SERVLETS
(Request Processing)

Requests and Responses

- What is a request ?
  - Information that is sent from client to a server
    - Who made the request
    - What user-entered data is sent
    - Which HTTP headers are sent
- What is a response ?
  - Information that is sent to client from a server
    - Text(html, plain) or binary(image) data
    - HTTP headers, cookies, etc

What is ServletRequest ?

- Contains data passed from client to servlet.
- All servlet requests implement **ServletRequest** interface which defines methods for accessing
  - Client sent parameters
  - Object-valued attributes
  - Locales
  - Client and server
  - Input stream
  - Protocol information
  - Content type
  - If request is made over secure channel (HTTPS)

ServletRequest Interface

- Javax.servlet.ServletRequest interface of Servlet API abstracts data from the client to the servlet.
- The ServletRequest object encapsulates the information requested by a client to the servlet and provides data, such as the request parameters and headers.
- The servlet container creates this object, sets the request information into it and passes it to the servlet as an argument of the service().
- When processing of a request is complete the servlet container sets the request object to default and uses the same object for other requests.

Sending Request from Clients

- A web browser sends an HTTP request to a web server when any of the following events happen:
  - **A user clicks on a hyperlink in an HTML page.**
  - **A user fills out a form in an HTML page and submits it.**
  - **A user enters a URL in the browser's address field and presses Enter.**

ServletRequest Interface contains the methods for

1. Read data from a Client
2. Get client information
3. Get Server Information

ServletRequest Methods for retrieving client-sent parameters

Methods	Description
String getParameter (String paramName)	Returns just one of the values associated with the given parameter and returns null if no such parameter is present.
String[ ] getParameterValues (String paramName)	Returns an array of string objects containing all the values in request parameter or returns null
Enumeration getParameterNames()	The method is useful when the names of the parameters are unknown. It is possible to iterate through the Enumeration of Strings returned by the method using the getParameter( ) or getParameterValues () .

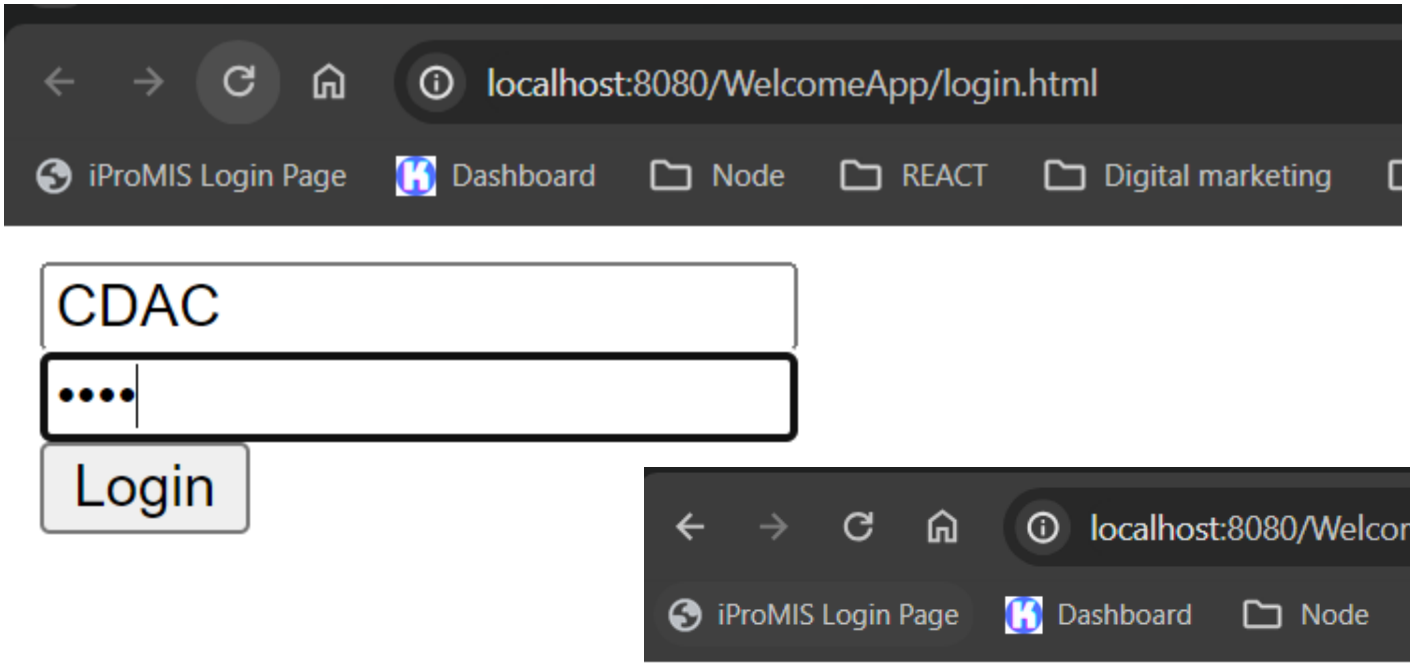
Login Example(Login.html)

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <meta charset="ISO-8859-1">
5  <title>Insert title here</title>
6  </head>
7  <body>
8  <form action="login" method="post">
9  <div>
10 <input type="text" name="txtUserName" placeholder="User Name"></input>
11 </div>
12 <div>
13 <input type="password" name="txtUserPwd" placeholder="Password"></input>
14 </div>
15 <button type="submit">Login</button>
16 </form>
17 </body>
18 </html>
```

```java
package com.requestprocess.servlets;
import java.io.IOException;
@WebServlet(name = "login", urlPatterns = { "/login" })
public class LoginServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter pw = response.getWriter();
        String uName = request.getParameter("txtUserName");
        String upwd = request.getParameter("txtUserPwd");
        pw.println("<html>");
        pw.println("<body>");
        pw.println("<h2> User Details are: <br>User Name : " + uName +
                "<br> Password : " + upwd + "</h2");
        pw.println("</body>");
        pw.println("</html>");
    }
}
```

O/p


localhost:8080/WelcomeApp/login.html
iProMIS Login Page
Dashboard
Node
REACT
Digital marketing
CDAC
Login
localhost:8080/WelcomeApp/login
iProMIS Login Page
Dashboard
Node
REACT
User Details are:
User Name : CDAC
Password : 1234

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://xmlns.jcp.org/xml/ns/javaee"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
   http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
  version="4.0">
  <welcome-file-list>
        <welcome-file>
        login.html
        </welcome-file>

    </welcome-file-list>
</web-app>
```

loginGetParamValues.html – 1/2

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    <form action="logingetparamvalues" method="post">
        <h3 align="center">Login Here...!</h3>
        <table border="1" align="center">
            <tr>
                <td>User Name</td>
                <td><input type="text" name="txtUserName"
                    placeholder="User Name"></input></td>
            </tr>
            <tr>
                <td>Password</td>
                <td><input type="password" name="txtUserPwd"
                    placeholder="Password"></input></td>
            </tr>
```

loginGetParamValues.html – 2/2

```
            <tr>
                <td>Select your Subjects:</td>
                <td>
                <input type="checkbox" name="subject" value="C" />C
                <input type="checkbox" name="subject" value="C++" />C++
                <input type="checkbox" name="subject" value="Java" />Java
                <input type="checkbox" name="subject" value="React" />React
                </td>
            </tr>
            <tr>
                <td colspan="2" align="center">
                    <button type="submit">Login</button>
                </td>
            </tr>
        </table>
    </form>
</body>
</html>
```

LoginGetParamValues.java -1/3

```java
package com.requestprocess.servlets;
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


@WebServlet(name = "logingetparams", urlPatterns = { "/logingetparams" })
public class LoginGetParamValues extends HttpServlet {
    private static final long serialVersionUID = 1L;
```

LoginGetParamValues.java -2/3

```
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter pw = response.getWriter();
    String userName = request.getParameter("txtUserName");
    String password = request.getParameter("txtUserPwd");
    String[] subject = request.getParameterValues("subject");
    if (userName.equals("cdac") && (password.equals("cdac"))) {
        System.out.println(".........inside if" + subject);
        pw.println("<html>");
        pw.println("<body>");
        pw.println("<h2> Welcome to :  " + userName + "<br> </h2");
        pw.println("<h3>Your Subjects are :</h3><br> ");
        for (String s : subject) {
            pw.println(s + " <br>");
            System.out.println(s + " ");
        }
```

LoginGetParamValues.java -3/3

```
            pw.println("</body>");
            pw.println("</html>");
        } else {
            pw.println("<html>");
            pw.println("<body>");
            pw.println("<h2> Login failed </h2");
            pw.println("</body>");
            pw.println("</html>");
        }

    }

}
```

loginGetParamNames.html – 1/2

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    <form action="logingetparamsnames" method="post">
        <h3 align="center">Login Here...!</h3>
        <table border="1" align="center">
            <tr>
                <td>User Name</td>
                <td><input type="text" name="txtUserName"
                    placeholder="User Name"></input></td>
            </tr>
            <tr>
                <td>Password</td>
                <td><input type="password" name="txtUserPwd"
                    placeholder="Password"></input></td>
            </tr>
```

loginGetParamNames.html – 2/2

```
            <tr>
                <td>Select your Subjects:</td>
                <td><input type="checkbox" name="subject" value="C" />C <input
                    type="checkbox" name="subject" value="C++" />C++ <input
                    type="checkbox" name="subject" value="Java" />Java <input
                    type="checkbox" name="subject" value="React" />React</td>
            </tr>
            <tr>
                <td colspan="2" align="center">
                    <button type="submit">Login</button>
                </td>
            </tr>
        </table>
    </form>
</body>
</html>
```

LoginGetParamNames.java -1/2

```java
package com.requestprocess.servlets;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.Enumeration;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet(name = "logingetparamsnames", urlPatterns = { "/logingetparamsnames" })
public class LoginGetParamNames extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
```

LoginGetParamNames.java -2/2

```
        response.setContentType("text/html");
        PrintWriter pw = response.getWriter();
        Enumeration<String> subject = request.getParameterNames();
        pw.println("<body>");
        pw.print("<h3>Values are");
        while (subject.hasMoreElements()) {
            pw.print("<h3>" + request.getParameter(subject.nextElement().toString()) +
                    "</h3>");
        }
        pw.println("</body>");
    }

}
```

Methods for getting server information

- Methods for getting server information

Methods	Description
String getServerName()	Returns the host name of the server to which the request was sent.
int getServerPort()	Returns the port number to which the request was sent.

Get server details from request object

```
package com.requestprocess.servlets;

import java.io.IOException;

@WebServlet(name = "getserverdetails", urlPatterns = { "/getserverdetails" })
public class GetRequestServerDetails extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter pw = response.getWriter();

        pw.println("<h3>Server details :"+request.getServerName()+"</h3");
        pw.println("<h3><br>Server Port :"+request.getServerPort()+"</h3");
        pw.println("<h3>Server details :"+request.getServletPath()+"</h3>");
        }

}
```

Methods for getting client information

- Servlet can get client information from the request.

Methods	Description
String getRemoteAddr()	Returns the IP address of the client or last proxy that sent the request
String getRemoteHost()	Returns the fully qualified name of the client or the last proxy that sent the request

Introduction to Servlet Response

- The ServletResponse interface contains various methods that enable a servlet to respond to the client requests.
- A servlet can send the response either as character or binary data.
- The PrintWriter stream can be used to send character data as servlet response, and ServletOutputStream stream to send binary data as servlet response.

Methods in **Servlet Response**

Methods	Description
String getCharacterEncoding()	It returns the name of the MIME charset used in the response sent to the client.
String getContentType()	It returns the response content type. e.g. text, html etc.
ServletOutputStream getOutputStream()	Returns a ServletOutputStream suitable for writing binary data in the response.
PrintWriter getWriter()	Returns the PrintWriter object.
void setCharacterEncoding(String charset)	Set the MIME charset (character encoding) of the response.
void setContentLength(int len)	It sets the length of the response body.
void setContentType(java.lang.String type)	Sets the type of the response data.

ResponseServlet.java

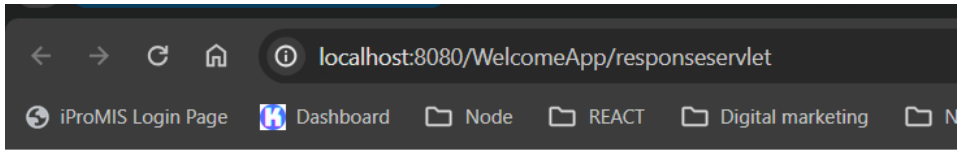
```java
package com.responseprocess.servlets;

import java.io.IOException;

@WebServlet(name = "responseservlet", urlPatterns = { "/responseservlet" })
public class ResponseServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
           response.setContentType("text/html;charset=UTF-8");
            PrintWriter out = response.getWriter();
            try {
               out.println("<!DOCTYPE html>");
                out.println("<html>");
                out.println("<head>");
                out.println("<title>My Servlet </title>");
                out.println("</head>");
                out.println("<body>");
                out.println("<h1>" + response.getCharacterEncoding() + "</h1>");
                out.println("<h1>" + response.getContentType() + "</h1>");
                out.println("</body>");
                out.println("</html>");
            } finally {
                out.close();
            }

    }
```



UTF-8

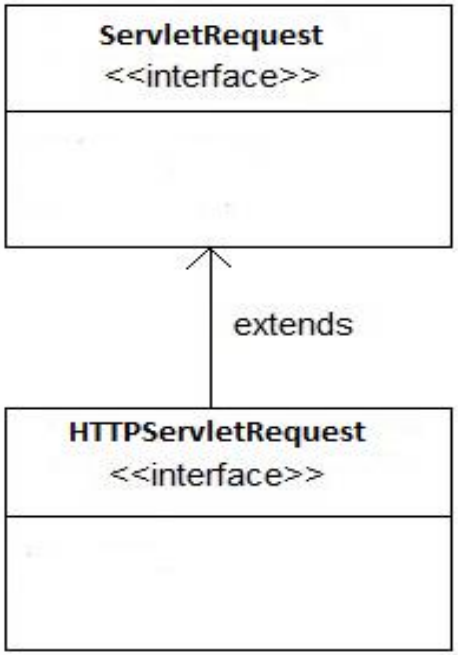
text/html;charset=UTF-8

```java
package com.responseprocess.servlets;

import java.io.BufferedInputStream;

@WebServlet(name = "streamservlet", urlPatterns = { "/streamservlet" })
public class StreamServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("image/jpeg");
        ServletOutputStream out;
        out = response.getOutputStream();
        FileInputStream img = new  FileInputStream(request.getServletContext().getRealPath("/")+
                "img/smile.jpg");
        BufferedInputStream bin = new BufferedInputStream(img);
        BufferedOutputStream bout = new BufferedOutputStream(out);
        int ch = 0;
        while ((ch = bin.read()) != -1) {
            bout.write(ch);
        }
        bin.close();
        img.close();
        bout.close();
        out.close();
    }
}
```

HTTPServletRequest Interface



Whenever a client makes an http request, it is received by the servlet container in http protocol semantics; the servlet then creates an HttpServletRequest object, stuffs it with request parameters, headers, cookies etc that is sent by the client and passes it to the servelt instance for further processing

Methods	Description
String getContextPath()	returns the portion of the request URI that indicates the context of the request
Cookies getCookies()	returns an array containing all of the Cookie objects the client sent with this request
String getQueryString()	returns the query string that is contained in the request URL after the path
HttpSession getSession()	returns the current HttpSession associated with this request or, if there is no current session and create is true, returns a new session
String getMethod()	Returns the name of the HTTP method with which this request was made, for example, GET, POST, or PUT.
String getServletPath()	returns the part of this request's URL that calls the servlet

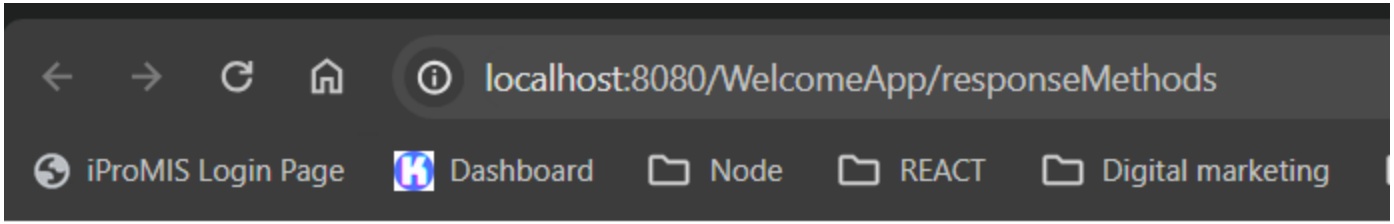
ResponseMethods.java

```
package com.responseprocess.servlets;

import java.io.IOException;
@WebServlet("/responseMethods")
public class ResponseMethods extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
    ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {

            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet httpreq</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet Context Path:  " + request.getContextPath() + "</h1>");
            out.println("<h1>Servlet Servlet Path: " + request.getServletPath() + "</h1>");
            out.println("<h1>Servlet GetMethod: " + request.getMethod()+ "</h1>");
            out.println("</body>");
            out.println("</html>");
        } finally {
            out.close();
        }
    }
```



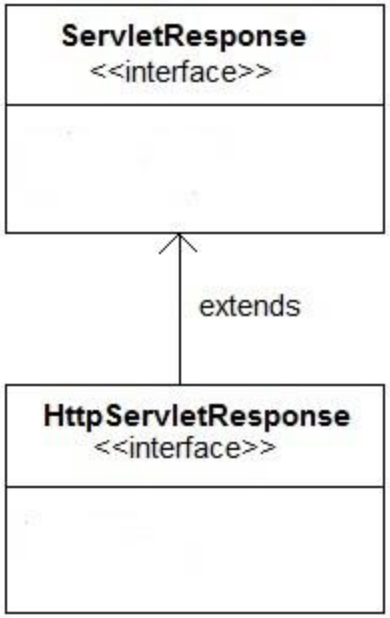
Servlet Context Path: /WelcomeApp

Servlet Servlet Path: /responseMethods

Servlet GetMethod: GET

HttpServletResponse Interface

- **HttpServletResponse** interface adds the methods that relates to the **HTTP** response.



Methods	Description
void addCookie(Cookie cookie)	adds the specified cookie to the response.
void sendRedirect(String location)	Sends a temporary redirect response to the client using the specified redirect location URL and clears the buffer
int getStatus()	gets the current status code of this response
String getHeader(String name)	gets the value of the response header with the given name.
void setHeader(String name, String value)	sets a response header with the given name and value
void setStatus(int sc)	sets the status code for this response
void sendError(int sc)	sends an error response to the client using the specified status

HTTP Response Status Codes

public void setStatus(int statusCode)

- Status codes are defined in HttpServletResponse
- Status codes are numeric fall into five general categories:

- 100-199 Informational
- 200-299 Successful
- 300-399 Redirection
- 400-499 Incomplete
- 500-599 Server Error
- Default status code is 200 (OK)

Introduction to sendRedirect() Method

- sendRedirect() method redirects the response to another resource. This method actually makes the client(browser) to create a new request to get to the resource. The client can see the new url in the browser.
- **sendRedirect()** accepts relative **URL**, so it can go for resources inside or outside the server.

SendRedirects.html

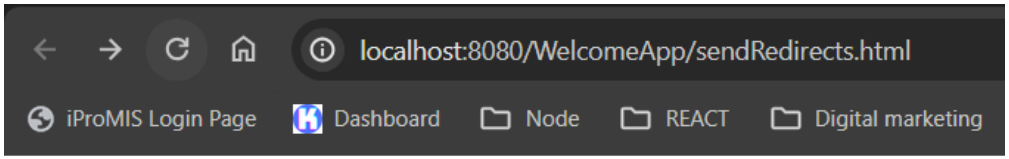
```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    <form action="sendredirect" method="post">
        <div>
            <input type="text" name="txtUserName" placeholder="User Name"></input>
        </div>
        <div>
            <input type="password" name="txtUserPwd" placeholder="Password"></input>
        </div>
        <button type="submit">Login</button>
    </form>
</body>
</html>
```

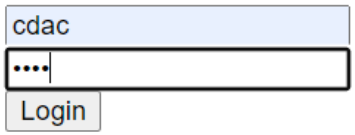
SendRedirectsServlets.java

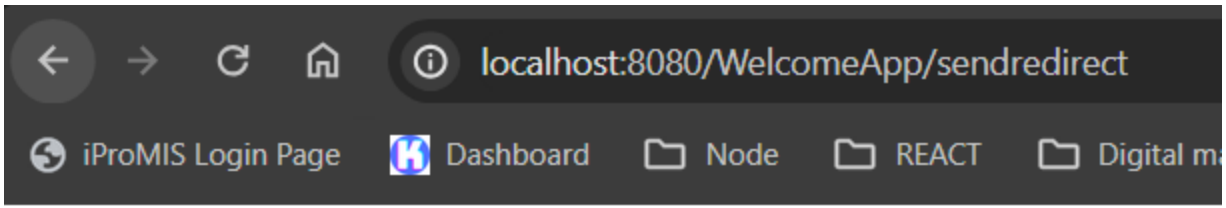
```java
@WebServlet(name = "sendredirect", urlPatterns = { "/sendredirect" })
public class SendRedirectServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter pw = response.getWriter();
        String userName = request.getParameter("txtUserName");
        String password = request.getParameter("txtUserPwd");
        if (userName.equals("cdac") && (password.equals("cdac"))) {
            pw.println("<html>");
            pw.println("<body>");
            pw.println("<h2> Welcome to :  " + userName + "<br> </h2");
            pw.println("</body>");
            pw.println("</html>");
        } else {
            try {
                response.sendRedirect("sendRedirects.html");
            } finally {
                pw.close();
            }
        }
    }
}
```

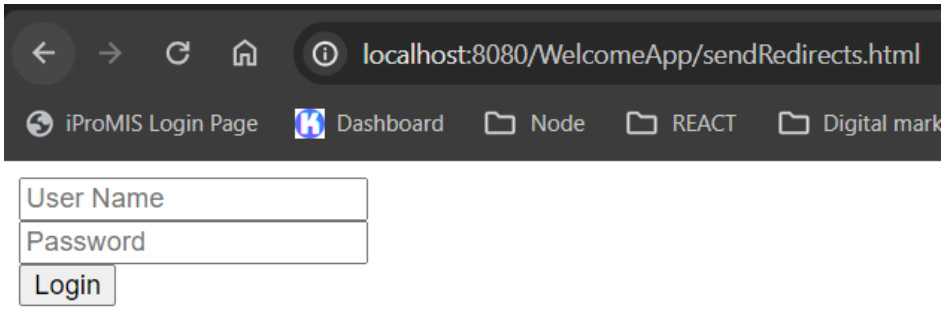
Output







Welcome to : cdac



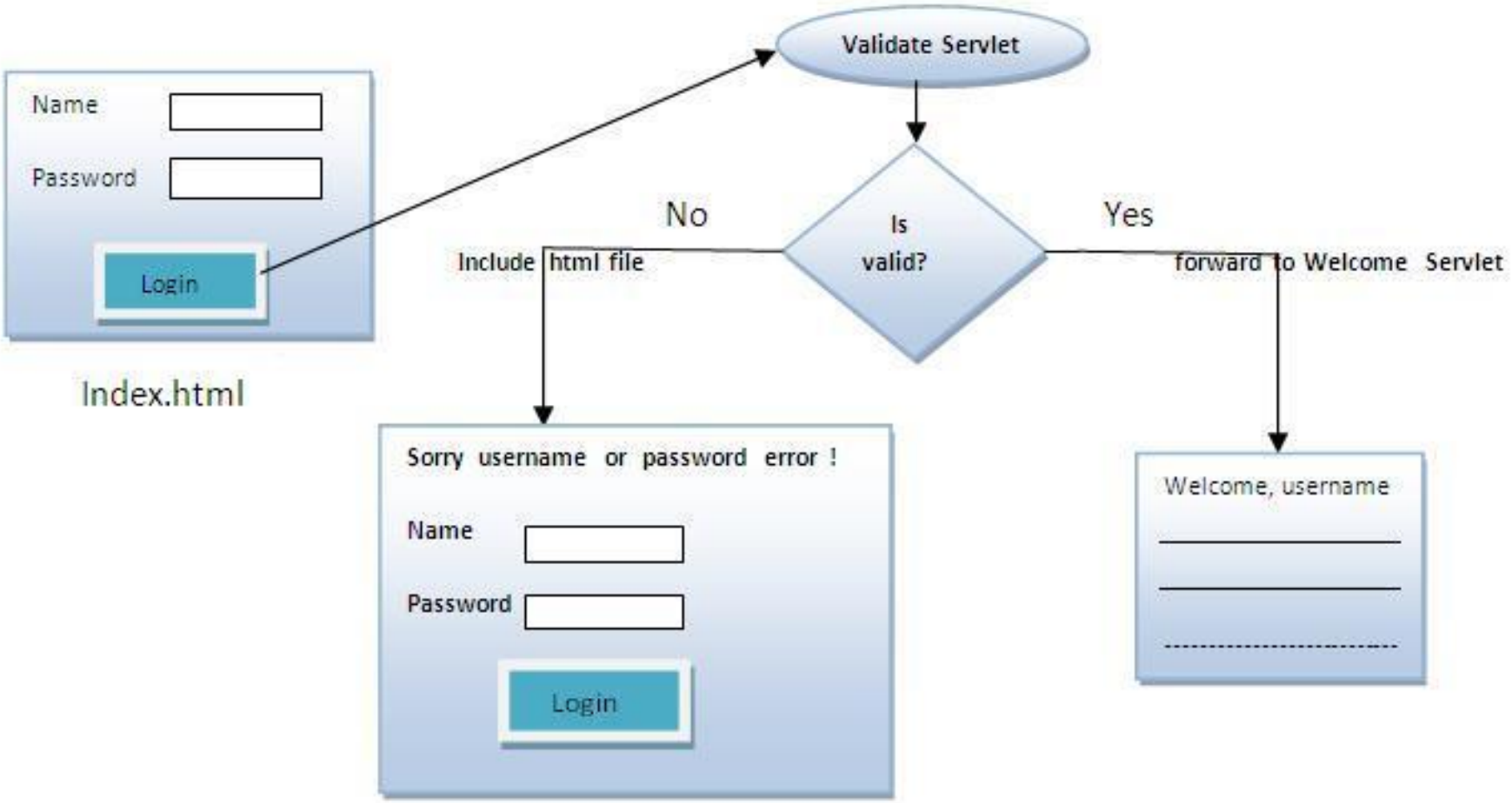
Introduction to Request Dispatcher

- **RequestDispatcher** is an interface, implementation of which defines an object which can dispatch request to any resources(such as HTML, Image, JSP, Servlet) on the server.

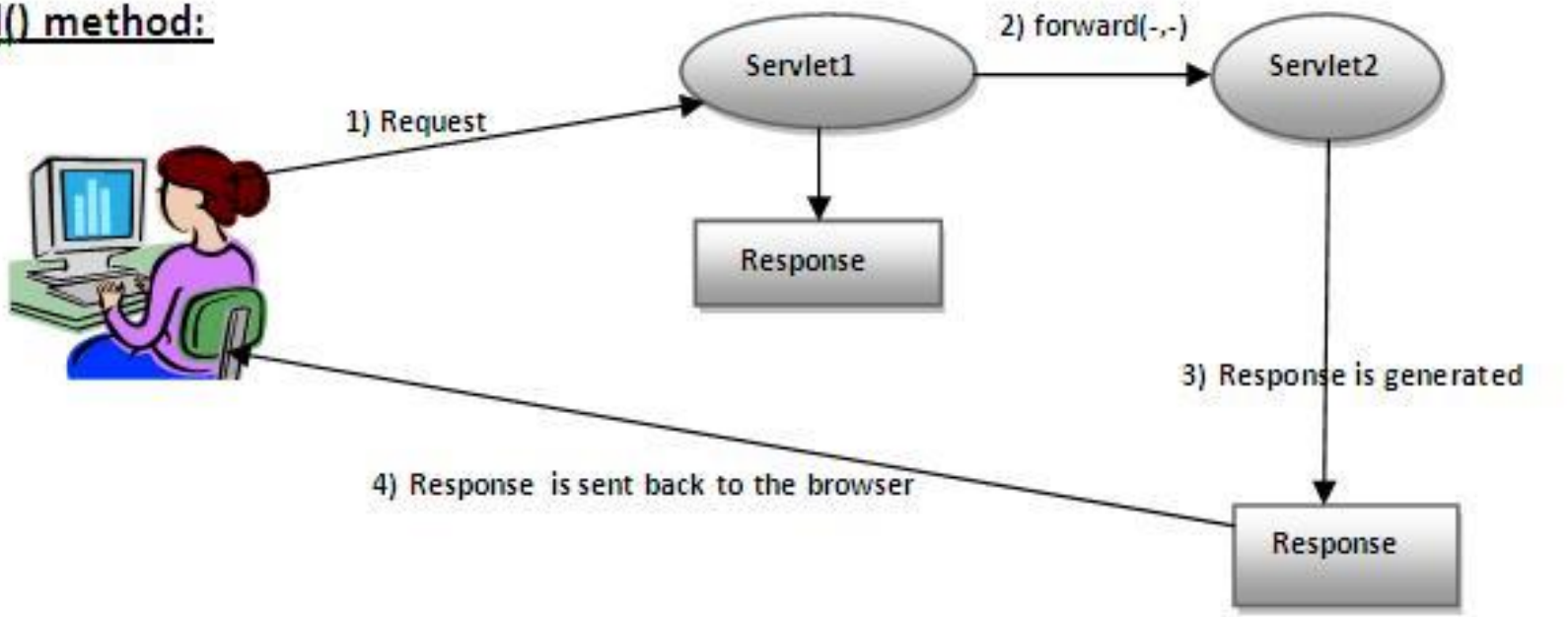
Methods of RequestDispatcher

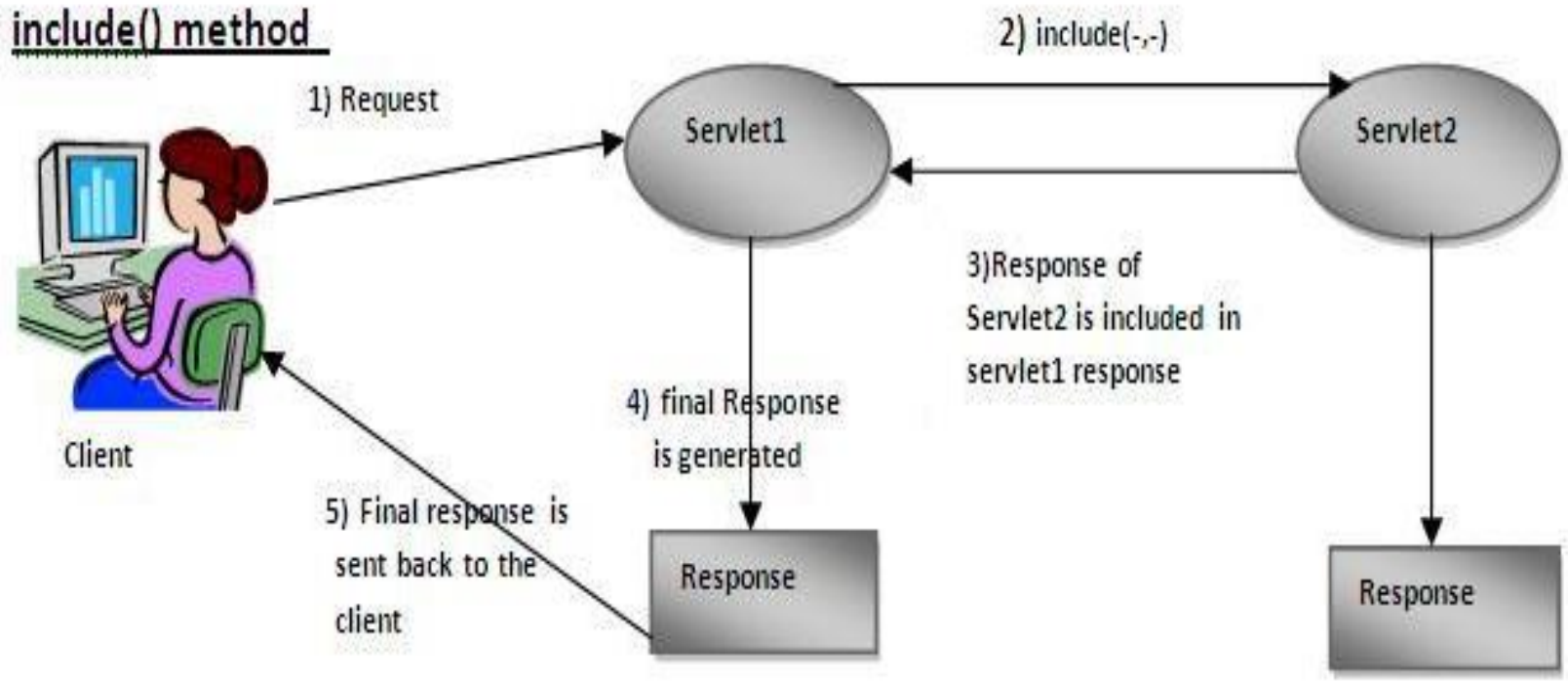
RequestDispatcher interface provides two important methods

Methods	Description
void `forward(ServletRequest request, ServletResponse response)`	forwards a request from a servlet to another resource (servlet, JSP file, or HTML file) on the server
void `include(ServletRequest request, ServletResponse response)`	includes the content of a resource (servlet, JSP page, HTML file) in the response


Validate Servlet
Name
Password
Login
Index.html
No
Is
valid?
Yes
Include html file
forward to Welcome Servlet
Sorry username or password error !
Name
Password
Login
Welcome, username

forward() method:




include() method
1) Request
2) include(-,-)
Servlet1
Servlet2
3)Response of
Servlet2 is included in
servlet1 response
Client
4) final Response
is generated
5) Final response is
sent back to the
client
Response
Response

How to get an Object of RequestDispatcher

- getRequestDispatcher() method of **ServletRequest** returns the object of **RequestDispatcher**.

```
RequestDispatcher rs = request.getRequestDispatcher("hello.html");
```

Forward()


ServletRequest object
resource name
RequestDispatcher rs = request.getRequestDispatcher("hello.html");
rs.forward(request,response);
forward the request and response to "hello.html" page

Include()


ServletRequest object
Resource name
RequestDispatcher rs = request.getRequestDispatcher("first.html");
rs.include(request,response);
include the response of "first.html" page in current servlet response

sendRedirect() and Request Dispatcher

- The main difference between a **redirection** and a **request dispatching** is that, redirection makes the client(browser) create a new request to get to the resource, the user can see the new URL while request dispatch get the resource in same request and URL does not changes.
- Also, another very important difference is that, sendRedirect() works on **response** object while request dispatch work on **request** object.

Thank You