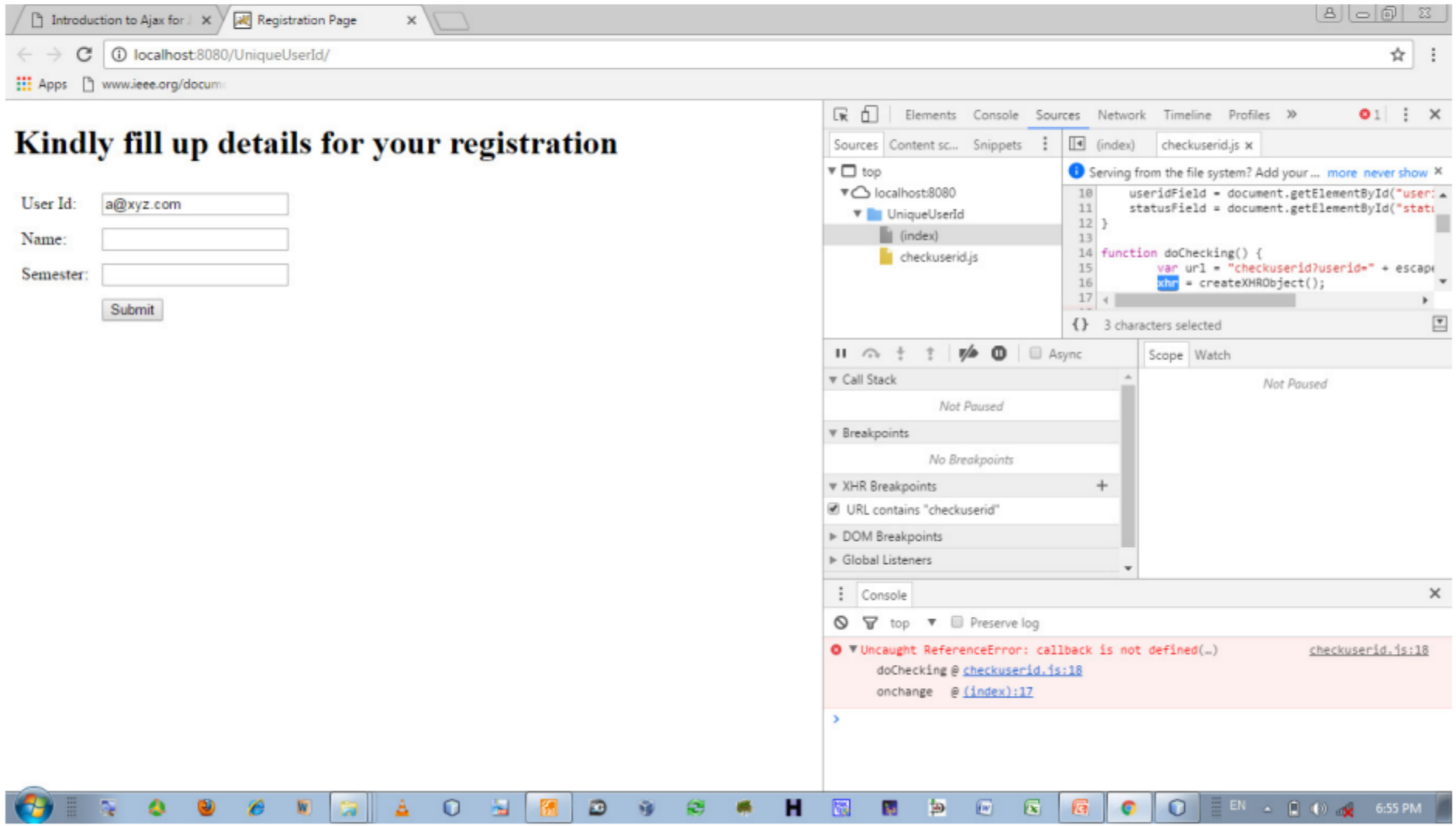
web.xml file

```
  <session-config>
    <session-timeout>
      30
    </session-timeout>
  </session-config>
</web-app>
```

How to debug client side code (on Chrome browser)



How to debug client side code (on Chrome browser)



Programming the client side (add callback processing)

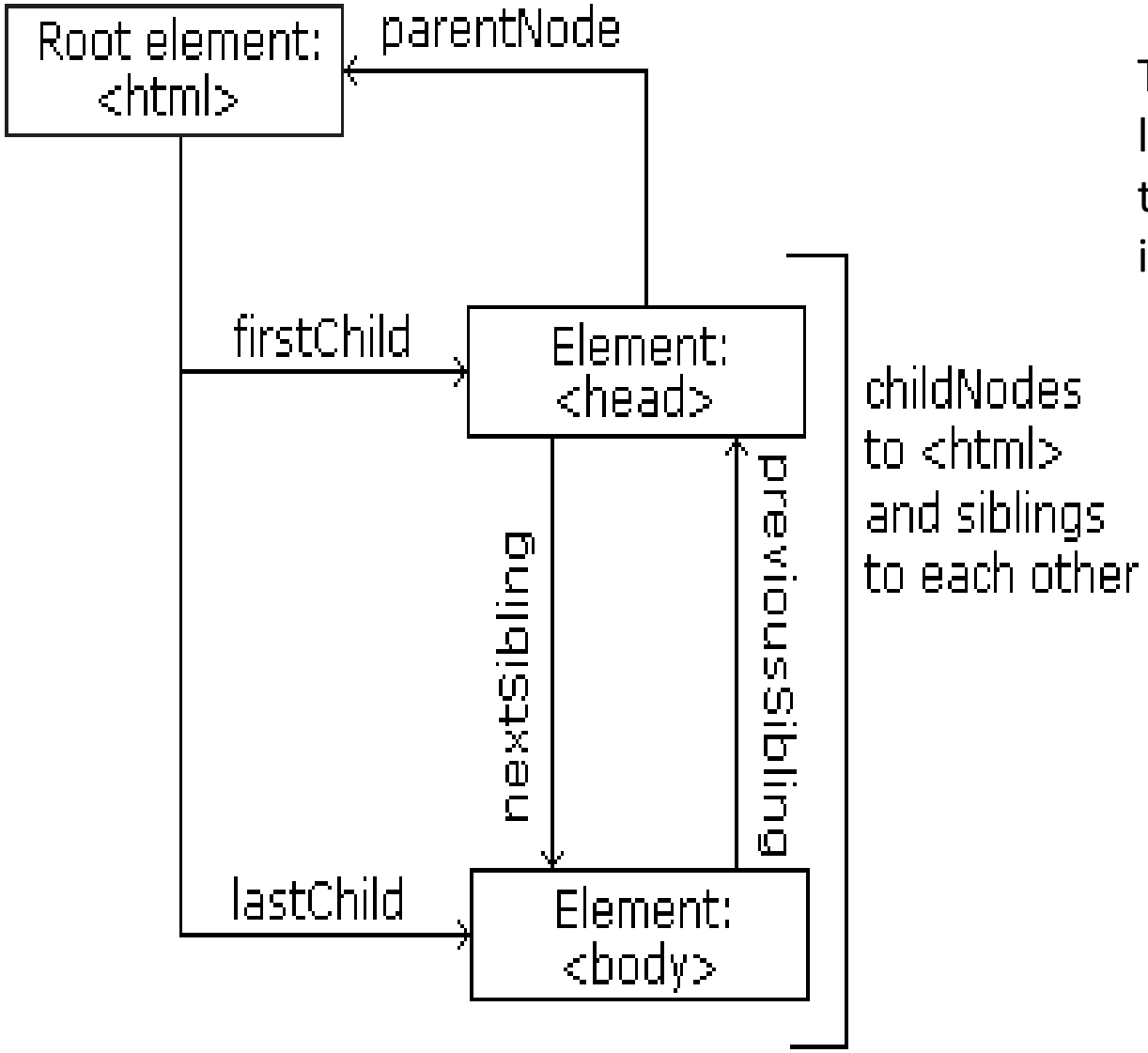
- Add following in checkuserid.js file

```
function callback() {
    if (xhr.readyState == 4) {
        if (xhr.status == 200) {
            setMessaage(xhr.responseXML);
        }
    }
}
```

Programming the client side (add callback processing)

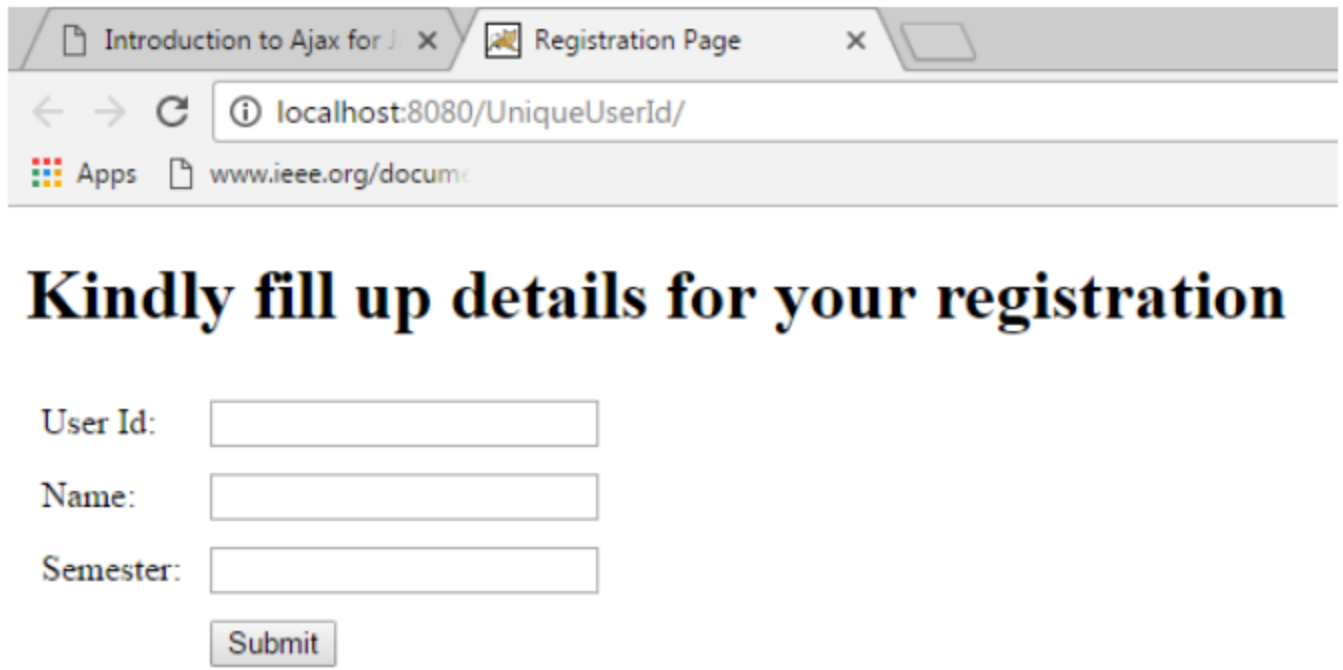
- Add following in checkuserid.js file

```
function setMessaage(responseXML){
   var statusElement=responseXML.getElementsByTagName("status")[0];
   var status=statusElement.childNodes[0].nodeValue;
   if(status=="NotAvailable"){
      statusField.innerHTML = "<font color=red>This User Id is already in use.
     Please choose some other User Id.</font>";
   }else{
      statusField.innerHTML = "<font color=green>You can use this User
     ID.</font>";
   }
}
```



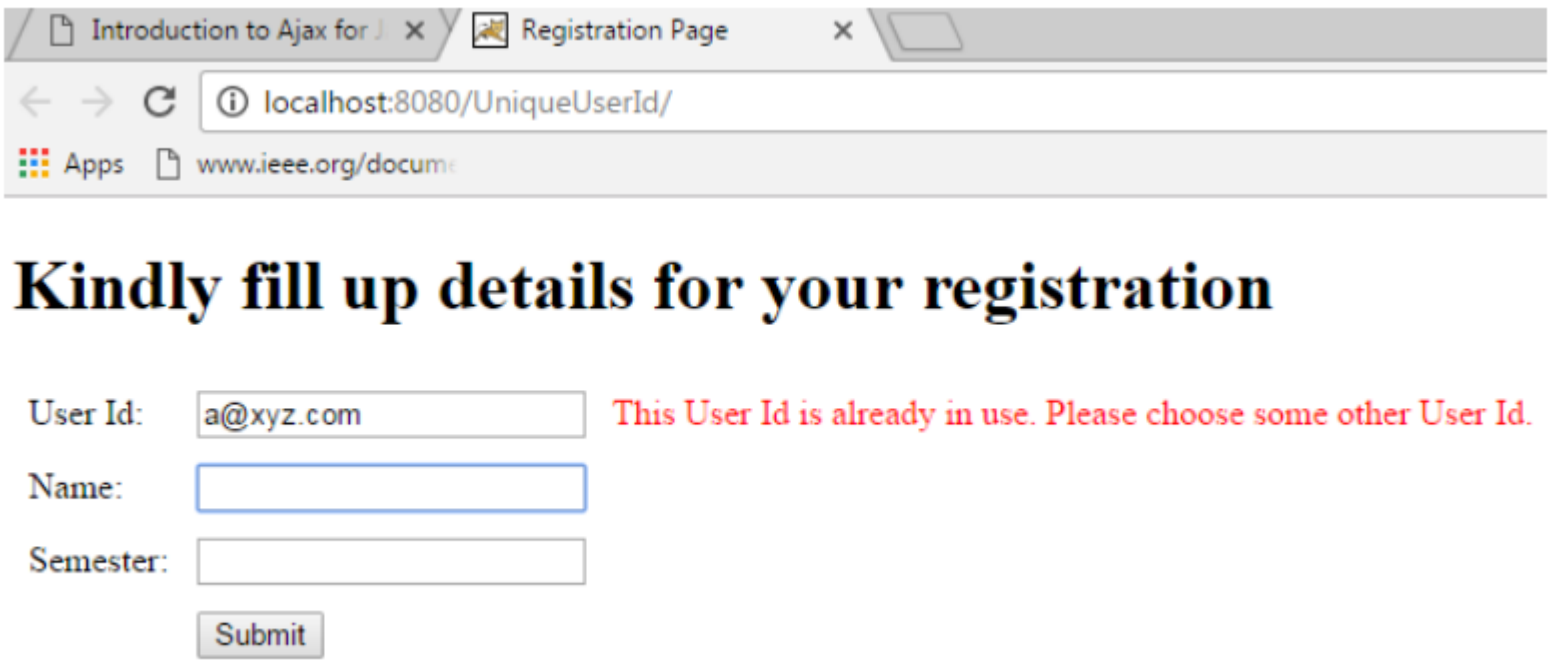
The childNodes property of the Node interface returns a live NodeList of child nodes of the given element where the **first child node is assigned index 0**. Child nodes include elements, text and comments.

Running the application



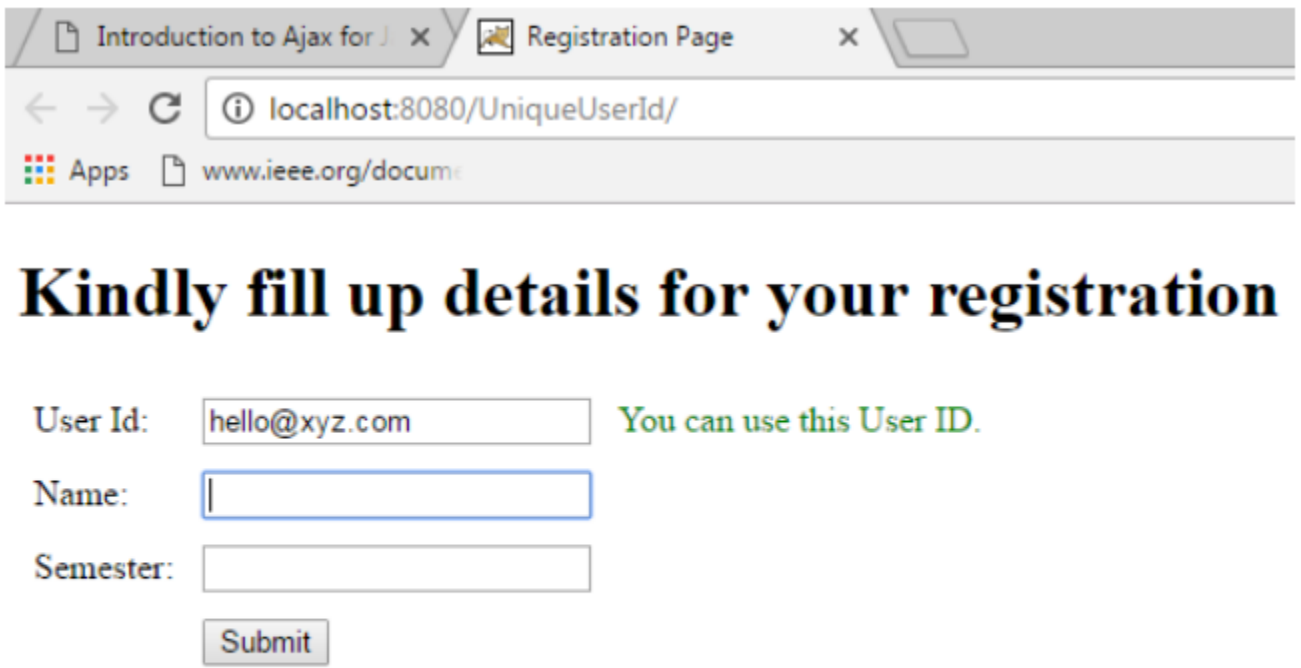
Running the application

- Write a user name that is already in use (a@xyz.com) and press tab key



Running the application

- Write a user name that is not already in use (hello@xyz.com) and press tab key



Asynchronous - True or False?

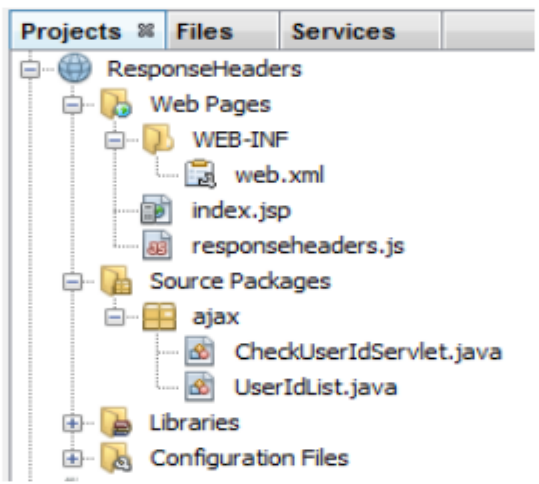
- To send the request asynchronously, the async parameter of the open() method has to be set to true:
  - Many of the tasks performed on the server are very time consuming or even network might be slow.
  - The JavaScript does not have to wait for the server response, but can instead execute other scripts while waiting for server response.
- When using async = true, we need to specify a function to execute when the response is ready in the onreadystatechange event.

Asynchronous - True or False?

- To use async=false, we need to pass the third parameter in the open() method to false:
  - xhttp.open("GET", url, false);
- Using async=false is not recommended, but for a few small requests this might be use.
  - JavaScript will NOT continue to execute, until the server response is ready.
  - If the server is busy or slow, the application will hang or stop.
- When we use async=false, we do not need to write an onreadystatechange function
- We can put the response processing code after the send() statement:

AJAX application – get response headers

- We copy our existing AJAX project “UniqueUserId” to “ResponseHeaders”
- We modify the following:
  - JavaScript file (name as well as code)
  - index.jsp
  - CheckUserIdServlet



AJAX application – get response headers

- index.jsp file

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <script type="text/javascript" src="responseheaders.js"></script>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Response headers</title>
  </head>
  <body onload="init()">
    <h1>Response headers</h1>
```

AJAX application – get response headers

- index.jsp file

```
 <form name="responseHeadersForm" action="#">
    <table border="0" cellpadding="5">
       <tr>
          <td><input type="button" value="Get all headers"
onclick="getHeaders();"/></td>
       </tr>
       <tr>
        <td><div id="headers"></div></td>
       </tr>
    </table>
 </form>
</body>
</html>
```

AJAX application – get response headers

- We change name of JavaScript file as responseheaders.js

```
var xhr;

function init() {
   headersField = document.getElementById("headers");
}

function getHeaders() {
      var url = "checkuserid";
      xhr = createXHRObject();
      xhr.open("GET", url, true);
      xhr.onreadystatechange = callback;
      xhr.send(null);
}
```

AJAX application – get response headers

- responseheaders.js file

```
function createXHRObject() {
   var xhrObject;
   try{
      // Opera 8.0+, Firefox, Safari
      xhrObject = new XMLHttpRequest();
   }catch (e){
      // Internet Explorer Browsers
      try{
        xhrObject = new ActiveXObject("Msxml2.XMLHTTP");
      }catch (e) {
         try{
         xhrObject = new ActiveXObject("Microsoft.XMLHTTP");
         }catch (e){
            // Something went wrong
            alert("Your browser broke!");
            return false;
         }
      }
   }
   return xhrObject;
}
```

AJAX application – get response headers

- responseheaders.js file

```
function callback() {
   if (xhr.readyState == 4) {
      if (xhr.status == 200) {
         setMessaage(xhr);
      }
   }
}

function setMessaage(xhr){
   headersField.innerHTML=xhr.getAllResponseHeaders();
}
```

AJAX application – get response headers

- CheckUserIdServlet.java file

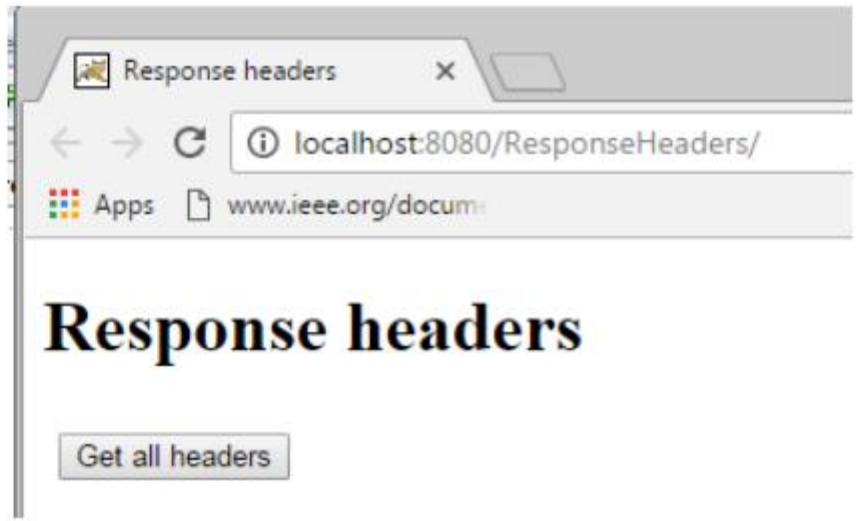
```
package ajax;

import java.io.IOException;
import javax.servlet.ServletConfig;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class CheckUserIdServlet extends HttpServlet {
   private UserIdList userIdList=new UserIdList();
   private ServletContext context;
```

AJAX application – get response headers

- CheckUserIdServlet.java file

```
@Override
  public void init(ServletConfig config) throws ServletException {
     this.context = config.getServletContext();
  }
  protected void processRequest(HttpServletRequest request, HttpServletResponse
    response)
        throws ServletException, IOException {
     StringBuffer sBuff=new StringBuffer();
     sBuff.append("<status>");
     sBuff.append("Successful");
     sBuff.append("</status>");
     response.setContentType("text/xml");
     response.getWriter().write(sBuff.toString());
  }
}
```

Running the AJAX application – get response headers



Running the AJAX application – get response headers