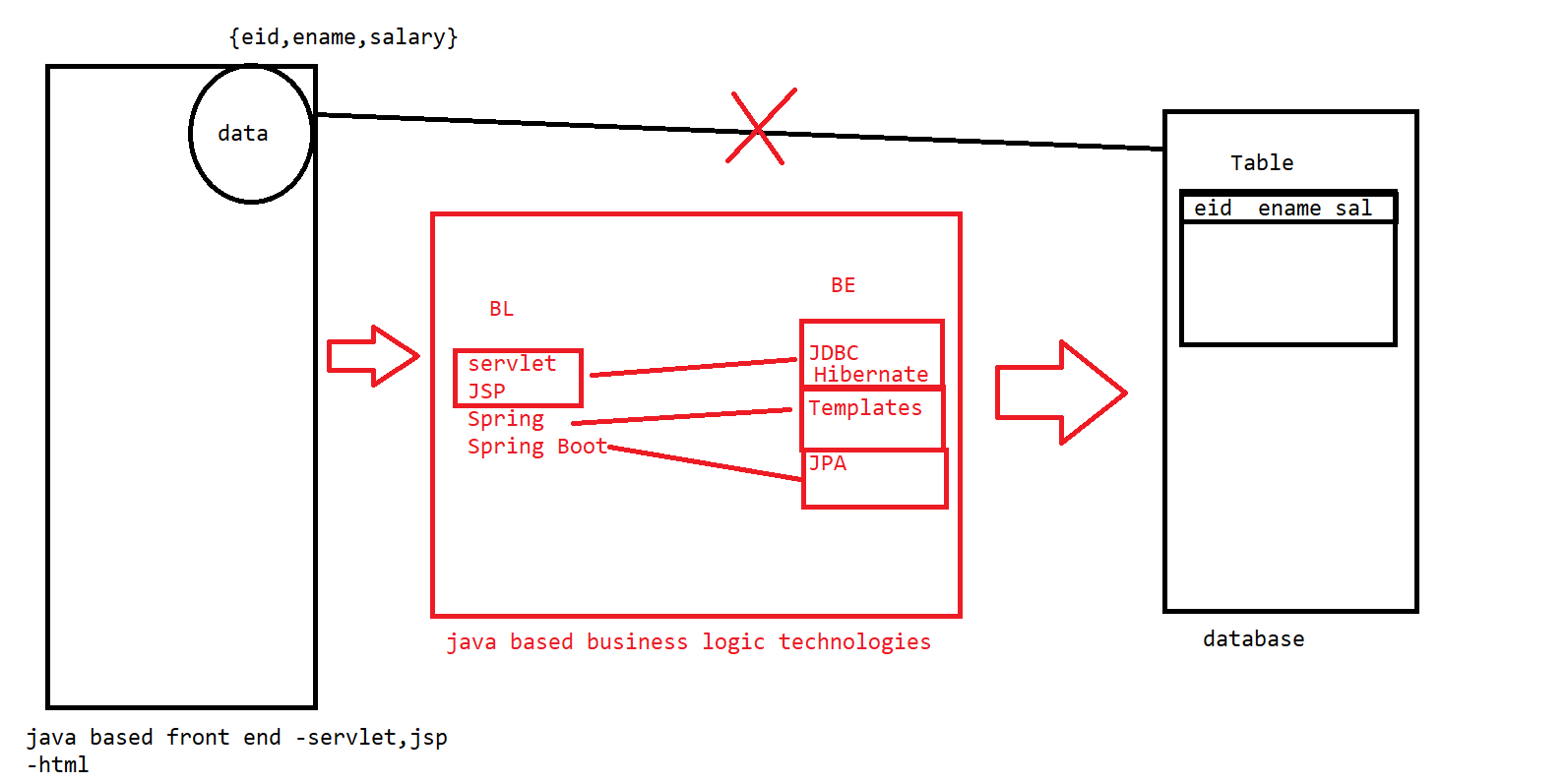
Phase 2:



We cannot send the data directly into the dB from the plain HTML files.

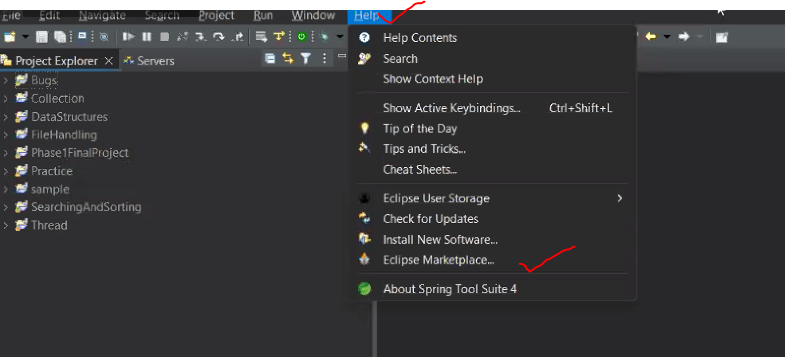
We need a technology in the middle that can support the FE data to be pushed into the BE databases.

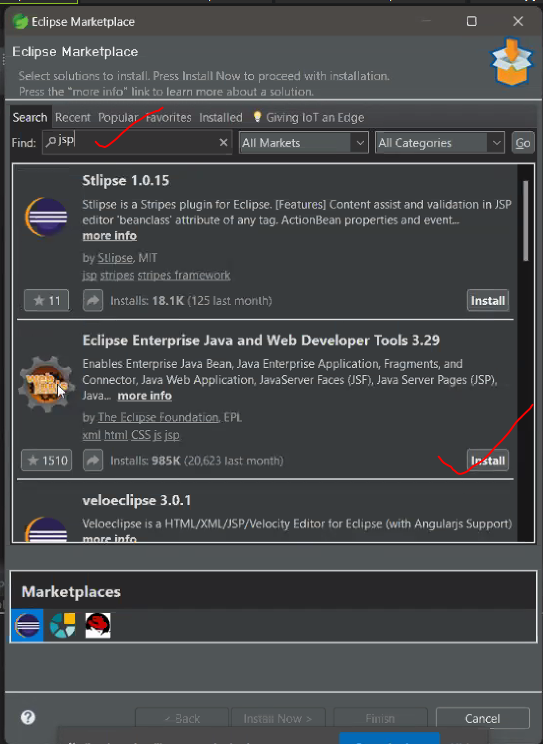
//middleware technologies

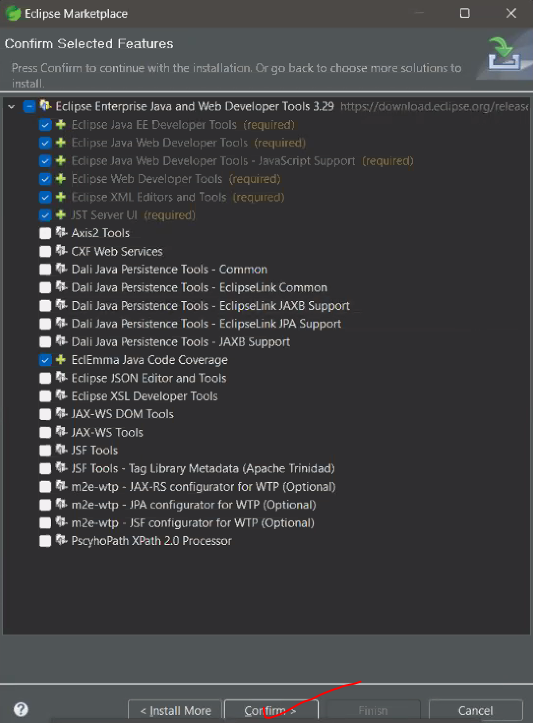
Software :

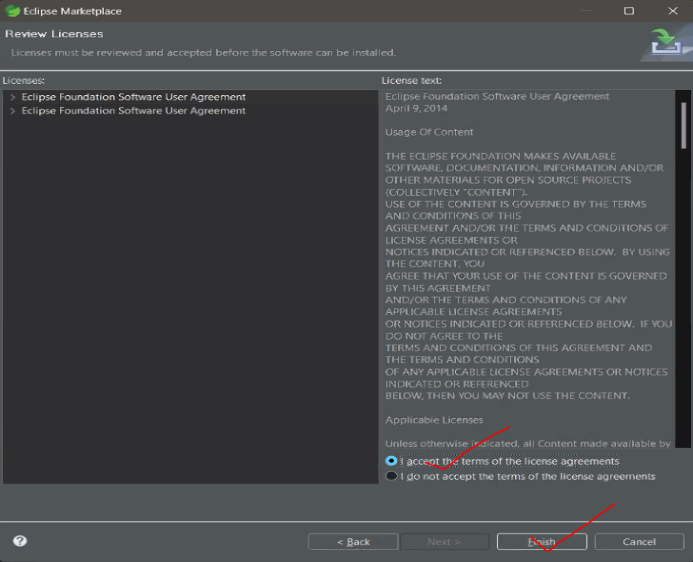
<https://drive.google.com/file/d/1_hBfQU1_Ufai1M2zbBGnLQdzONIFrHfx/view?usp=share_link>

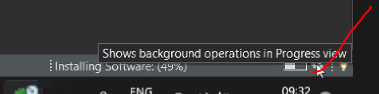
Upgrade STS

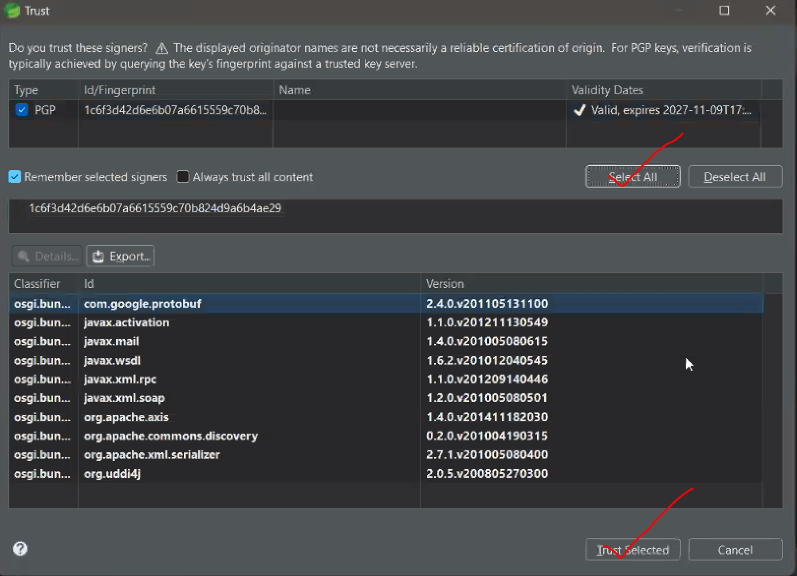




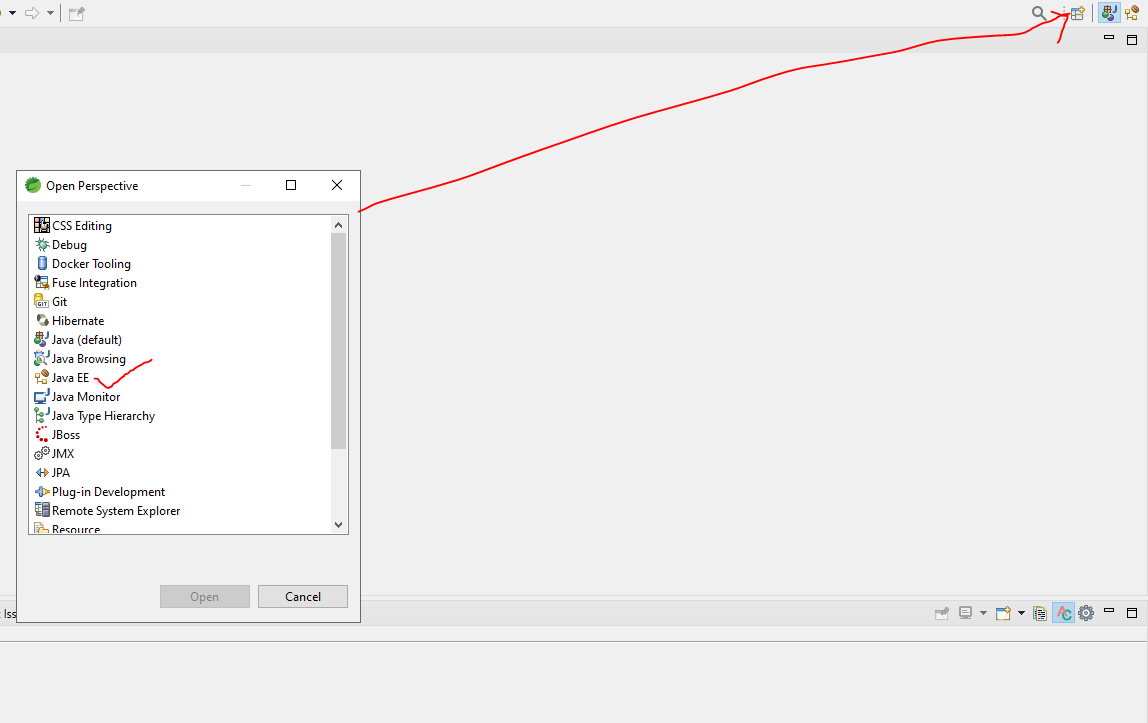


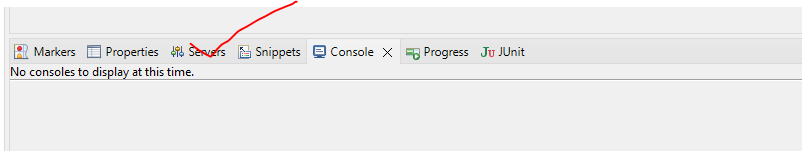


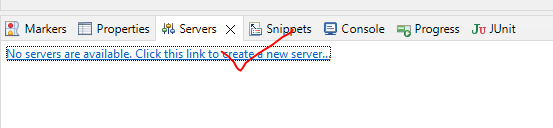


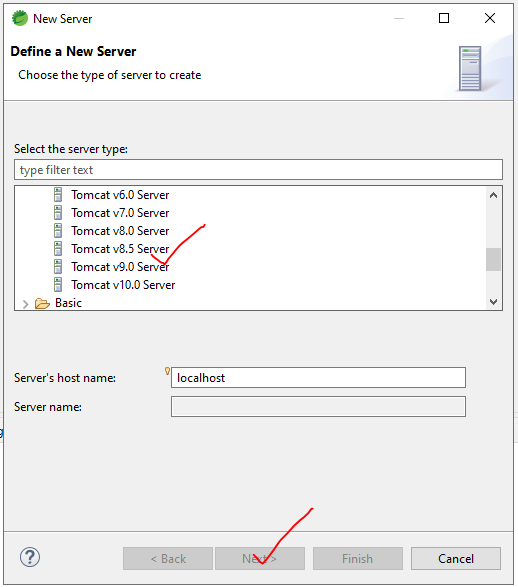


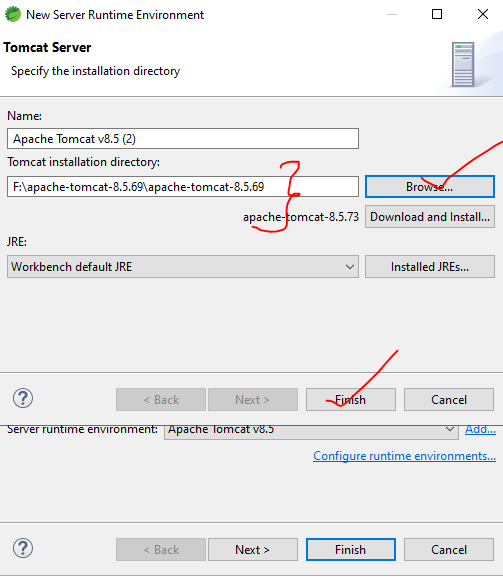
Add Tomcat to STS

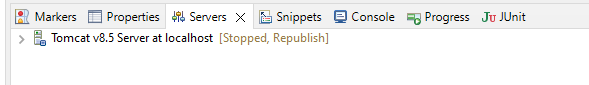


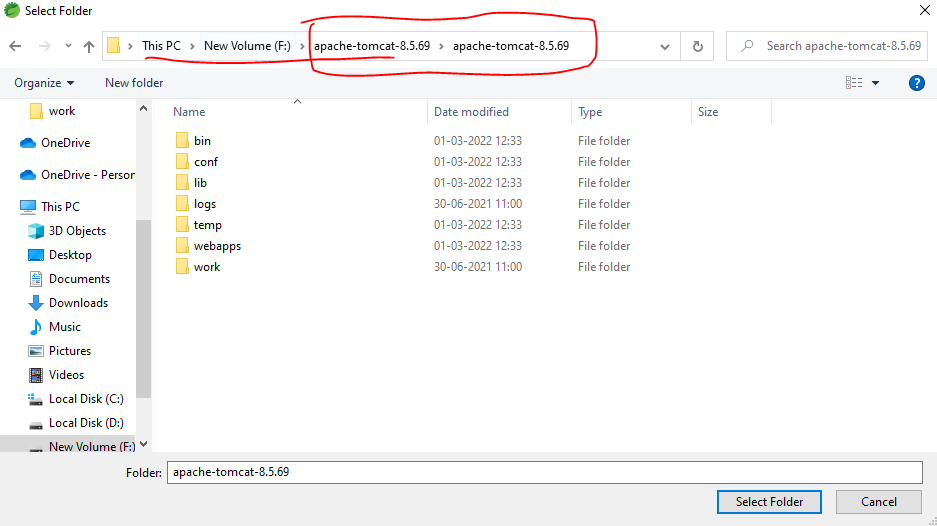




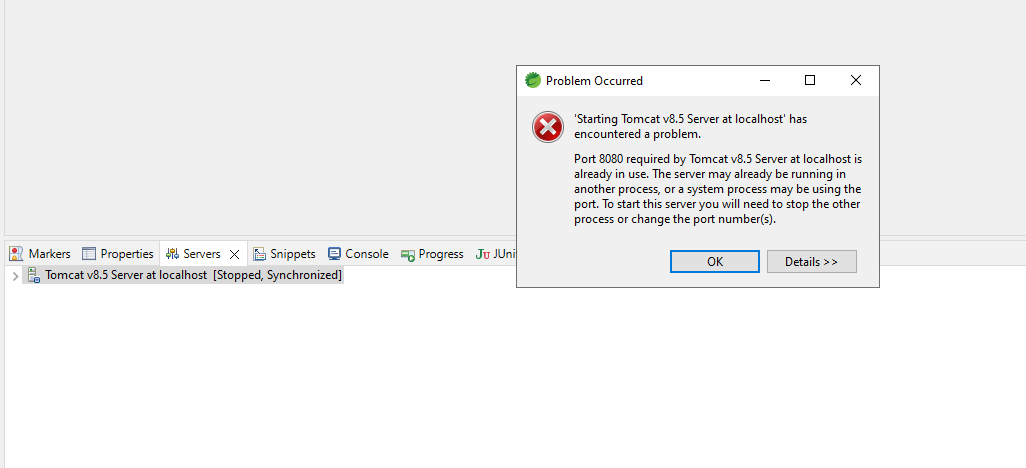




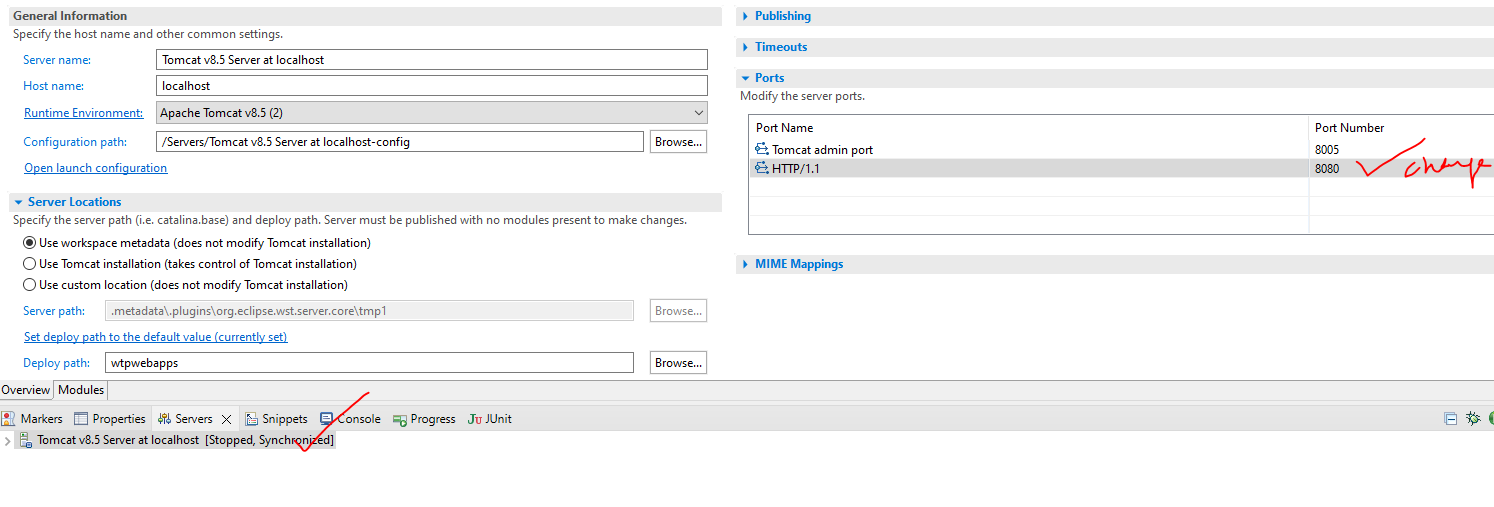




If we get any port issues

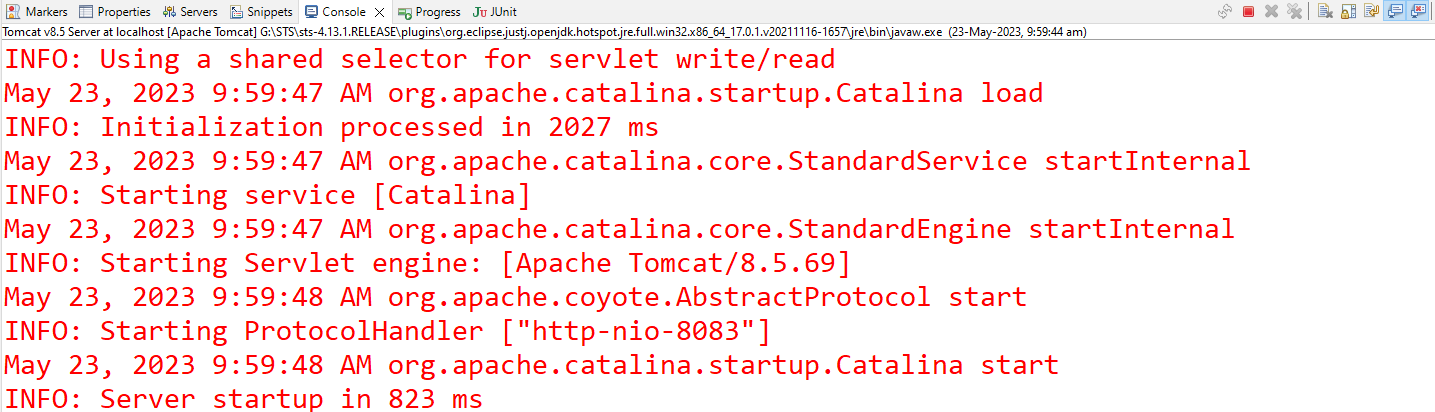


Double-click on the server



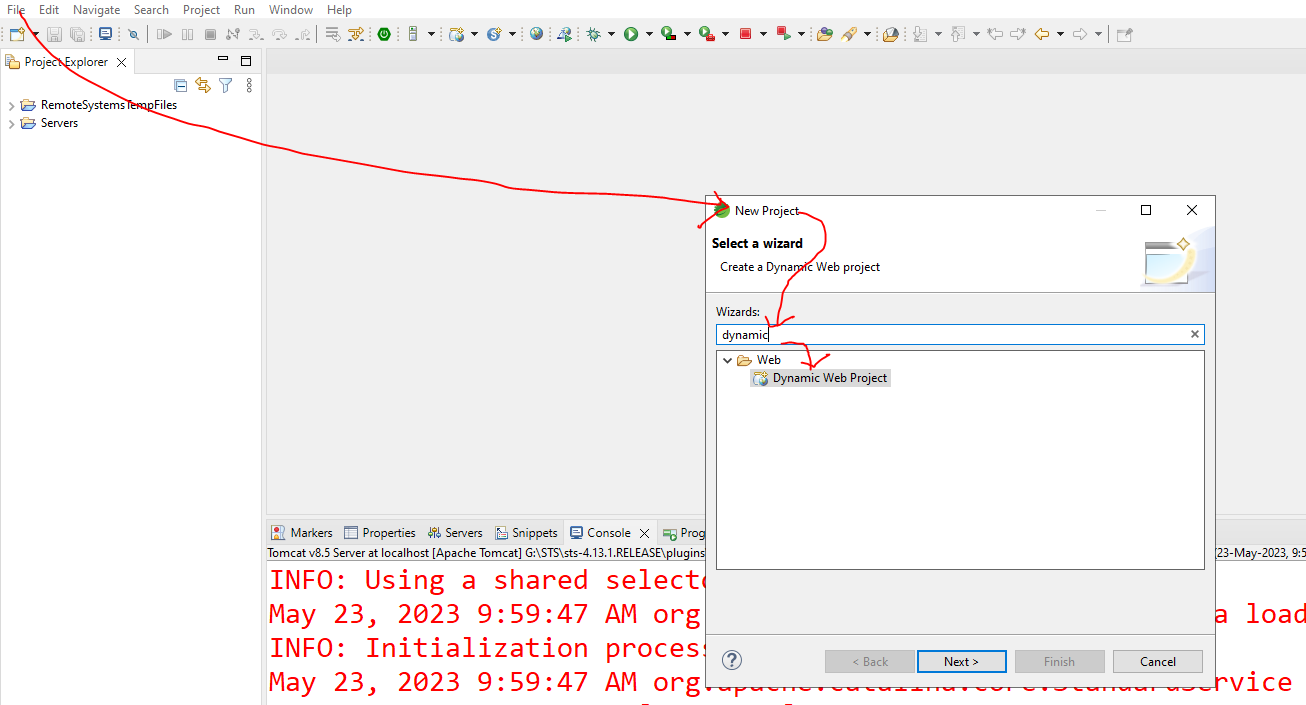
Cntl+s to save

Then start the server

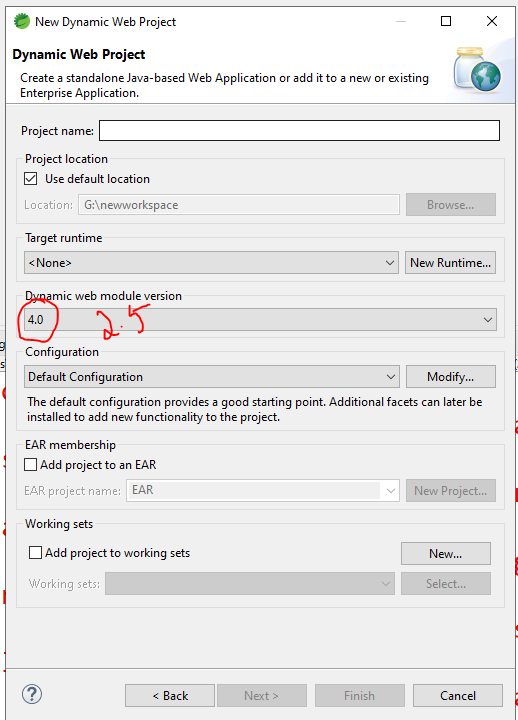


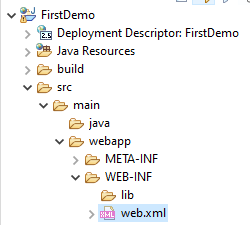
By default, the Apache Tomcat server will be on the port 8080.

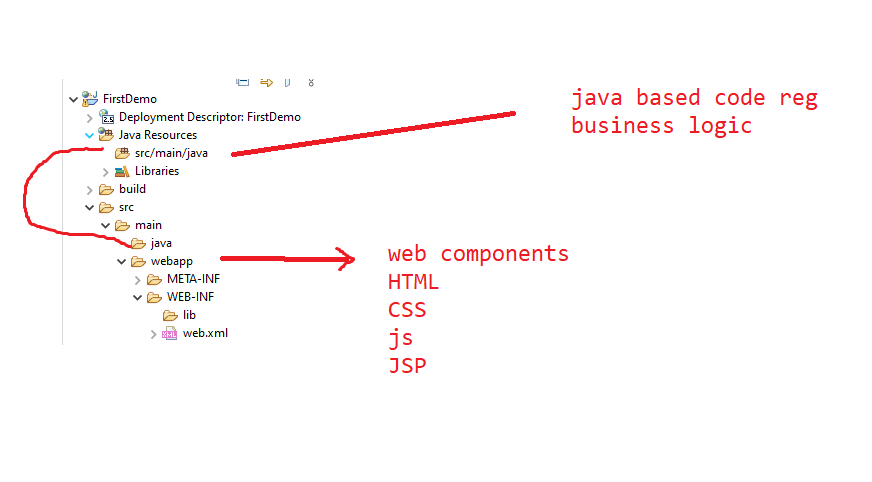
>Creation of web project



Note: take the project on the web module version of 2.5 ===> so that we can get the config file of web.xml

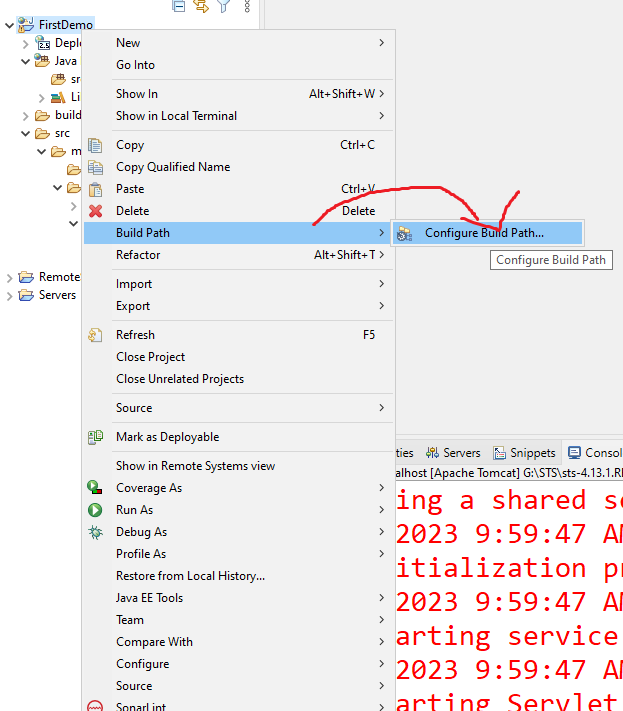


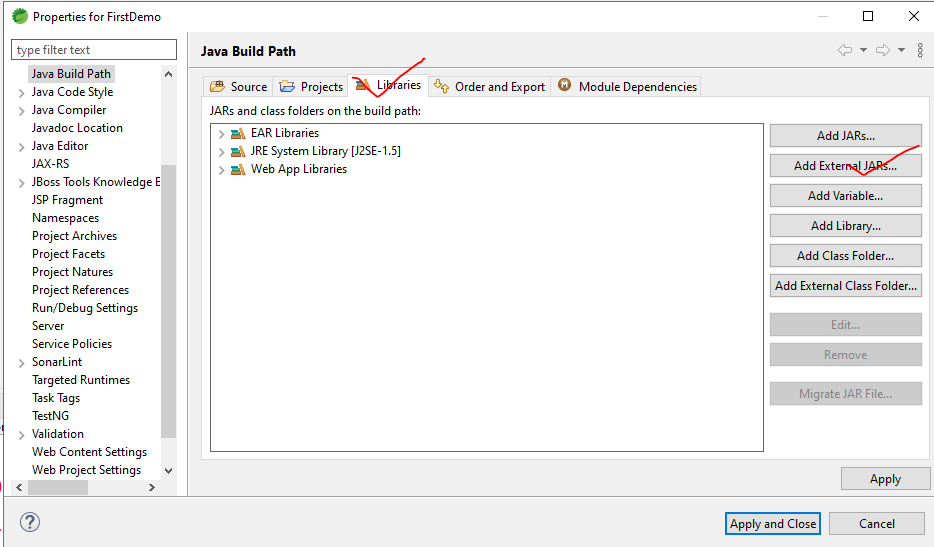


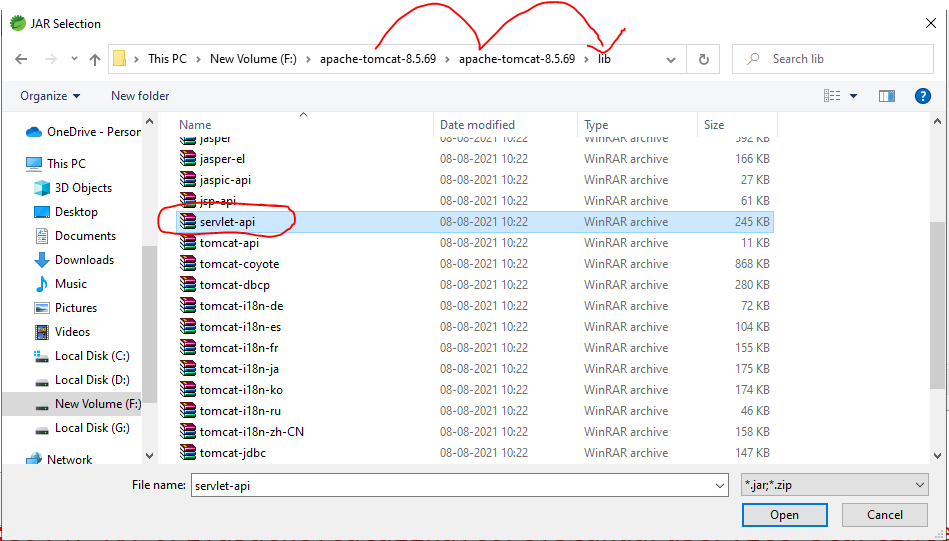


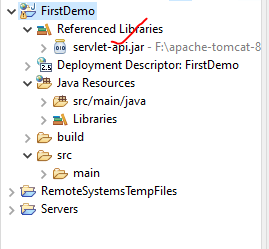
Steps Create a project and run it on Tomcat.

1.right click ->build path



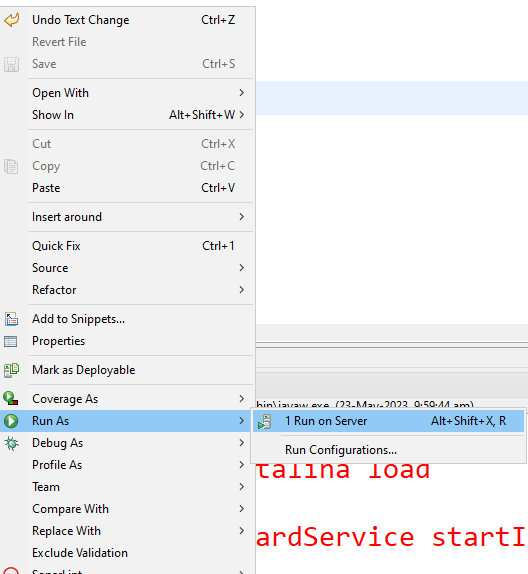


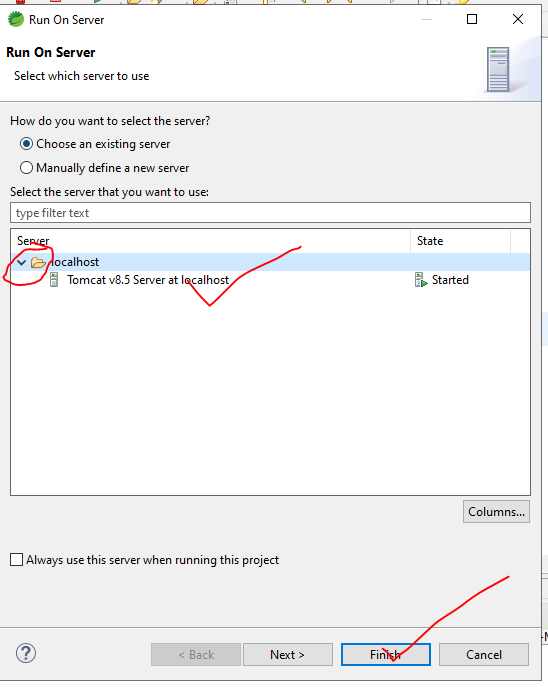


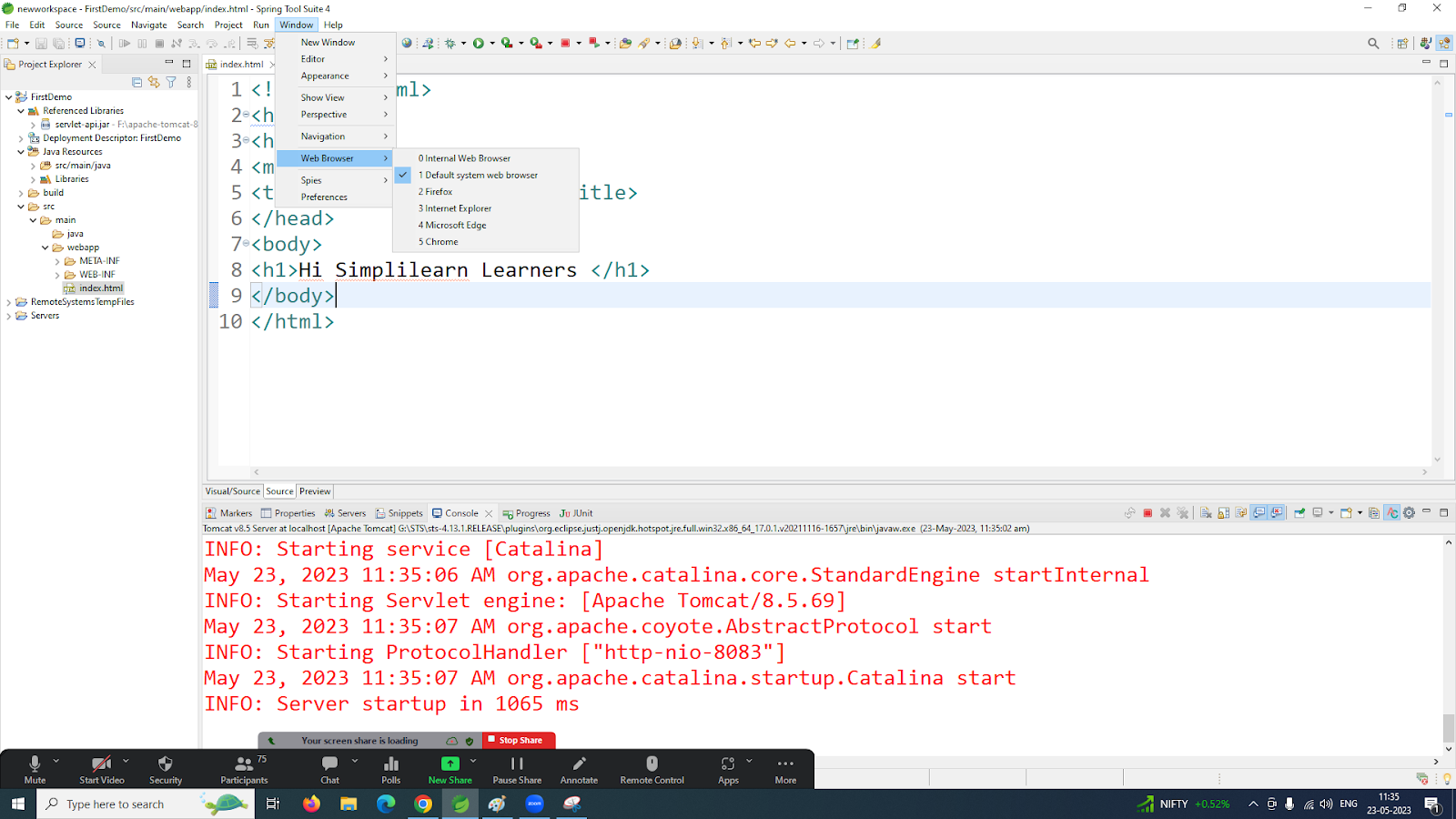


How to load a web page on the server?

Right click -> new file -> other / HTML







<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1>Hi Simplilearn Learners </h1>

</body>

</html>

How to create a servlet- java code ⇒java resource

In web.xml if we get any errors do replace the code with the  below line

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/j2ee" xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee ,http://java.sun.com/xml/ns/j2ee/web-app\_2\_5.xsd" id="WebApp\_ID" version="2.5">

In web.xml

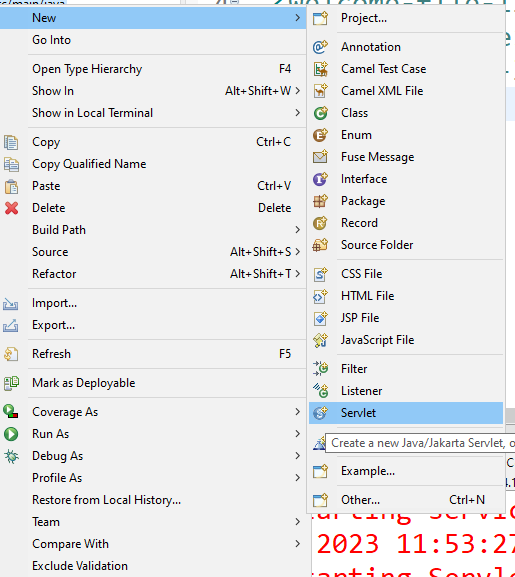
  <welcome-file-list>

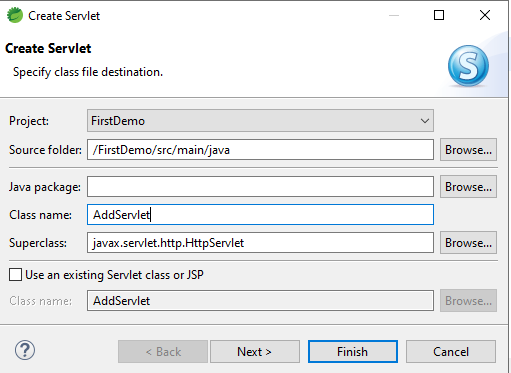
    <welcome-file>index.html</welcome-file>

  </welcome-file-list>

-Used to config the first base page that needs to be loaded onto the server.

>create a servlet

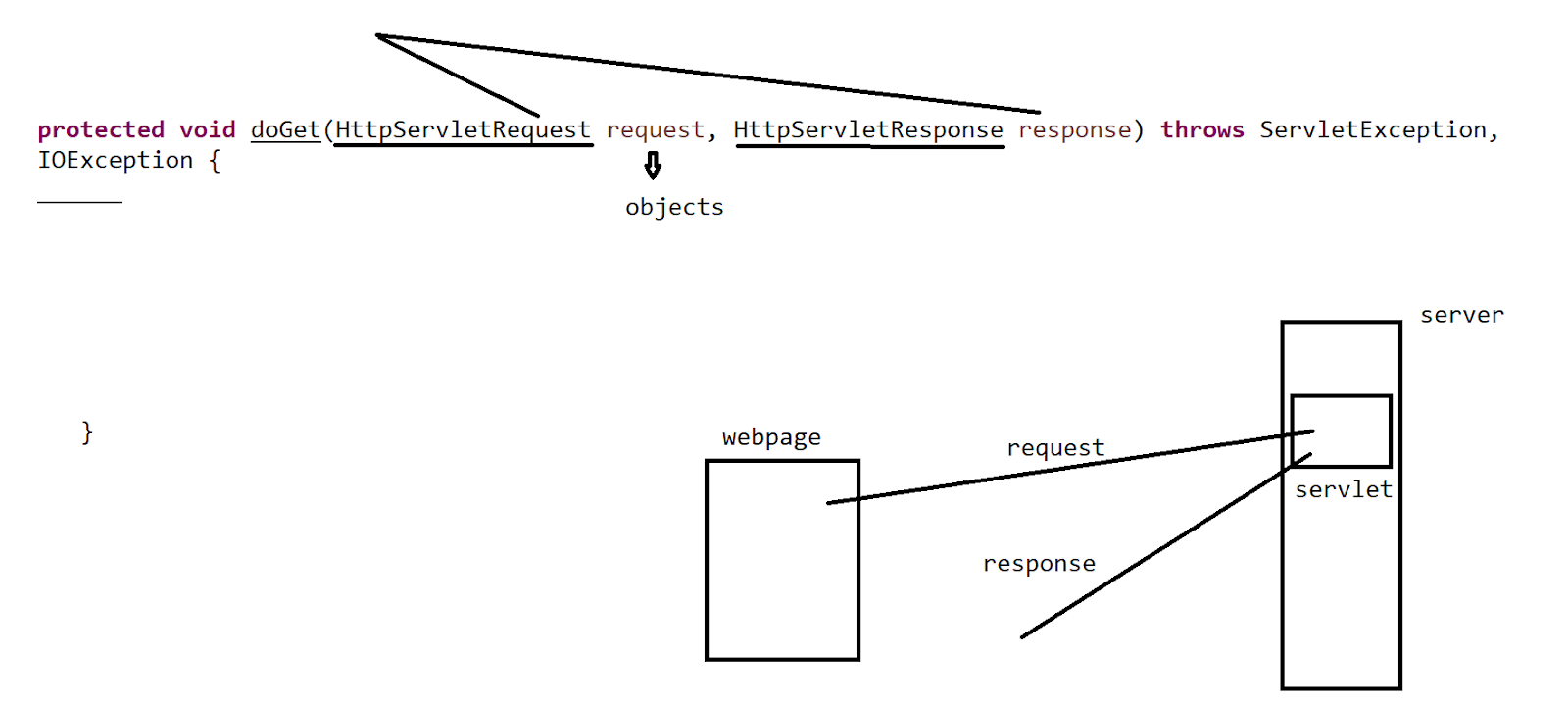




Note :

<form action="AddServlet">

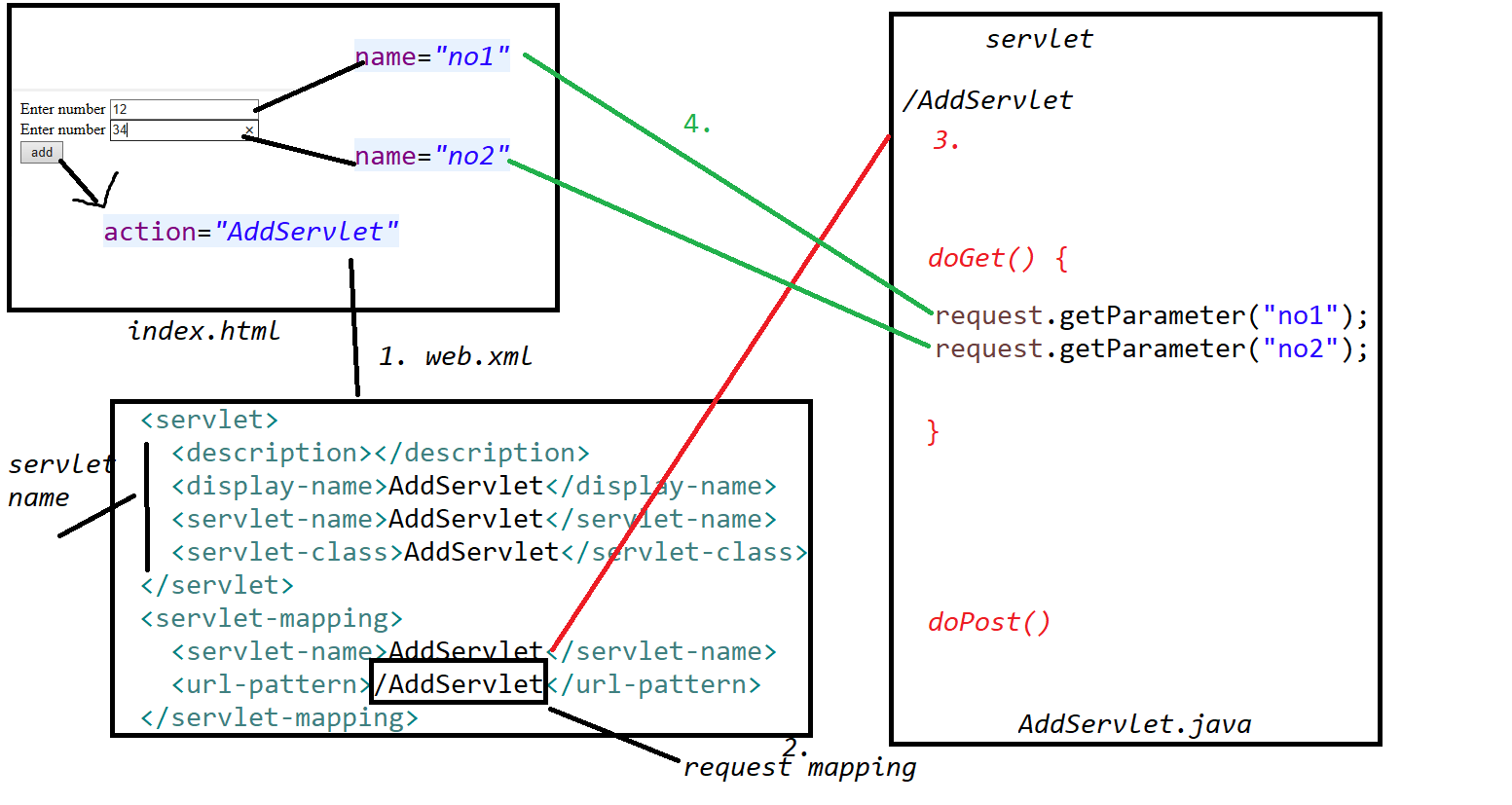
If the method is not mentioned in the form by default it goes to doGet() in the servlet



HttpServletResponse response

Response works in 2 ways

1. Virtual page
2. Physical page



<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="AddServlet">

Enter number <input type="text" name="no1"><br>

Enter number <input type="text" name="no2"><br>

<input type="submit" value="add">

</form>

</body>

</html>

//AddServlet

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class AddServlet

 \*/

public class AddServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public AddServlet() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

    //String

int no1=Integer.parseInt(request.getParameter("no1"));

int no2=Integer.parseInt(request.getParameter("no2"));

PrintWriter pw=response.getWriter();  //this class is used to generate a virtual response

pw.print("the output of 2 numbers is " +(no1+no2));

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

doGet(request, response);

}

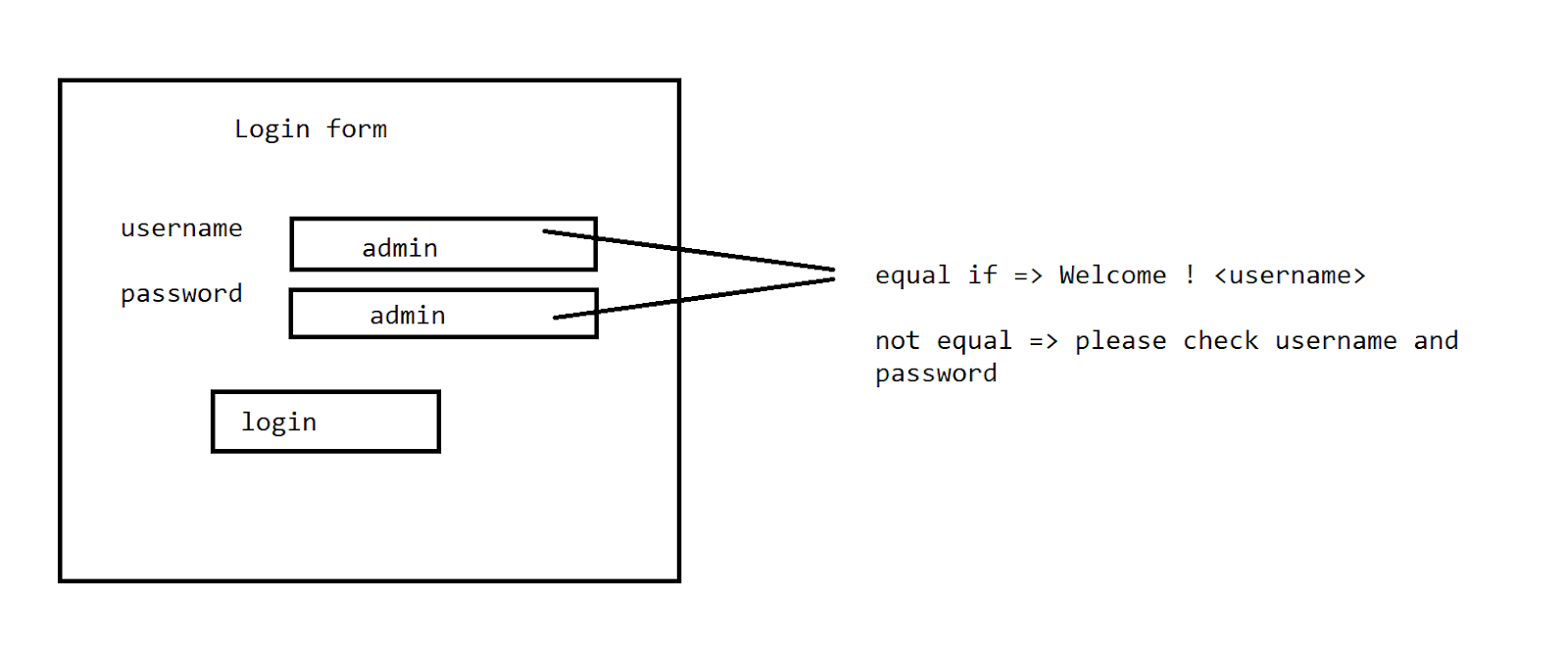
}

Get Vs Post





Task -10 min

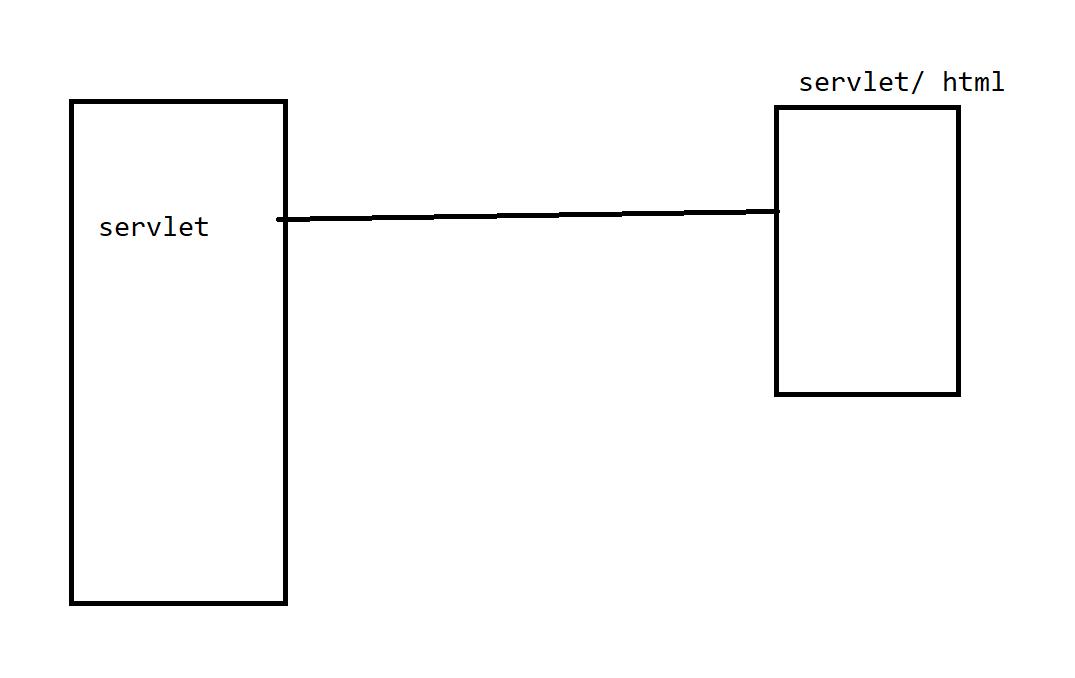


Practice: -Home assignment

Take input of a number a find whether it is a palindrome or not => via virtual response

<http://localhost/> ⇒ local server

// Servlet Collaboration



1. response.sendRedirect(“”);  physical servlet or physical page

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="AddServlet" method="post">

Enter Username <input type="text" name="user"><br>

Enter Password <input type="password" name="pwd"><br>

<input type="submit" value="login">

</form>

</body>

</html>

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class AddServlet

 \*/

public class AddServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public AddServlet() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

    //String

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

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

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

if(user.equals(pwd)) {

response.sendRedirect("success.html");

}

else {

response.sendRedirect("fail.html");

}

}

}

//success.html

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1>Logged in successfully...</h1>

</body>

</html>

//fail.html

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

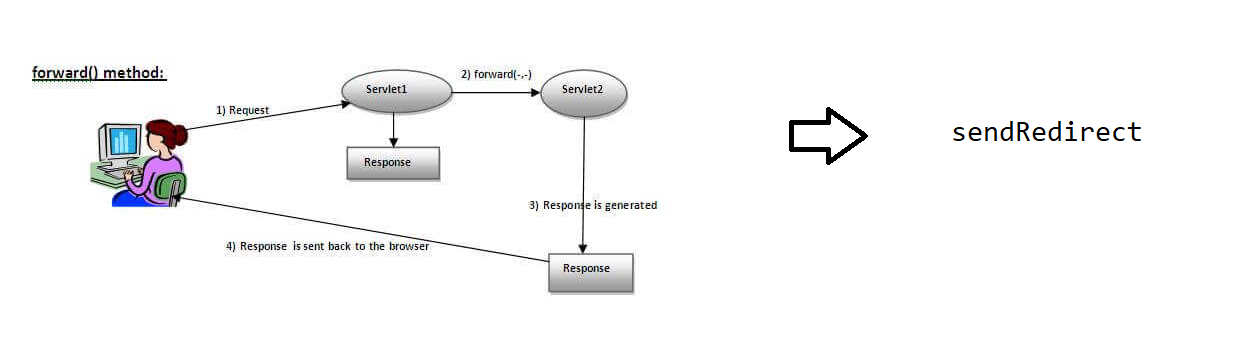
<h1>login failed ...</h1>

</body>

</html>

>Request Dispatcher

1. forward()
2. include()



Case 1 :

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

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

PrintWriter pw=response.getWriter();

pw.print(user+"  "+pwd);

RequestDispatcher rd=request.getRequestDispatcher("success.html");

rd.forward(request, response);

In case 1 the virtual response is generated by the servlet itself and control lies with the servlet only that's the reason we are able to go to the next step

Case 2: String user=request.getParameter("user");

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

RequestDispatcher rd=request.getRequestDispatcher("success.html");

rd.forward(request, response);

PrintWriter pw=response.getWriter();

pw.print(user+"  "+pwd);

In case 2 the flow of control already went to the success.html so it will not return back . so the flow in the servlet gets terminated.

//servlet code

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class AddServlet

 \*/

public class AddServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public AddServlet() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

    //String

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

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

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

PrintWriter pw=response.getWriter();

pw.print(user+"  "+pwd);

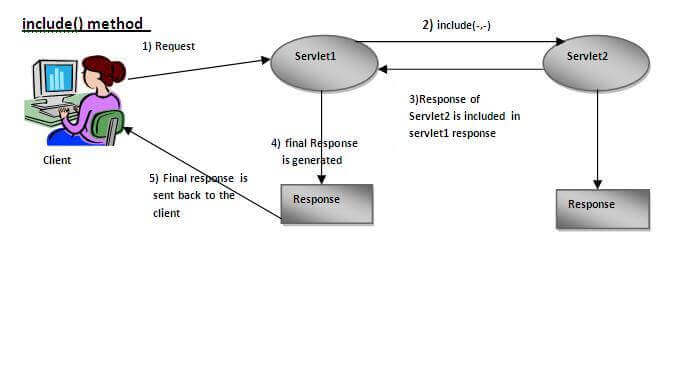
RequestDispatcher rd=request.getRequestDispatcher("success.html");

rd.forward(request, response);

}

}

//include()



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

response.setContentType("text/html");

RequestDispatcher rd=request.getRequestDispatcher("success.html");

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

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

PrintWriter pw=response.getWriter();

pw.print(user+"  "+pwd);

rd.include(request, response);

}

Servlet -> servlet

//AddServlet

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

response.setContentType("text/html");

RequestDispatcher rd=request.getRequestDispatcher("AddServlet2");

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

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

PrintWriter pw=response.getWriter();

pw.print(user+"  "+pwd);

rd.include(request, response);

}

//AddServlet2

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

PrintWriter pw=response.getWriter();

pw.print("hi welcome " +request.getParameter("user"));

}

Note:

In case if the web.xml updations are not happening with the servlet name then use an annotation

@WebServlet(“/urlpattern”)

public class AddServlet2 extends HttpServlet

Urlpattern need to be same as the action in the form

Session management:

Session is an identity.



Persistent (store the info over the browser)

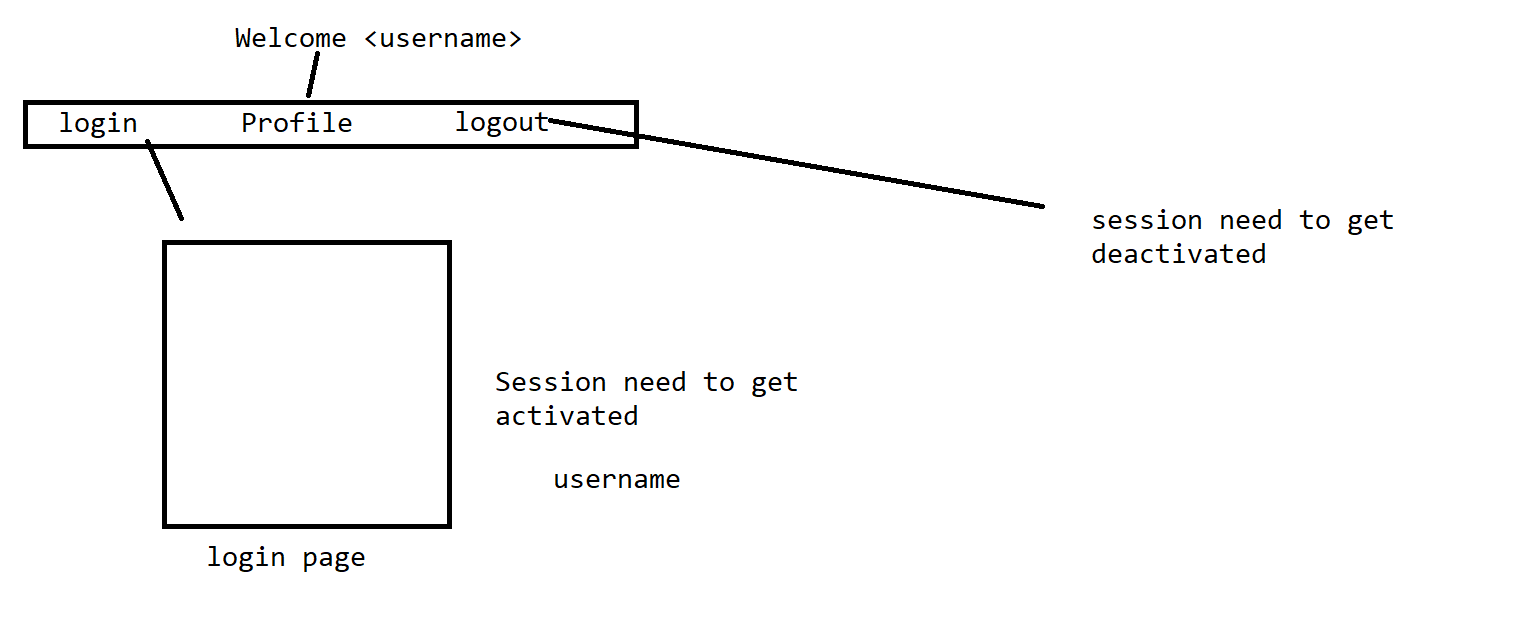
>Cookie

>HttpSession

NonPersistent (not store any info)

>URLRewriting

>Hidden Form Field



>Cookie

Step 1: add the servlet-api.jar

Syntax

Add a cookie

Cookie c=new Cookie("alias name over the server",”original value that need to be passed ”);

//index.html

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<a href="login.html">Login</a>

<a href="./Profile">Profile</a>

<a href="./Logout">Logout</a>

</body>

</html>

//login.html

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1>Login Form </h1>

<form action="LoginServlet" method="post">

username<input type="text" name="user"><br>

password<input type="password" name="pwd"><br>

<input type="submit" value="login">

</form>

</body>

</html>

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.Cookie;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class LoginServlet

 \*/

//@WebServlet("/LoginServlet")

public class LoginServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public LoginServlet() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

response.getWriter().append("Served at: ").append(request.getContextPath());

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

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

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

PrintWriter pw=response.getWriter();

response.setContentType("text/html");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);

if(user.equals(pwd)) {

Cookie c=new Cookie("userid",user);

response.addCookie(c);

pw.print("login is successfull..");

}

else {

pw.print("check the login ....");

}

}

}

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.Cookie;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class Logout

 \*/

public class Logout extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public Logout() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

PrintWriter pw=response.getWriter();

response.setContentType("text/html");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);

Cookie c=new Cookie("userid","");

c.setMaxAge(0);

response.addCookie(c);

pw.print("Logged out .....");

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

doGet(request, response);

}

}

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.Cookie;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class Profile

 \*/

public class Profile extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public Profile() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

PrintWriter pw=response.getWriter();

response.setContentType("text/html");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);

Cookie c[]=request.getCookies();

if(c!=null) {

pw.print("Welcome to "+ c[0].getValue());

}

else {

pw.print("hey login first...");

}

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

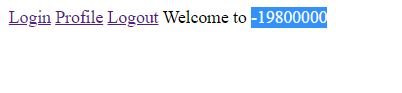
// TODO Auto-generated method stub

doGet(request, response);

}

}

On chrome if u get the below issue change the browser





//HttpSession

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.Cookie;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

/\*\*

 \* Servlet implementation class LoginServlet

 \*/

//@WebServlet("/LoginServlet")

public class LoginServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public LoginServlet() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

response.getWriter().append("Served at: ").append(request.getContextPath());

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

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

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

PrintWriter pw=response.getWriter();

response.setContentType("text/html");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);

if(user.equals(pwd)) {

HttpSession session=request.getSession();

session.setAttribute("userid", user);

pw.print("login is successfull..");

}

else {

pw.print("check the login ....");

}

}

}

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.Cookie;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

/\*\*

 \* Servlet implementation class Logout

 \*/

public class Logout extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public Logout() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

PrintWriter pw=response.getWriter();

response.setContentType("text/html");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);

HttpSession session=request.getSession();

session.invalidate();

pw.print("Logged out .....");

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

doGet(request, response);

}

}

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.Cookie;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

/\*\*

 \* Servlet implementation class Profile

 \*/

public class Profile extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public Profile() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

PrintWriter pw=response.getWriter();

response.setContentType("text/html");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);

HttpSession session=request.getSession();

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

if(userid!=null) {

pw.print("Welcome to "+ userid);

}

else {

pw.print("hey login first...");

}

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

doGet(request, response);

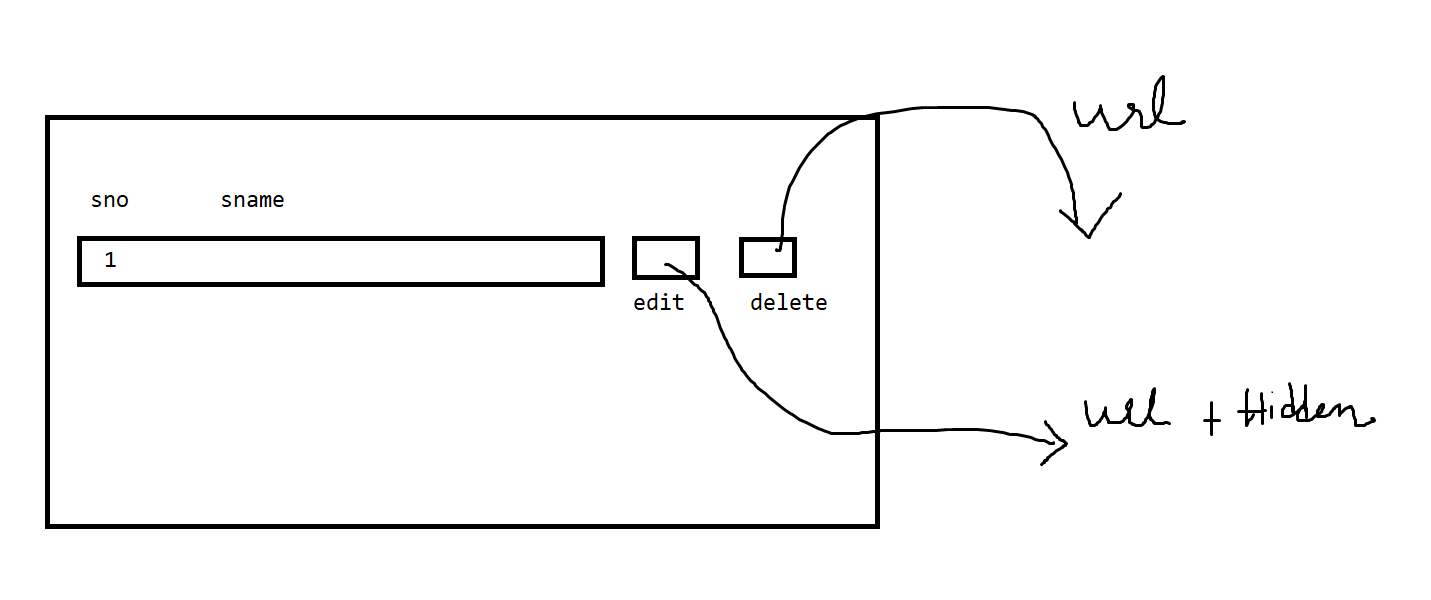
}

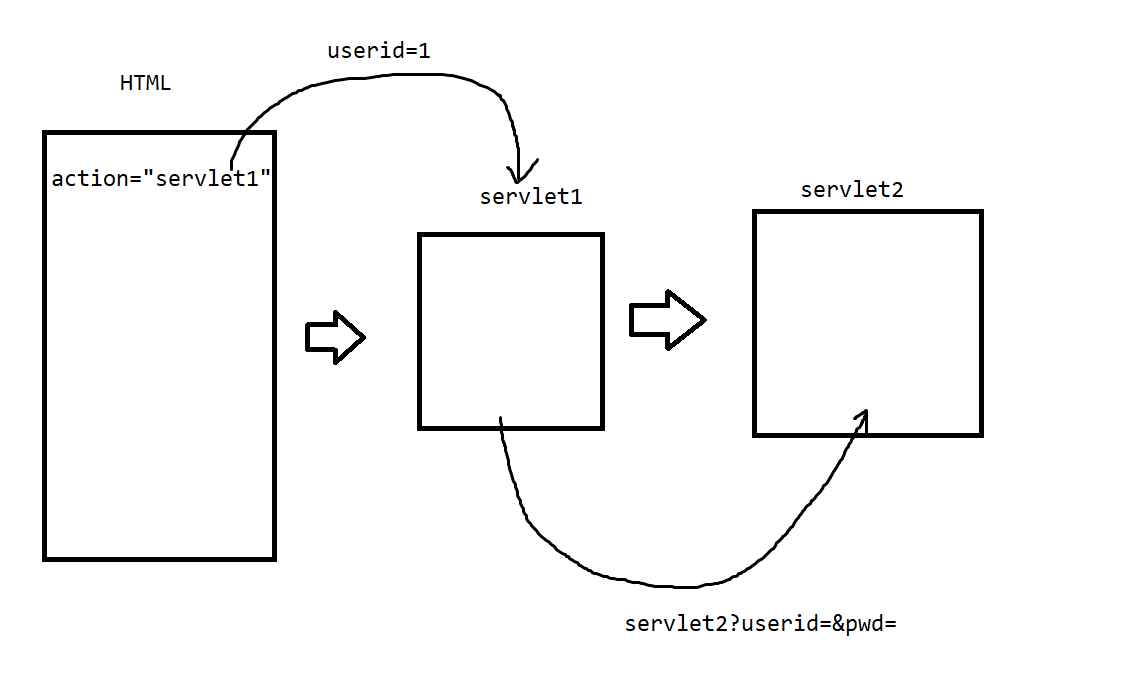
}

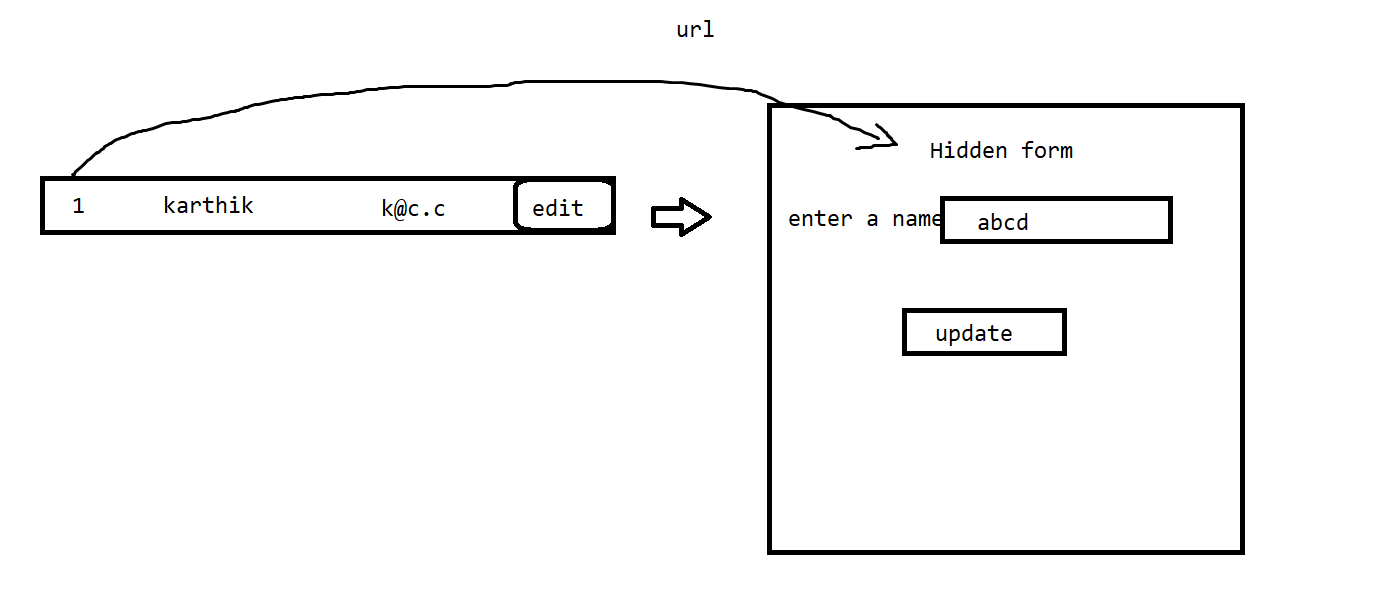
//Nonpersistent Session Management

URL Rewriting

Hidden form field







<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="Servlet1" method="post">

Userid<input type="text" name="userid"><br>

<input type="submit" value="go">

</form>

</body>

</html>

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class Servlet1

 \*/

public class Servlet1 extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* Default constructor.

     \*/

    public Servlet1() {

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

response.getWriter().append("Served at: ").append(request.getContextPath());

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

String user=request.getParameter("userid");

response.sendRedirect("Servlet2?user="+user);

}

}

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class Servlet2

 \*/

public class Servlet2 extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public Servlet2() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

PrintWriter pw=response.getWriter();

pw.print("welcome to " + request.getParameter("user"));

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

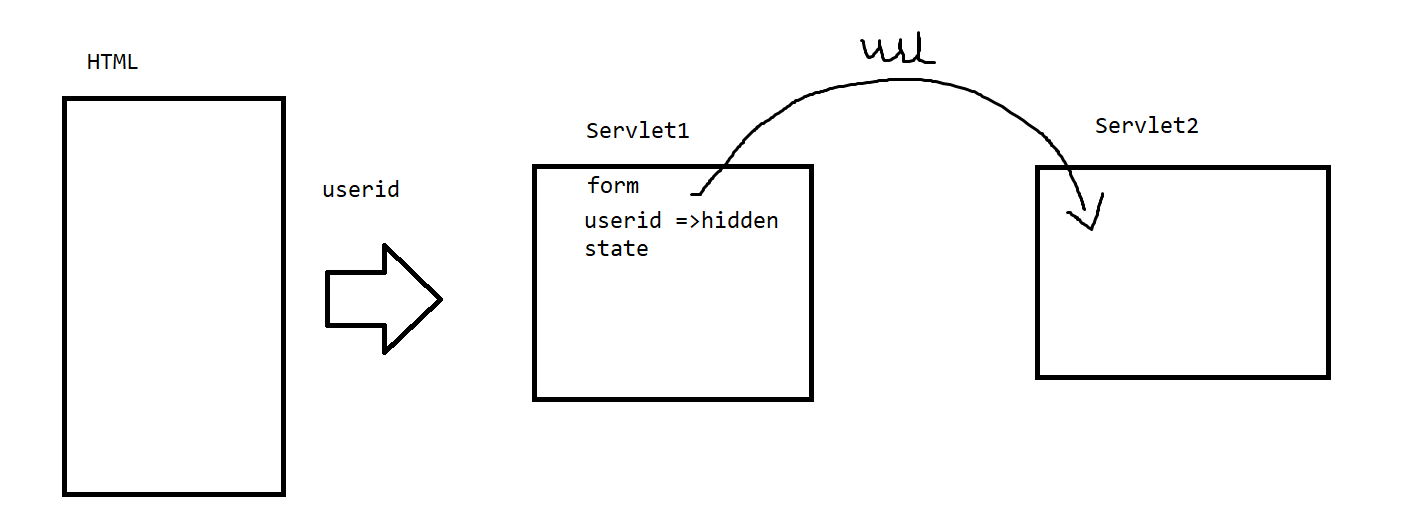
// TODO Auto-generated method stub

doGet(request, response);

}

}

//Hidden form field



<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="Servlet1" method="post">

Userid<input type="text" name="userid"><br>

<input type="submit" value="go">

</form>

</body>

</html>

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class Servlet1

 \*/

public class Servlet1 extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* Default constructor.

     \*/

    public Servlet1() {

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

response.getWriter().append("Served at: ").append(request.getContextPath());

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

String user=request.getParameter("userid");

PrintWriter pw=response.getWriter();

response.setContentType("text/html");

//pw is not only taken for a output based object but also we can write a HTML as an output over here

pw.print("<HTML><body><form action='Servlet2'>"

+ "<input type='hidden' name='user' value='"+user+"'>"

+ "<input type='submit'>"

+ "</form></body></HTML>");

}

}

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class Servlet2

 \*/

public class Servlet2 extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public Servlet2() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

PrintWriter pw=response.getWriter();

pw.print("welcome to " + request.getParameter("user"));

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

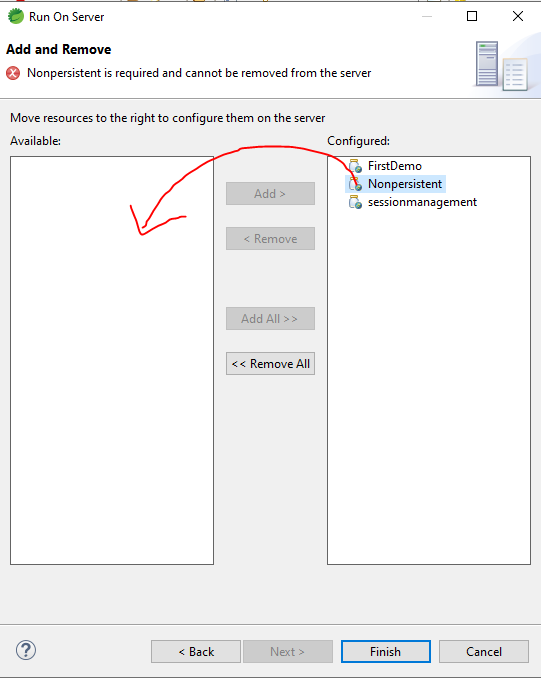
doGet(request, response);

}

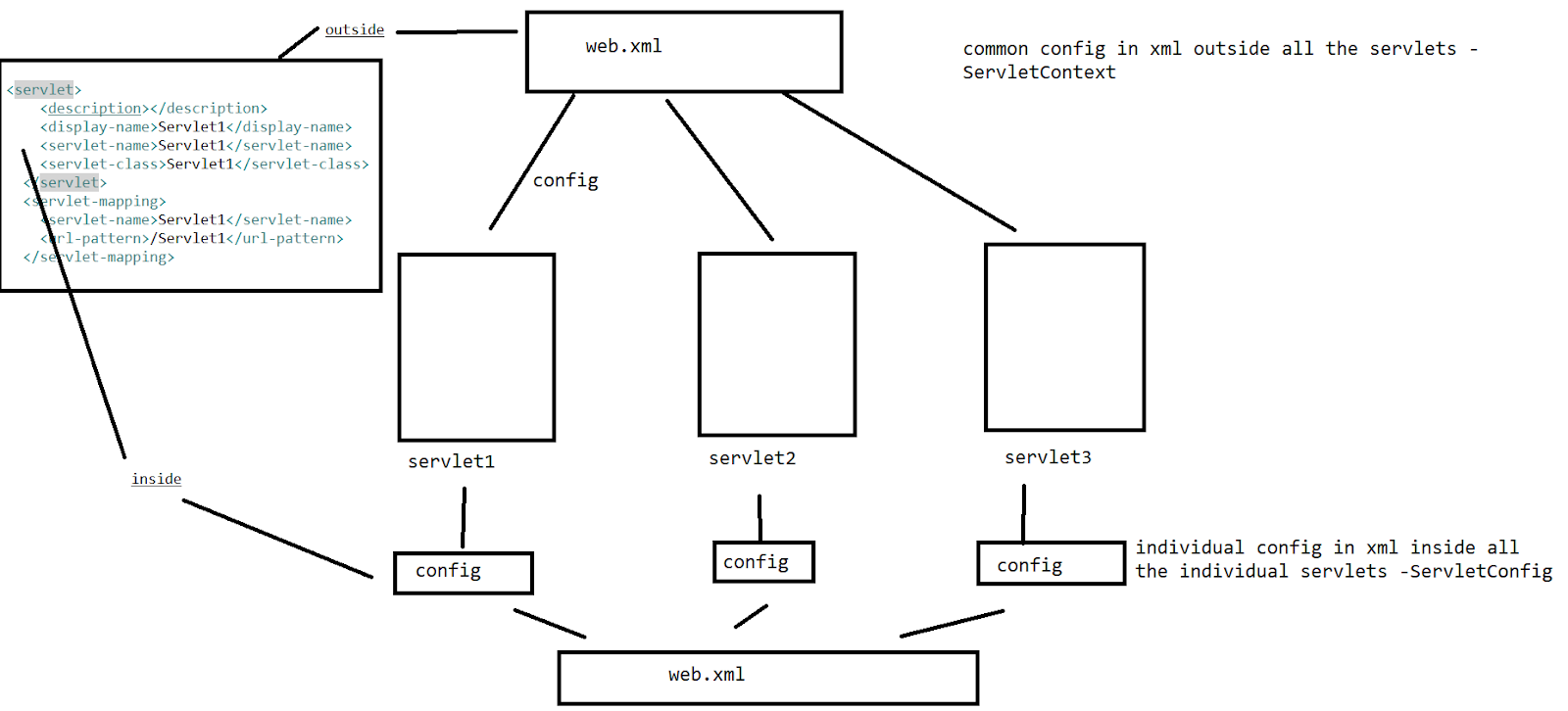
}

Tomcat server is failed to start

1. Servlet code has issues
2. Web.xml has issues
3. On the sever we have  a project which is having issue .



Servlet Context and Servlet Config



web.xml on this we are going to config out details

Web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:web="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee http://java.sun.com/xml/ns/j2ee, http://java.sun.com/xml/ns/j2ee/web-app\_2\_5.xsd" id="WebApp\_ID" version="2.5">

  <display-name>ContextConfigEx</display-name>

  <welcome-file-list>

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

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

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

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

    <welcome-file>default.jsp</welcome-file>

    <welcome-file>default.htm</welcome-file>

  </welcome-file-list>

  <context-param>

  <param-name>url</param-name>

  <param-value>jdbc:mysql://localhost:3306/sonoo</param-value>

  </context-param>

  <servlet>

    <description></description>

    <display-name>Servlet1</display-name>

    <servlet-name>Servlet1</servlet-name>

    <servlet-class>Servlet1</servlet-class>

  </servlet>

  <servlet-mapping>

    <servlet-name>Servlet1</servlet-name>

    <url-pattern>/Servlet1</url-pattern>

  </servlet-mapping>

  <servlet>

    <description></description>

    <display-name>Servlet2</display-name>

    <servlet-name>Servlet2</servlet-name>

    <servlet-class>Servlet2</servlet-class>

    <init-param>

      <param-name>password</param-name>

   <param-value>123456</param-value>

    </init-param>

  </servlet>

  <servlet-mapping>

    <servlet-name>Servlet2</servlet-name>

    <url-pattern>/Servlet2</url-pattern>

  </servlet-mapping>

</web-app>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="Servlet1">

<input type="submit" value="servlet1">

</form>

<form action="Servlet2">

<input type="submit" value="servlet2">

</form>

</body>

</html>

import java.io.IOException;

import java.io.PrintWriter;

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;

/\*\*

 \* Servlet implementation class Servlet1

 \*/

public class Servlet1 extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* Default constructor.

     \*/

    public Servlet1() {

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

PrintWriter pw=response.getWriter();

ServletContext context=getServletContext();

String value=context.getInitParameter("url");

pw.println("in servlet 1 context"+value);

ServletConfig config=getServletConfig();

pw.println("in servlet 1 config"+config.getInitParameter("password"));

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

doGet(request, response);

}

}

import java.io.IOException;

import java.io.PrintWriter;

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;

/\*\*

 \* Servlet implementation class Servlet2

 \*/

public class Servlet2 extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public Servlet2() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

PrintWriter pw=response.getWriter();

ServletContext context=getServletContext();

String value=context.getInitParameter("url");

pw.println("in servlet 2 context"+value);

ServletConfig config=getServletConfig();

pw.println("in servlet 2 config"+config.getInitParameter("password"));

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

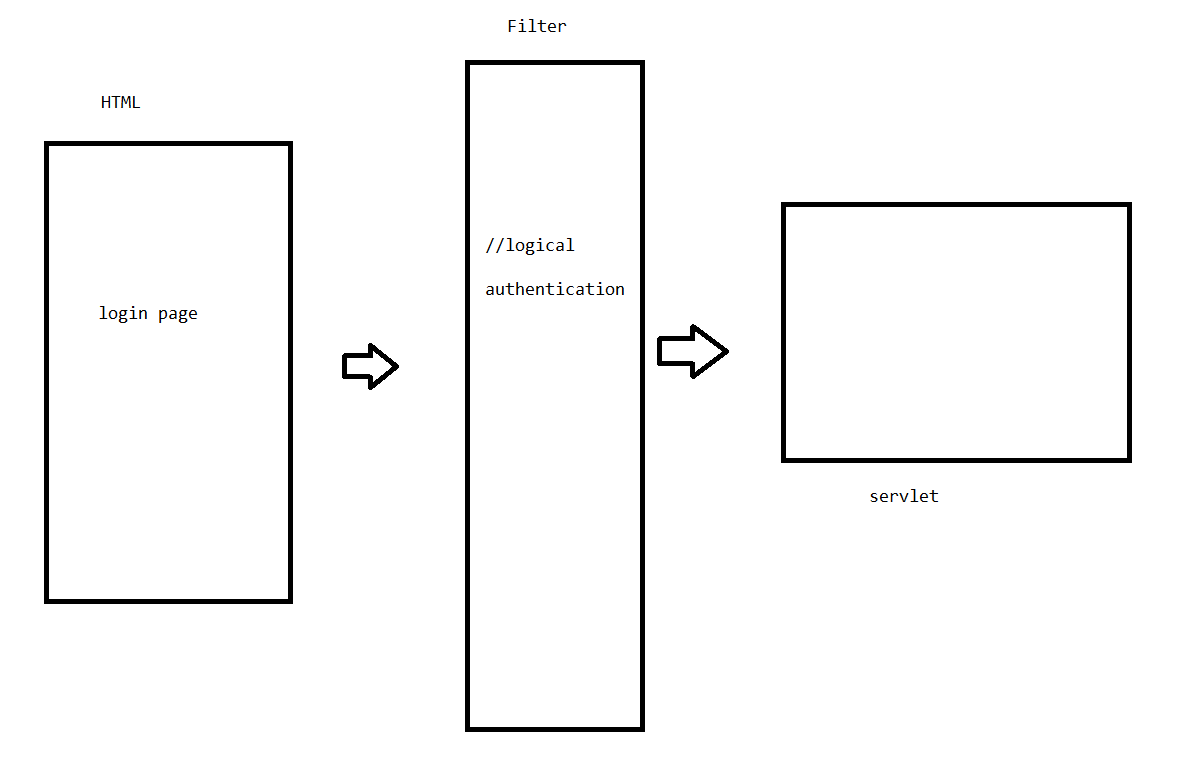
// TODO Auto-generated method stub

doGet(request, response);

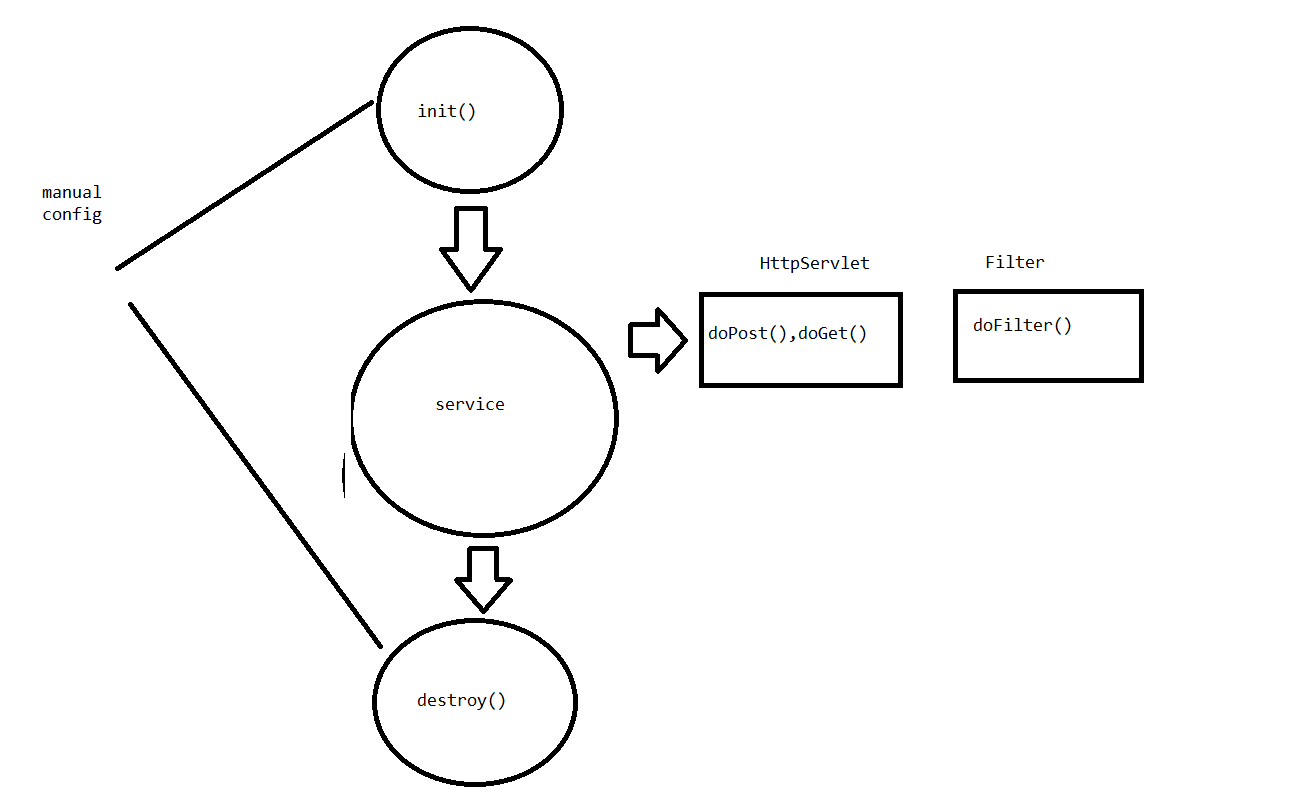
}

}

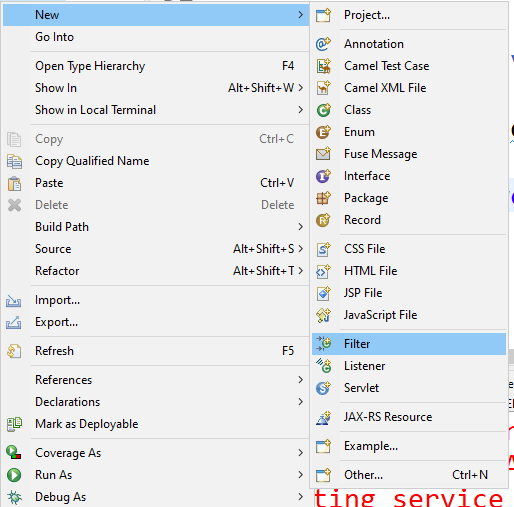
>Filter

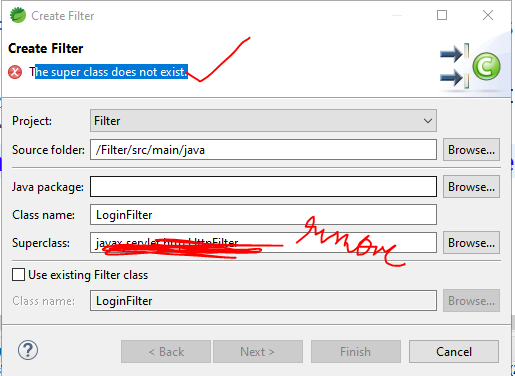


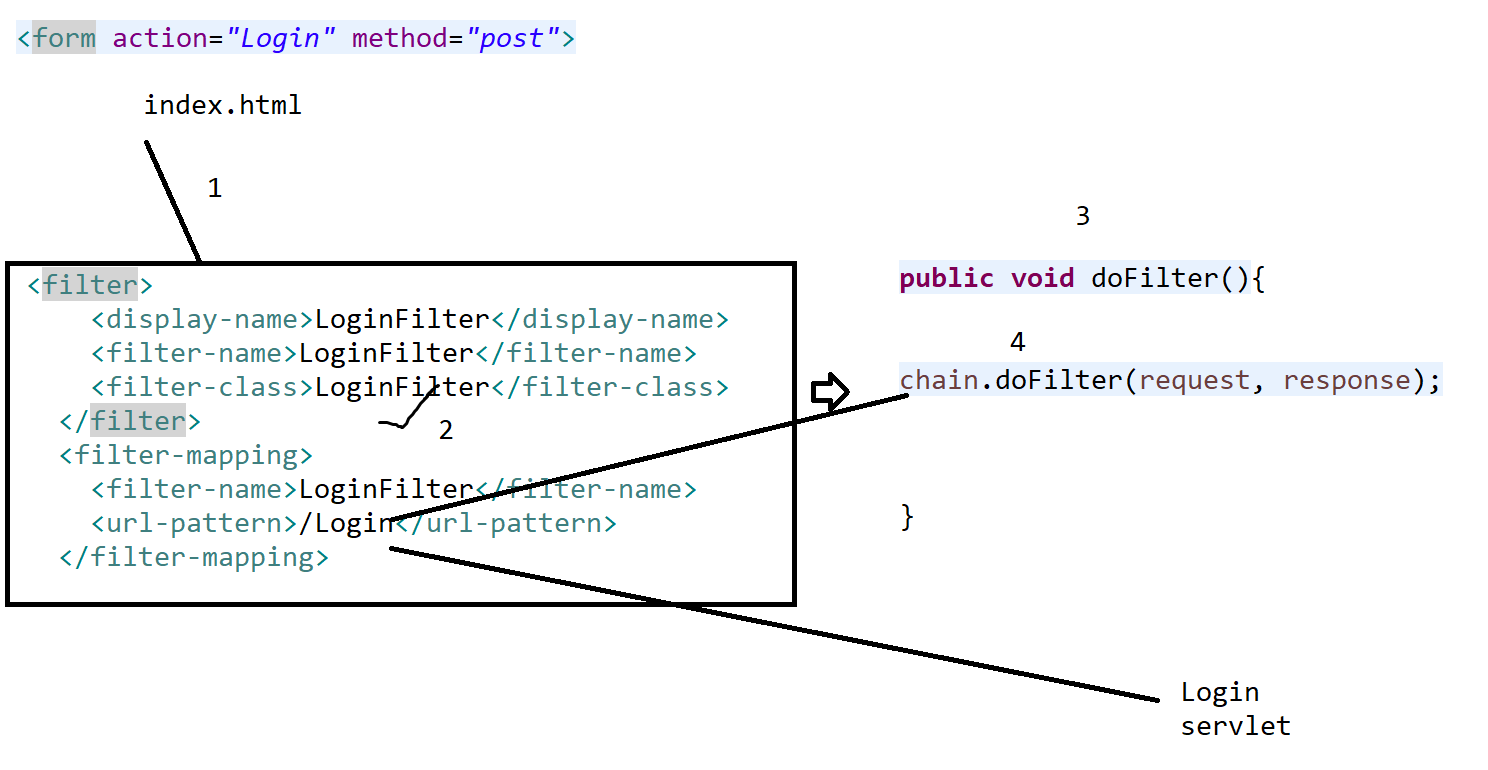
Lifecycle of a servlet



How to create a filter?







<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="Login" method="post">

username <input type="text" name="user"><br>

password <input type="password" name="pwd"><br>

<input type="submit">

</form>

</body>

</html>

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

/\*\*

 \* Servlet Filter implementation class LoginFilter

 \*/

public class LoginFilter implements Filter {

    /\*\*

     \* Default constructor.

     \*/

    public LoginFilter() {

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see Filter#destroy()

\*/

public void destroy() {

// TODO Auto-generated method stub

}

/\*\*

\* @see Filter#doFilter(ServletRequest, ServletResponse, FilterChain)

\*/

public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws IOException, ServletException {

// TODO Auto-generated method stub

// place your code here

PrintWriter pw=response.getWriter();

if(request.getParameter("user").equals(request.getParameter("pwd"))) {

// pass the request along the filter chain

chain.doFilter(request, response);

}

else {

pw.print("check the credentials ");

}

}

/\*\*

\* @see Filter#init(FilterConfig)

\*/

public void init(FilterConfig fConfig) throws ServletException {

// TODO Auto-generated method stub

}

}

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class Login

 \*/

public class Login extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public Login() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

response.getWriter().append("Served at: ").append(request.getContextPath());

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

PrintWriter pw=response.getWriter();

pw.print("welcome to "+request.getParameter("user"));

}

}

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:web="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee, http://java.sun.com/xml/ns/j2ee/web-app\_2\_5.xsd http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd" id="WebApp\_ID" version="2.5">

  <display-name>Filter</display-name>

  <welcome-file-list>

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

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

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

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

    <welcome-file>default.jsp</welcome-file>

    <welcome-file>default.htm</welcome-file>

  </welcome-file-list>

  <servlet>

    <description></description>

    <display-name>Login</display-name>

    <servlet-name>Login</servlet-name>

    <servlet-class>Login</servlet-class>

  </servlet>

  <servlet-mapping>

    <servlet-name>Login</servlet-name>

    <url-pattern>/Login</url-pattern>

  </servlet-mapping>

  <filter>

    <display-name>LoginFilter</display-name>

    <filter-name>LoginFilter</filter-name>

    <filter-class>LoginFilter</filter-class>

  </filter>

  <filter-mapping>

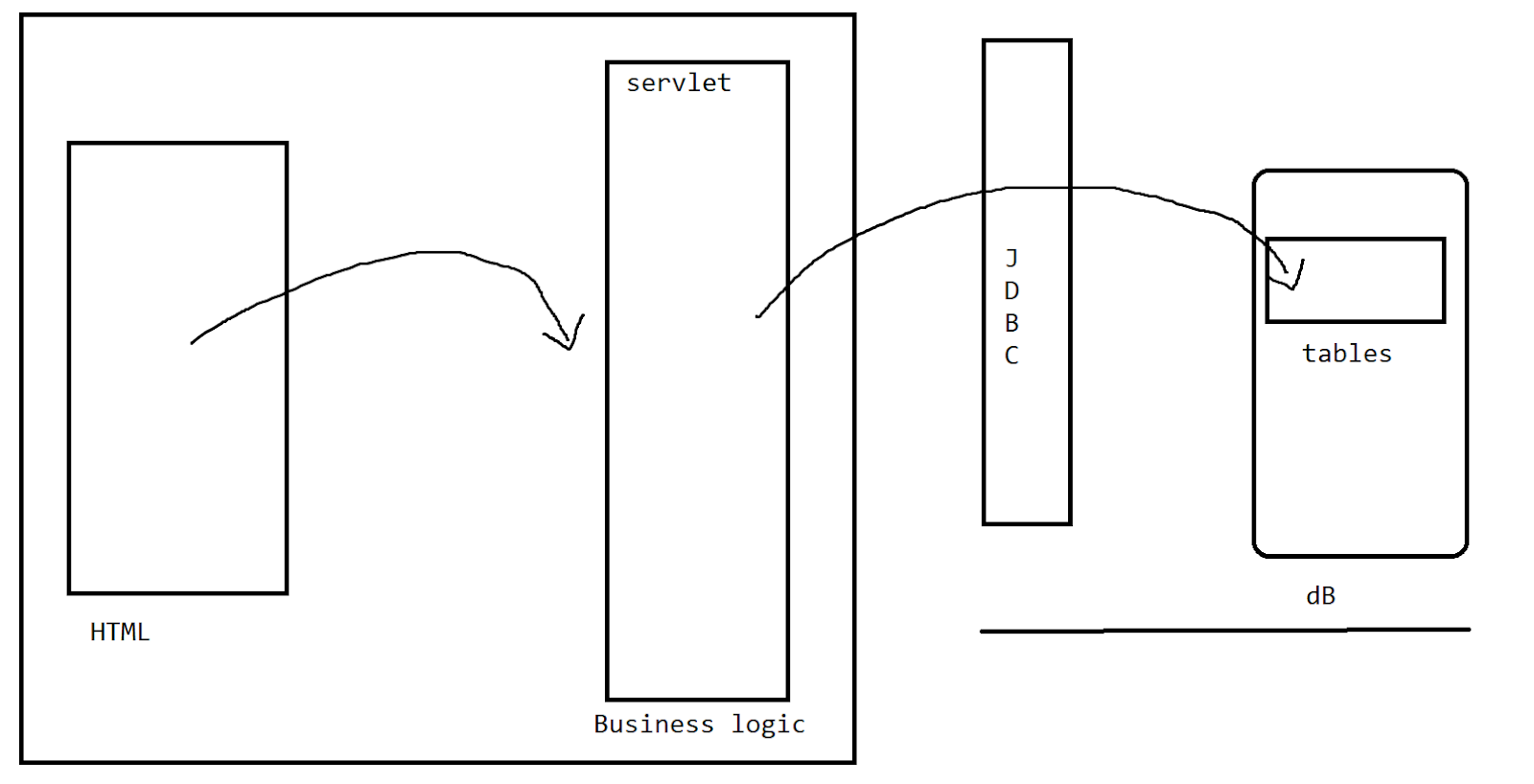
    <filter-name>LoginFilter</filter-name>

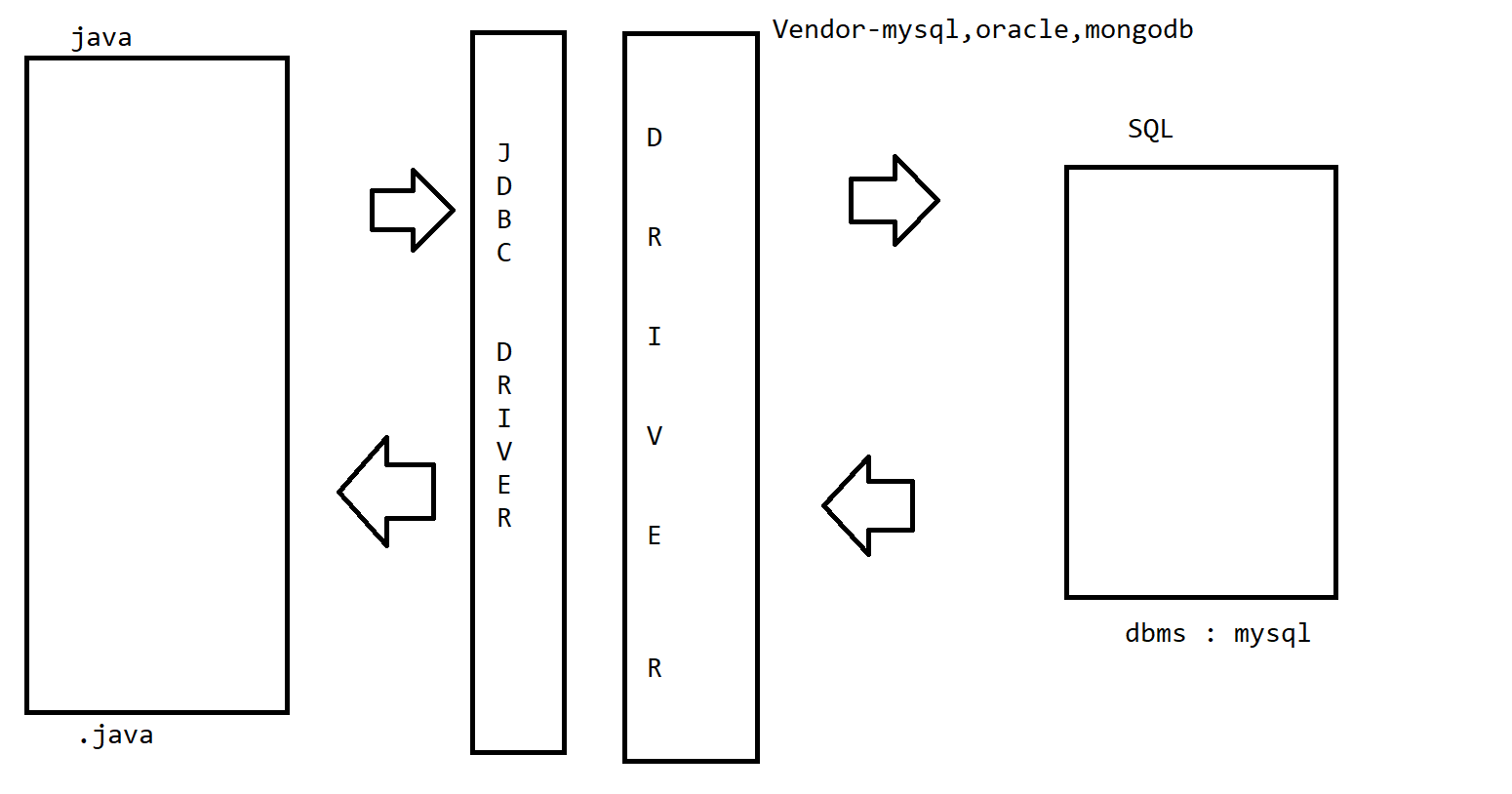
    <url-pattern>/Login</url-pattern>

  </filter-mapping>

</web-app>

JDBC-[Java database connectivity]





Drivers are the translators which are provided by the vendors.

Oracle 10g -<https://drive.google.com/file/d/1mjTnxqwI5m22XE1RDZtHv1IvrqXVH4US/view?usp=drive_link>

Types of drivers :

1.JDBC-ODBC bridge driver -Type1

2.NativeAPI driver-Type2

3.Network protocol driver-Type3

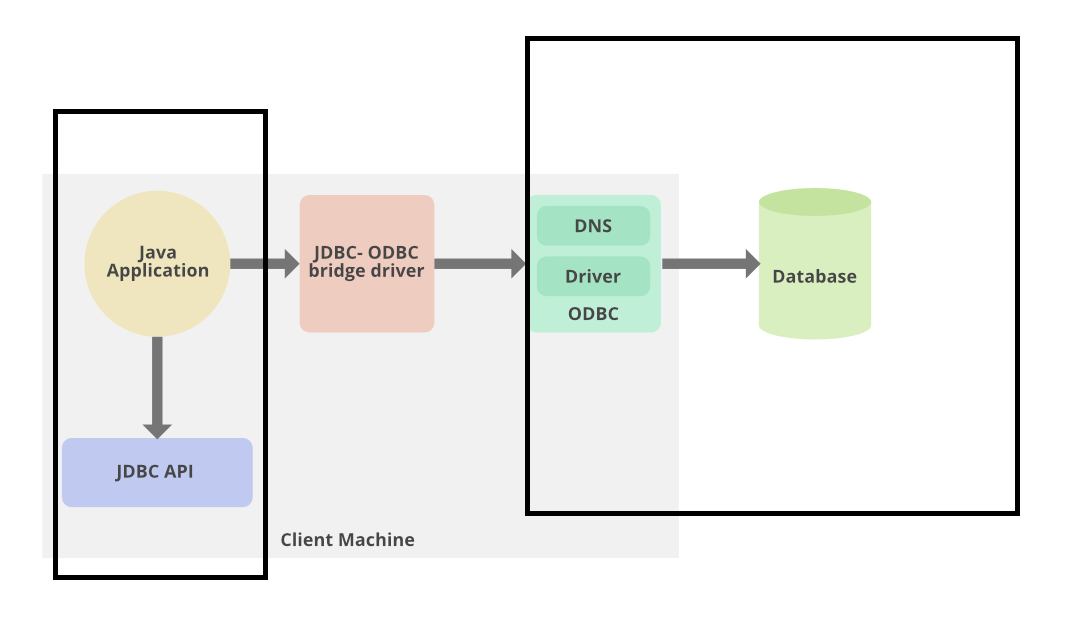
4.Thin driver-Type4  \*\*\*\*\*

When we install java on our machine we get the thin driver as the jdbc driver automatically on our machine . so no need of any external config of the drivers .

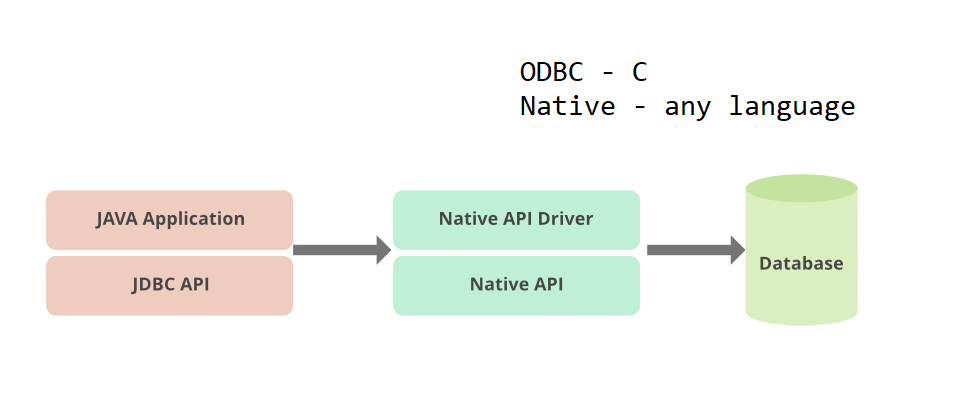
JDBCAPI=>Application program interface

//all the inbuilt classes,interfaces, methods

Type 1:



Type 2:

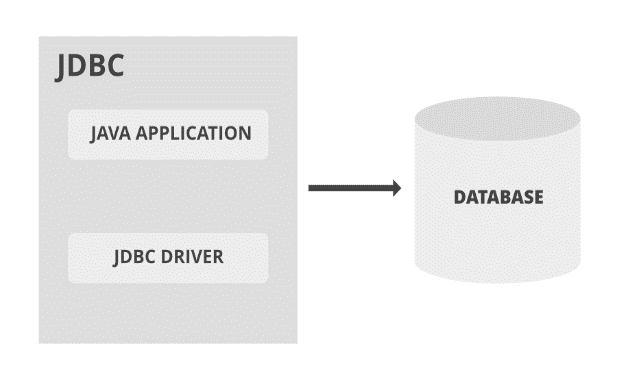


Type 1 , Type 2 can be configured only on the standalone machines .

 Type 3



Type 4:



JDBC Connection Steps :

1. Register Driver - vendor
2. Connection with the dB
3. Statement of SQL  -Statement , PreparedStatement
4. Execute the statement
5. Close the statement

Configurations :

1. Register Driver - vendor
2. Connection with the dB

DAO - data access object- CRUD

{

creation of data - insert ,

 reading / retrieving of data -select

Updating the data - update

Delete the data -delete

}

Steps in designing a JDBC APP

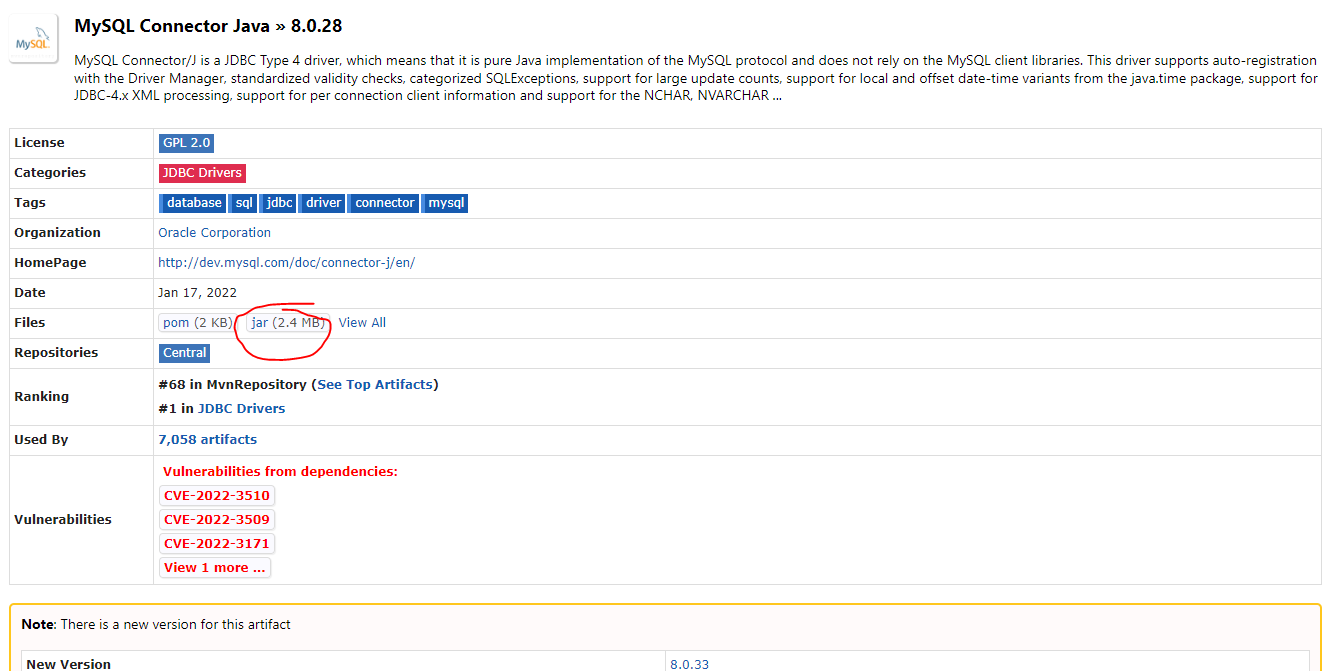
1. Bean / pojo - plain old java object

Class Employee{eid,ename,salary}=>private =>[getters and setters]=>table=>eid,ename,salary

1. Create a class dBUtil -config of Register Driver ,Connection with the dB
2. Create a DAO class - crud operation
3. Create a main to operate the crud operations of dao.

Vendor driver

<https://mvnrepository.com/artifact/mysql/mysql-connector-java/8.0.28>



->Load the above driver into the STS application by using build path

//Test the conn is established with dB

create database db1;

use db1;

package com.mphasis.dbUtil;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

//db Conn - 1. driver 2. url 3. username 4. password

public class dbUtil {

private static final String DRIVER\_CLASS="com.mysql.jdbc.Driver";

//by using jdbc driver - connect to mysql which is a local dB server on the port of 3306 which has db1 as dB

private static final String DB\_URL="jdbc:mysql://localhost:3306/db1";

private static final String USERNAME="root";

private static final String PASSWORD="123456";

public static Connection getConn() throws ClassNotFoundException, SQLException {

//register the driver

Class.forName(DRIVER\_CLASS);

//connection with the dB

Connection con=DriverManager.getConnection(DB\_URL,USERNAME,PASSWORD);

return con;

}

}

package com.mphasis.main;

import java.sql.Connection;

import java.sql.SQLException;

import com.mphasis.dbUtil.dbUtil;

public class dbMain {

public static void main(String[] args) throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection is successfully established ....!!!");

}

}

}

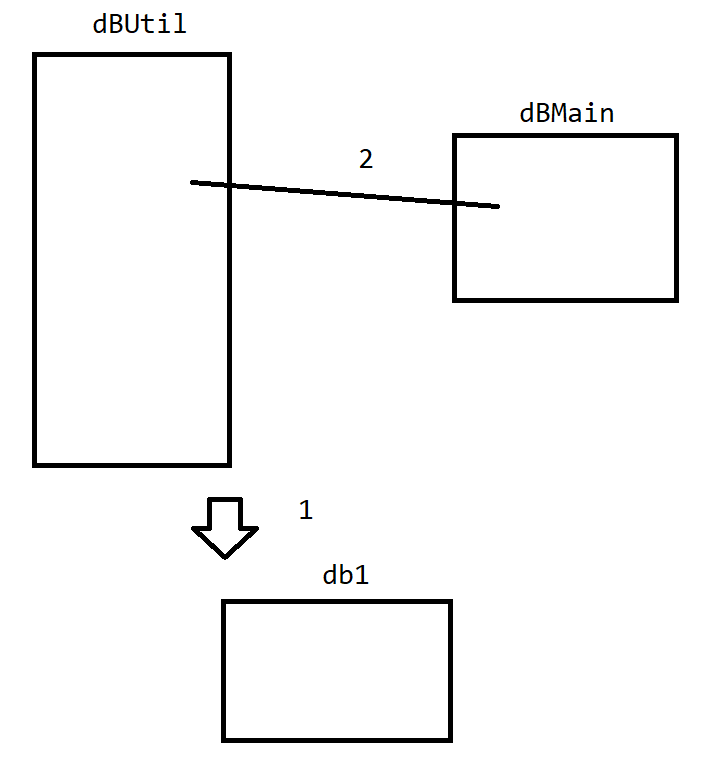


Table + pojo

create table product(pid int primary key, pname varchar(20),cost int);

package com.mphasis.pojo;

public class Product {

private int pid;

private String pname;

private int cost;

public int getPid() {

return pid;

}

public void setPid(int pid) {

this.pid = pid;

}

public String getPname() {

return pname;

}

public void setPname(String pname) {

this.pname = pname;

}

public int getCost() {

return cost;

}

public void setCost(int cost) {

this.cost = cost;

}

@Override

public String toString() {

return "Product [pid=" + pid + ", pname=" + pname + ", cost=" + cost + "]";

}

}

DAO

package com.mphasis.dao;

import java.sql.SQLException;

import java.util.List;

import com.mphasis.pojo.Product;

public interface ProductCrudInterface {

//this method is used to add a product

public int addProduct(Product product) throws ClassNotFoundException, SQLException;

//this method is used to delete a product

public int deleteProduct(int id) throws ClassNotFoundException, SQLException;

//this method is used to select the list of products

public List<Product> selectProducts() throws ClassNotFoundException, SQLException;

//update product name

public int updateProductName(int id, String name) throws ClassNotFoundException, SQLException;

}

package com.mphasis.dao;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.ArrayList;

import java.util.List;

import com.mphasis.dbUtil.dbUtil;

import com.mphasis.pojo.Product;

public class ProductCrudImpl implements ProductCrudInterface {

@Override

public int addProduct(Product product) throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection is successfully established ....!!!");

}

//preparing an sql statement

Statement st=con.createStatement();

String sql="insert into product values("+product.getPid()+","+"'"+product.getPname()+"',"+product.getCost()+")";

System.out.println(sql);

//insert, update ,delete -->return int

int rows=st.executeUpdate(sql);

return rows;

}

@Override

public int deleteProduct(int id) throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection is successfully established ....!!!");

}

Statement st=con.createStatement();

String sql="delete from product where pid="+id;

System.out.println(sql);

//insert, update ,delete -->return int

int rows=st.executeUpdate(sql);

return rows;

}

@Override

public List<Product> selectProducts() throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection is successfully established ....!!!");

}

Statement st=con.createStatement();

String sql="select \* from product";

//table address is stored in rs obj

ResultSet rs=st.executeQuery(sql);

List<Product> list=new ArrayList<Product>();

//row 1

while(rs.next()) {

Product product=new Product();

product.setPid(rs.getInt("pid"));

product.setPname(rs.getString("pname"));

product.setCost(rs.getInt("cost"));

list.add(product);

}

return list;

}

@Override

public int updateProductName(int id,String name) throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection is successfully established ....!!!");

}

Statement st=con.createStatement();

String sql="update product set pname='"+name+"' where pid="+id;

System.out.println(sql);

//insert, update ,delete -->return int

int rows=st.executeUpdate(sql);

return rows;

}

}

dbMain

package com.mphasis.main;

import java.sql.SQLException;

import java.util.List;

import java.util.Scanner;

import com.mphasis.dao.ProductCrudImpl;

import com.mphasis.pojo.Product;

public class dbMain {

public static void main(String[] args) throws ClassNotFoundException, SQLException {

Scanner sc=new Scanner(System.in);

ProductCrudImpl dao=null;

while(true) {

System.out.println("Menu 1. add product 2. delete product  3. update product 4.retrive product 5. exit");

System.out.println("enter choice");

int ch=sc.nextInt();

switch(ch) {

case 1:Product product=new Product();

System.out.println("enter product id");

product.setPid(sc.nextInt());

System.out.println("enter product name");

product.setPname(sc.next());

System.out.println("enter product cost");

product.setCost(sc.nextInt());

dao=new ProductCrudImpl();

if(dao.addProduct(product)>0) {

System.out.println("product added successfully...");

}

else {

System.out.println("product not added check once...");

}

break;

case 2:System.out.println("enter product id to be deleted");

int id=sc.nextInt();

dao=new ProductCrudImpl();

if(dao.deleteProduct(id)>0) {

System.out.println("product got deleted successfully with id:"+id+" ...");

}

else {

System.out.println("product not deleted check once...");

}

break;

case 3:System.out.println("enter product id to be updated with name");

int idupdate=sc.nextInt();

System.out.println("enter product name to be updated");

String name=sc.next();

dao=new ProductCrudImpl();

if(dao.updateProductName(idupdate, name)>0) {

System.out.println("product name got updated successfully for id:"+idupdate+" ..as ."+name);

}

else {

System.out.println("product not updated check once...");

}

break;

case 4:dao=new ProductCrudImpl();

List<Product> list=dao.selectProducts();

for(Product products:list) {

System.out.println(products);

}

break;

case 5:System.exit(0);

break;

}

}

}

}

String sql="insert into product values("+product.getPid()+","+"'"+product.getPname()+"',"+product.getCost()+")";

=>when we use statement SQL construction is difficult

PreparedStatement  is intoduced

String sql="insert into product values(?,?,?)";

PreparedStatement ps=con.prepareStatement(sql);

ps.setInt(1,product.getPid());

ps.setString(2, product.getPname());

ps.setInt(3, product.getCost());

//updated code with prepared statement

package com.mphasis.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.ArrayList;

import java.util.List;

import com.mphasis.dbUtil.dbUtil;

import com.mphasis.pojo.Product;

public class ProductCrudImpl implements ProductCrudInterface {

@Override

public int addProduct(Product product) throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection is successfully established ....!!!");

}

String sql="insert into product values(?,?,?)";

PreparedStatement ps=con.prepareStatement(sql);

ps.setInt(1,product.getPid());

ps.setString(2, product.getPname());

ps.setInt(3, product.getCost());

//insert, update ,delete -->return int

int rows=ps.executeUpdate();

return rows;

}

@Override

public int deleteProduct(int id) throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection is successfully established ....!!!");

}

String sql="delete from product where pid=?";

PreparedStatement ps=con.prepareStatement(sql);

ps.setInt(1, id);

//insert, update ,delete -->return int

int rows=ps.executeUpdate();

return rows;

}

@Override

public List<Product> selectProducts() throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection is successfully established ....!!!");

}

String sql="select \* from product";

//table address is stored in rs obj

PreparedStatement ps=con.prepareStatement(sql);

ResultSet rs=ps.executeQuery();

List<Product> list=new ArrayList<Product>();

//row 1

while(rs.next()) {

Product product=new Product();

product.setPid(rs.getInt("pid"));

product.setPname(rs.getString("pname"));

product.setCost(rs.getInt("cost"));

list.add(product);

}

return list;

}

@Override

public int updateProductName(int id,String name) throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection is successfully established ....!!!");

}

String sql="update product set pname=? where pid=?";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1, name);

ps.setInt(2, id);

//insert, update ,delete -->return int

int rows=ps.executeUpdate();

return rows;

}

}

Task: 15 min

Take a Student Pojo with id,name, email,fee, course,department -CRUD using preparedStatement

Foreign key

create table author(authorid int primary key,authorname varchar(20),authoremail varchar(20));

create table book(bookid int primary key,bookname varchar(20),authorid int,foreign key(authorid) references author(authorid));

package com.mphasis.pojo;

public class Book {

private int bookid;

private String bookname;

//aggregation

Author author;

public int getBookid() {

return bookid;

}

public void setBookid(int bookid) {

this.bookid = bookid;

}

public String getBookname() {

return bookname;

}

public void setBookname(String bookname) {

this.bookname = bookname;

}

public Author getAuthor() {

return author;

}

public void setAuthor(Author author) {

this.author = author;

}

@Override

public String toString() {

return "Book [bookid=" + bookid + ", bookname=" + bookname + ", author=" + author + "]";

}

}

package com.mphasis.pojo;

public class Author {

private int authorid;

private String authorname;

private String authoremail;

public int getAuthorid() {

return authorid;

}

public void setAuthorid(int authorid) {

this.authorid = authorid;

}

public String getAuthorname() {

return authorname;

}

public void setAuthorname(String authorname) {

this.authorname = authorname;

}

public String getAuthoremail() {

return authoremail;

}

public void setAuthoremail(String authoremail) {

this.authoremail = authoremail;

}

@Override

public String toString() {

return "Author [authorid=" + authorid + ", authorname=" + authorname + ", authoremail=" + authoremail + "]";

}

}

package com.mphasis.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import com.mphasis.dbUtil.dbUtil;

import com.mphasis.pojo.Author;

import com.mphasis.pojo.Book;

public class AuthorBookDAO {

public void addbookInfo(Book book,Author author) throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection is successfully established ....!!!");

}

String sql="insert into author values(?,?,?)";

PreparedStatement ps=con.prepareStatement(sql);

ps.setInt(1,author.getAuthorid());

ps.setString(2, author.getAuthorname());

ps.setString(3,author.getAuthoremail());

if(ps.executeUpdate()>0) {

System.out.println("author record inserted.. ");

}

else {

System.out.println("author record insert failed ");

}

ps.close();

String sql1="insert into book values(?,?,?)";

PreparedStatement ps1=con.prepareStatement(sql1);

ps1.setInt(1,book.getBookid());

ps1.setString(2, book.getBookname());

ps1.setInt(3, book.getAuthor().getAuthorid());

int row=ps1.executeUpdate();

if(row>0) {

System.out.println("book record inserted.. ");

}

else {

System.out.println("book record insert failed ");

}

}

}

package com.mphasis.main;

import java.sql.SQLException;

import java.util.Scanner;

import com.mphasis.dao.AuthorBookDAO;

import com.mphasis.pojo.Author;

import com.mphasis.pojo.Book;

public class AuthorBookMain {

public static void main(String[] args) throws ClassNotFoundException, SQLException {

Author author=new Author();

Book book=new Book();

Scanner sc=new Scanner(System.in);

System.out.println("enter the author id ");

author.setAuthorid(sc.nextInt());

System.out.println("enter author name");

author.setAuthorname(sc.next());

System.out.println("enter author email");

author.setAuthoremail(sc.next());

System.out.println("enter book id ");

book.setBookid(sc.nextInt());

System.out.println("enter book name");

book.setBookname(sc.next());

book.setAuthor(author);

//System.out.println(book.getAuthor().getAuthorname());

AuthorBookDAO dao=new AuthorBookDAO();

dao.addbookInfo(book, author);

}

}

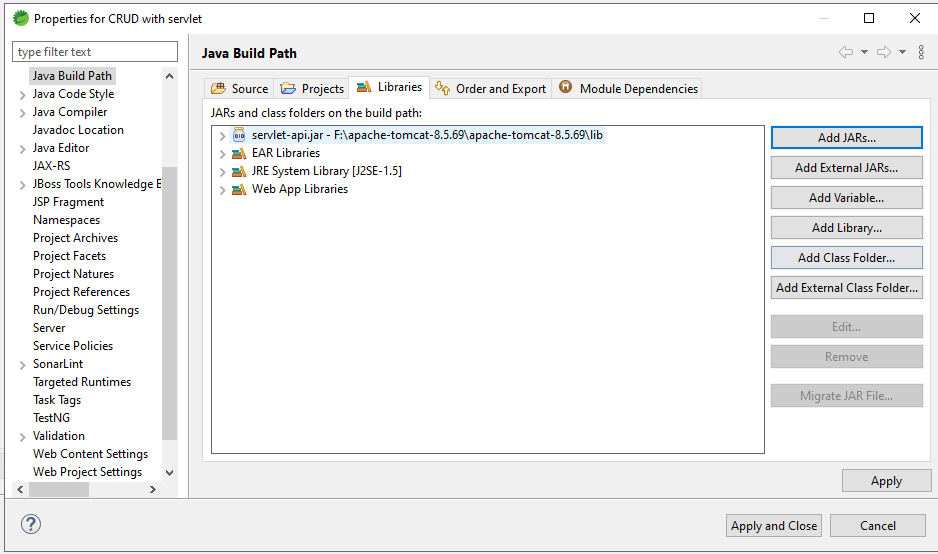
Home assignment - delete, retrieve, update

# need the author name who has written the book java–insert 10 records

select author name from author, book where author.authorid=book.authorid and book.bookname='java';

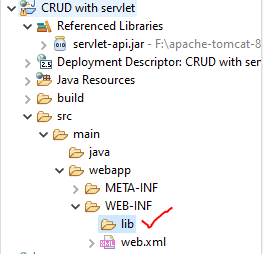
===Servlet as the BE technology with the front end

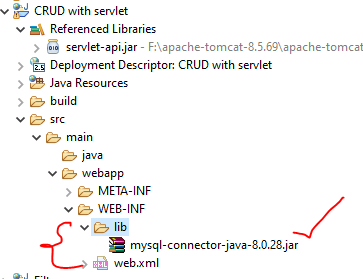
1. Add the dependency [.jar ] -servlet-api, mysql-connector

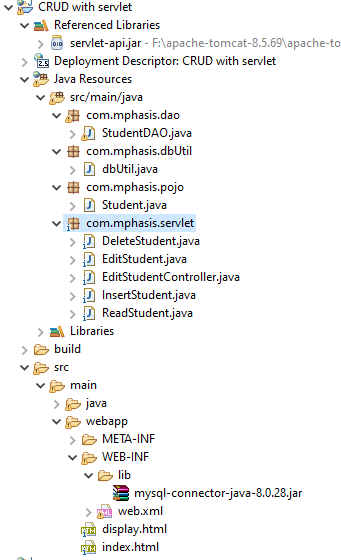


1. Add mysql-connector in the app

Any external jar files into the application is taken from lib folder-physically







<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="InsertStudent" method="post">

Sid <input type="text" name="id"><br>

Sname<input type="text" name="name"><br>

Semail<input type="email" name="email"><br>

<input type="submit" value="register">

</form>

</body>

</html>

package com.mphasis.pojo;

public class Student {

private String sid;

private String sname;

private String semail;

public String getSid() {

return sid;

}

public void setSid(String sid) {

this.sid = sid;

}

public String getSname() {

return sname;

}

public void setSname(String sname) {

this.sname = sname;

}

public String getSemail() {

return semail;

}

public void setSemail(String semail) {

this.semail = semail;

}

}

//table

create table student(sid varchar(10) primary key,sname varchar(20),semail varchar(20));

package com.mphasis.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import com.mphasis.dbUtil.dbUtil;

import com.mphasis.pojo.Student;

public class StudentDAO {

public int insert(Student student) throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection successfull");

}

String sql="insert into student values(?,?,?)";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1,student.getSid());

ps.setString(2, student.getSname());

ps.setString(3,student.getSemail());

return ps.executeUpdate();

}

public List<Student> displaystudent() throws ClassNotFoundException, SQLException {

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection successfull");

}

String sql="select \* from  student";

PreparedStatement ps=con.prepareStatement(sql);

ResultSet rs=ps.executeQuery();

List<Student> list=new ArrayList();

while(rs.next()) {

Student student=new Student();

student.setSid(rs.getString("sid"));

student.setSname(rs.getString("sname"));

student.setSemail(rs.getString("semail"));

list.add(student);

}

return list;

}

public List<Student> deletestudent(String id) throws SQLException, ClassNotFoundException{

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection successfull");

}

String sql="delete from  student where sid=?";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1, id);

ps.executeUpdate();

ps.close();

return displaystudent();

}

public List<Student> updatestudent(String id,String name) throws SQLException, ClassNotFoundException{

Connection con=dbUtil.getConn();

if(con==null) {

System.out.println("connection failed");

}

else {

System.out.println("connection successfull");

}

System.out.println(id+" "+name);

String sql="update student set sname=? where sid=?";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1, name);

ps.setString(2, id);

ps.executeUpdate();

ps.close();

return displaystudent();

}

}

package com.mphasis.servlet;

import java.io.IOException;

import java.sql.SQLException;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.mphasis.dao.StudentDAO;

import com.mphasis.pojo.Student;

/\*\*

 \* Servlet implementation class InsertStudent

 \*/

public class InsertStudent extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public InsertStudent() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

response.getWriter().append("Served at: ").append(request.getContextPath());

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

Student student=new Student();

student.setSid(request.getParameter("id"));

student.setSname(request.getParameter("name"));

student.setSemail(request.getParameter("email"));

StudentDAO dao=new StudentDAO();

try {

if(dao.insert(student)>0)

{

response.sendRedirect("display.html");

}

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1><i>insertion done successfully....</i></h1>

<form action="ReadStudent">

<input type="submit" value="read all students">

</form>

</body>

</html>

package com.mphasis.servlet;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.SQLException;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.mphasis.dao.StudentDAO;

import com.mphasis.pojo.Student;

/\*\*

 \* Servlet implementation class ReadStudent

 \*/

public class ReadStudent extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public ReadStudent() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

StudentDAO dao=new StudentDAO();

PrintWriter pw=response.getWriter();

try {

List<Student> list=dao.displaystudent();

response.setContentType("text/html");

pw.print("<html><body><center><h1><i>StudentDetails</i></h1></center>");

pw.print("<table border='1' align='center'>"

+ "<tr><th>Studentid</th><th>Studentname</th><th>Studentemail</th><th>Edit Student</th><th>Delete Student</th></tr>");

for(Student student:list) {

pw.print("<tr><td>"+student.getSid()+"</td><td>"+student.getSname()+"</td><td>"+student.getSemail()+"</td><td><a href='./EditStudent?id="+student.getSid()+"'>EditStudentName</a></td><td><a href='./DeleteStudent?id="+student.getSid()+"'>DeleteStudent</a></td></tr>");

}

pw.print("</table></body></html>");

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

doGet(request, response);

}

}

package com.mphasis.servlet;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.SQLException;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.mphasis.dao.StudentDAO;

import com.mphasis.pojo.Student;

/\*\*

 \* Servlet implementation class DeleteStudent

 \*/

public class DeleteStudent extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public DeleteStudent() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

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

System.out.println("id value is "+id);

StudentDAO dao=new StudentDAO();

PrintWriter pw=response.getWriter();

try {

List<Student> list=dao.deletestudent(id);

response.setContentType("text/html");

pw.print("<html><body><center><h1><i>StudentDetails</i></h1></center>");

pw.print("<table border='1' align='center'>"

+ "<tr><th>Studentid</th><th>Studentname</th><th>Studentemail</th><th>Edit Student</th><th>Delete Student</th></tr>");

for(Student student:list) {

pw.print("<tr><td>"+student.getSid()+"</td><td>"+student.getSname()+"</td><td>"+student.getSemail()+"</td><td><a href='./EditStudent'>EditStudent</a></td><td><a href='./DeleteStudent?id="+student.getSid()+"'>DeleteStudent</a></td></tr>");

}

pw.print("</table></body></html>");

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

doGet(request, response);

}

}

package com.mphasis.servlet;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

 \* Servlet implementation class EditStudent

 \*/

public class EditStudent extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public EditStudent() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

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

System.out.println("id value is "+id);

PrintWriter pw=response.getWriter();

response.setContentType("text/html");

pw.print("<html><body><form action='EditStudentController'>"

+ "<input type='hidden' name='id' value="+id+">"

+ "Enter the studentname to be edited<input type='text' name='name'>"

+ "<input type='submit' value='update'></form></body></html> ");

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

doGet(request, response);

}

}

package com.mphasis.servlet;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.SQLException;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.mphasis.dao.StudentDAO;

import com.mphasis.pojo.Student;

/\*\*

 \* Servlet implementation class EditStudentController

 \*/

public class EditStudentController extends HttpServlet {

private static final long serialVersionUID = 1L;

    /\*\*

     \* @see HttpServlet#HttpServlet()

     \*/

    public EditStudentController() {

        super();

        // TODO Auto-generated constructor stub

    }

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

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

System.out.println("the id is "+id);

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

StudentDAO dao=new StudentDAO();

PrintWriter pw=response.getWriter();

response.setContentType("text/html");

List<Student> list;

try {

list = dao.updatestudent(id, name);

pw.print("<html><body><center><h1><i>StudentDetails</i></h1></center>");

pw.print("<table border='1' align='center'>"

+ "<tr><th>Studentid</th><th>Studentname</th><th>Studentemail</th><th>Edit Student</th><th>Delete Student</th></tr>");

for(Student student:list) {

pw.print("<tr><td>"+student.getSid()+"</td><td>"+student.getSname()+"</td><td>"+student.getSemail()+"</td><td><a href='./EditStudent'>EditStudent</a></td><td><a href='./DeleteStudent?id="+student.getSid()+"'>DeleteStudent</a></td></tr>");

}

pw.print("</table></body></html>");

}

catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

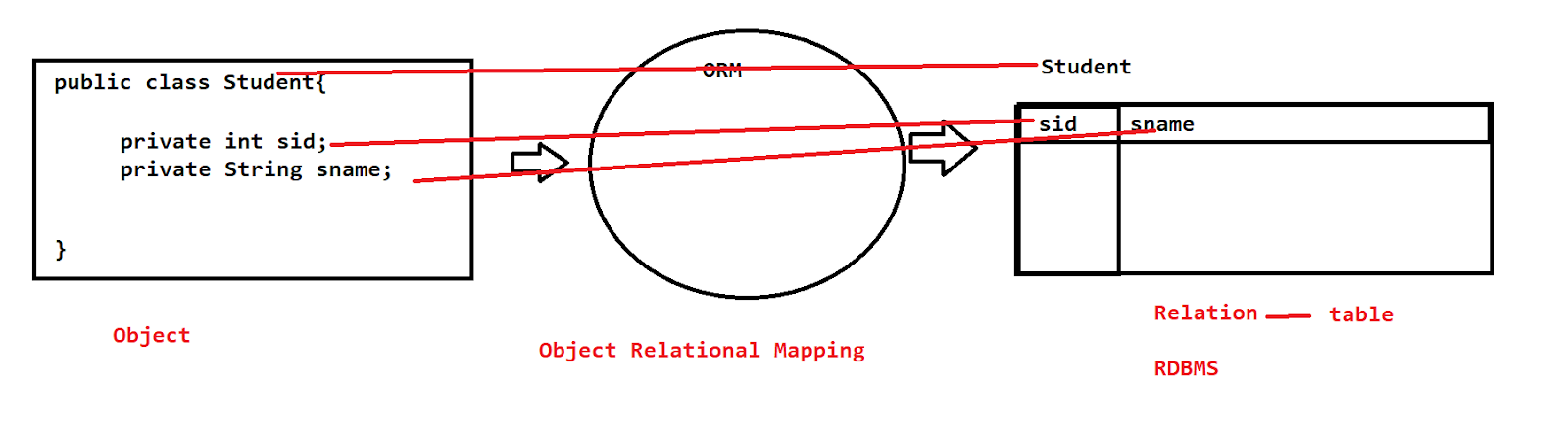
doGet(request, response);

}

}

>Hibernate :

It is a framework used to operate on the POJO class which helps in connecting with the tables.



Hibernate is considered as an ORM tool

Pojo -> ORM -> tables

1. Open source
2. Faster performance
3. Inbuild methods that perform crud oprs
4. Query by using HQL

JDBC

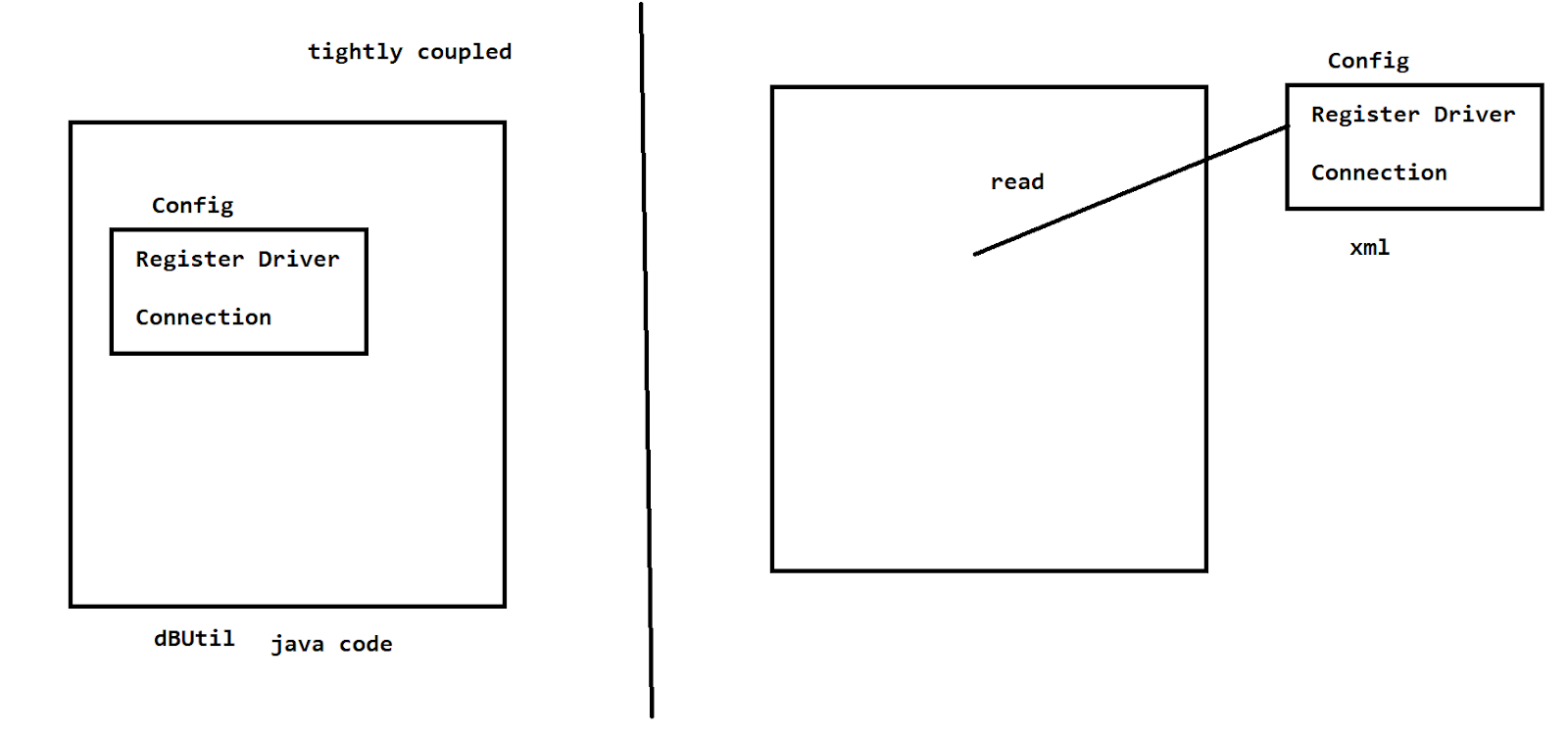
>Driver

>Connection

>Statement

>Execute

>Close



>In future on any change on the java code we need to recompile the entire code on it .

> We are hard coding the config on the java file => not correct

By Xml config we can reduce this 2 statements .

POJO is mapped only by using the @ not by using the xml file

Steps for hibernate :

1. StandardServiceRegistry
2. Metadata
3. SessionFactory
4. Session
5. Transaction
6. Close the connections

External config - hibernate.cfg.xml

Annotations are the predefined metadata [all the actions that are predefined on this]

@Id- used to define the pK

@Column-used to define the feild

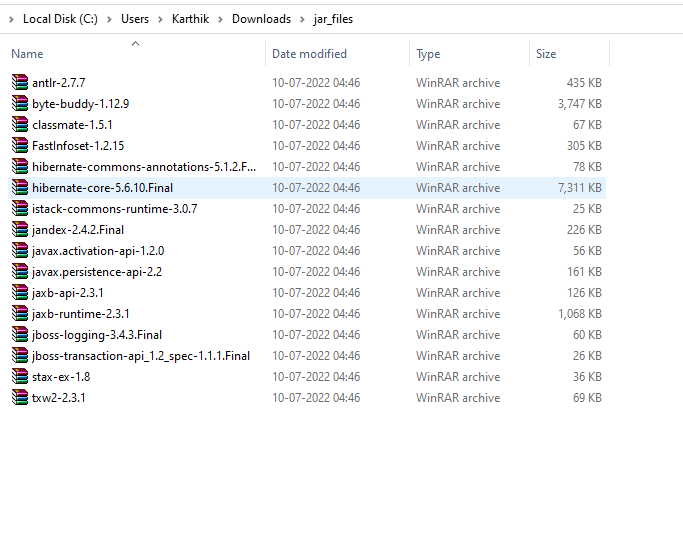
@Entity - used to define the tbl name as the class name

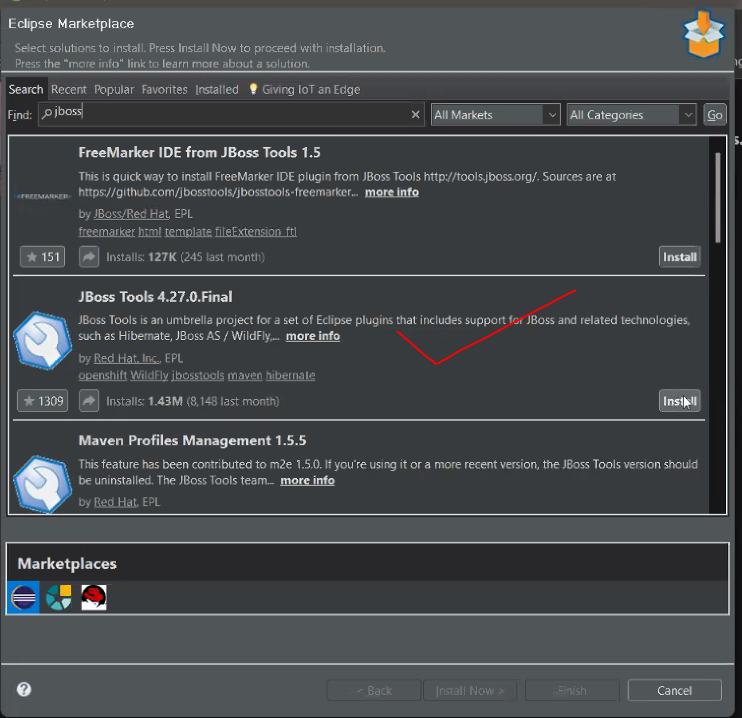
@Table - used to define the tbl name of ur wish.

For performing hibernate operations we need jar files :

<https://drive.google.com/file/d/1Q4Km389L2fqF5qzsFgztpnsZuWtiHPFu/view?usp=sharing>

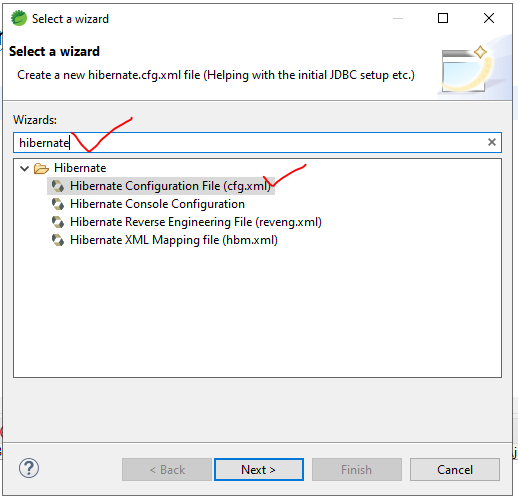
Add mysql driver to the build path

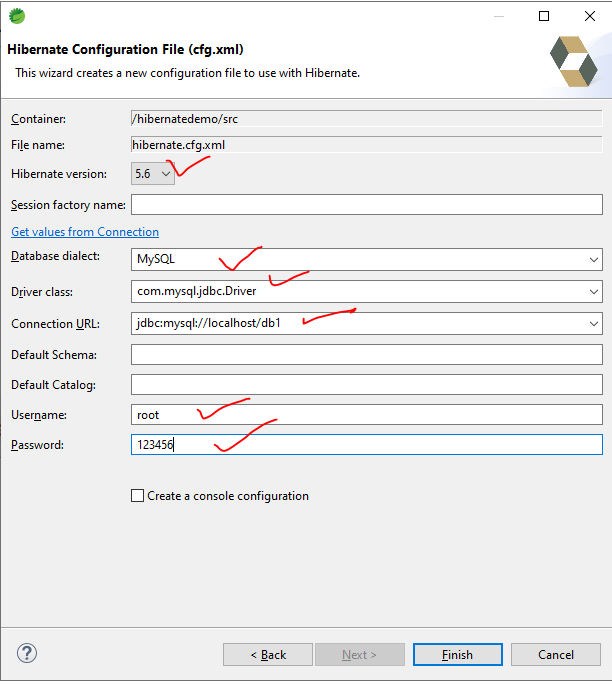


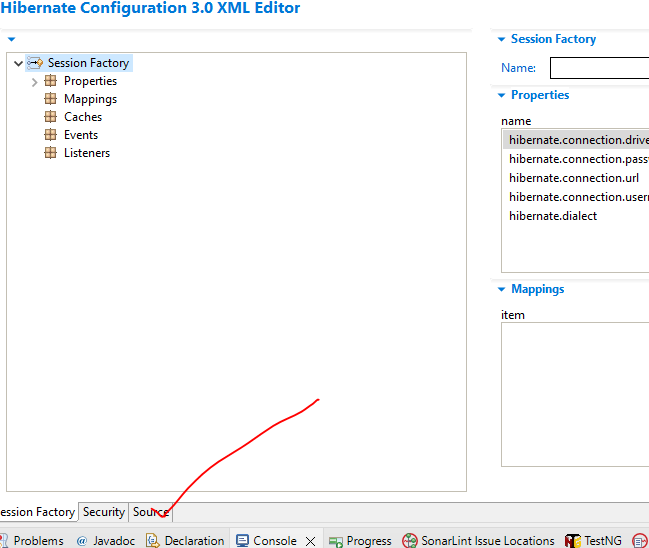


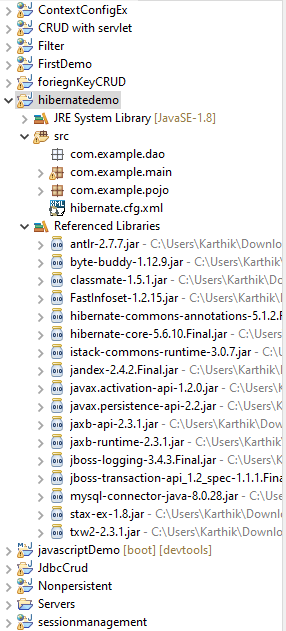
Build path with the jar files into the application.

hibernate.cfg.xml  is created at the src level of the project folder .









package com.example.pojo;

import javax.persistence.Entity;

import javax.persistence.Id;

@Entity

public class Employee {

@Id

private int eid;

//@Column(name="ename")

private String empname;

private String empemail;

private double salary;

public int getEid() {

return eid;

}

public void setEid(int eid) {

this.eid = eid;

}

public String getEmpname() {

return empname;

}

public void setEmpname(String empname) {

this.empname = empname;

}

public String getEmpemail() {

return empemail;

}

public void setEmpemail(String empemail) {

this.empemail = empemail;

}

public double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

@Override

public String toString() {

return "Employee [eid=" + eid + ", empname=" + empname + ", empemail=" + empemail + ", salary=" + salary + "]";

}

}

package com.example.main;

import java.util.Scanner;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

import org.hibernate.boot.Metadata;

import org.hibernate.boot.MetadataSources;

import org.hibernate.boot.registry.StandardServiceRegistry;

import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

import com.example.pojo.Employee;

/\*

StandardServiceRegistry

Metadata

SessionFactory

Session

Transaction

Close the connections

 \* \*/

public class EmpMain {

public static void main(String[] args) {

//SSR is used to map the config file and execute it .

StandardServiceRegistry  ssr=new StandardServiceRegistryBuilder().configure("hibernate.cfg.xml").build();

//Metadata of the xml file is read by this object

Metadata md=new MetadataSources(ssr).getMetadataBuilder().build();

//session-factory- db

SessionFactory sf=md.getSessionFactoryBuilder().build();

//all the crud operations need to be done in Session

Session s=sf.openSession();

//Transaction- perform sql operations and commit it permenantly on the db

Transaction t=s.beginTransaction();

Scanner sc=new Scanner(System.in);

Employee e=new Employee();

System.out.println("enter the eid");

e.setEid(sc.nextInt());

System.out.println("enter the ename");

e.setEmpname(sc.next());

System.out.println("enter the email");

e.setEmpemail(sc.next());

System.out.println("enter the salary");

e.setSalary(sc.nextDouble());

s.save(e);  //insert

t.commit();

s.close();

sf.close();

}

}

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

    <session-factory>

        <!--db driver , connection  -->

        <property name="hibernate.connection.driver\_class">com.mysql.jdbc.Driver</property>

        <property name="hibernate.connection.password">123456</property>

        <property name="hibernate.connection.url">jdbc:mysql://localhost/db1</property>

        <property name="hibernate.connection.username">root</property>

        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>

         <!--hibernate config  -->

       <property name="hbm2ddl.auto">update</property>

       <property name="show\_sql">true</property>

        <!--pojo -->

       <mapping class="com.example.pojo.Employee"/>

    </session-factory>

</hibernate-configuration>

 <property name="hbm2ddl.auto">update</property>

When we write update if the table is not existing it creates the table and perform insert into it  or if the table is existing it performs an insert on it .

 <property name="hbm2ddl.auto">create</property>

When we write create the table gets dropped and every time on execution a new table will be created.

//delete

System.out.println("enter the eid");

e.setEid(sc.nextInt());

s.delete(e);

//update

Employee e=new Employee();

System.out.println("enter the eid");

e.setEid(sc.nextInt());

System.out.println("enter the ename");

e.setEmpname(sc.next());

System.out.println("enter the email");

e.setEmpemail(sc.next());

System.out.println("enter the salary");

e.setSalary(sc.nextDouble());

s.saveOrUpdate(e);  //update

//select with id

Employee e1=s.get(Employee.class,2);

System.out.println(e1);

HQL =>hibernate query language

HQL = SQL

SQL - select  \*  from employee[table]

HQL-select  \*  from Employee[pojo]

Select empname[class attribute] from Employee[pojo]

>we design the quires which are independent of the SQL in the dB

>Steps

1. Query interface
2. Session object
3. createQuery()

HQL - 2 basic outputs

1. Single object -q.unquieResult()
2. List of object -q.list()

//for(int i=1;i<10;i++) {

// Employee e=new Employee();

// e.setEid(i);

// e.setEmpname("name"+i);

// e.setEmpemail("name"+i+"@c.c");

// e.setSalary(i\*10000);

// s.save(e);

//}

//HQl

//select \* from employee;

// Query q=s.createQuery("from Employee");

// List<Employee> list=q.list();

// for(Employee e:list) {

// System.out.println(e);

// }

//select \* from employee where salary=10000

// Query q=s.createQuery("from Employee where salary=10000");

// LHS employee object = RHS object

// Employee e=(Employee) q.uniqueResult();

// System.out.println(e);

//select \* from employee where salary>10000

// Query q=s.createQuery("from Employee where salary>10000");

// List<Employee> list=q.list();

// for(Employee e:list) {

// System.out.println(e);

// }

// select empname,empemail from employee where salary=10000

// Query q=s.createQuery("select empname,empemail from Employee where salary=10000");

// Object[] emp= (Object[]) q.uniqueResult();

// System.out.println(emp[0]+" "+emp[1]) ;

//select empname,empemail from employee where salary>10000

// Query q=s.createQuery("select empname,empemail from Employee where salary>10000");

// List<Object[]> emp=q.list();

// for(Object e[]:emp) {

// System.out.println(e[0]+"  "+e[1]);

// }

// Query q=s.createQuery("select empname from Employee where salary=10000");

// String empname=(String) q.uniqueResult();

// System.out.println(empname);

// Query q=s.createQuery("select sum(salary) from Employee where salary>10000");

// double sum=(double) q.uniqueResult();

// System.out.println(sum);

//update query by hql

Query q=s.createQuery("update Employee set empname=:name where eid=:no");

q.setParameter("name","karthik");

q.setParameter("no", 1);

q.executeUpdate();

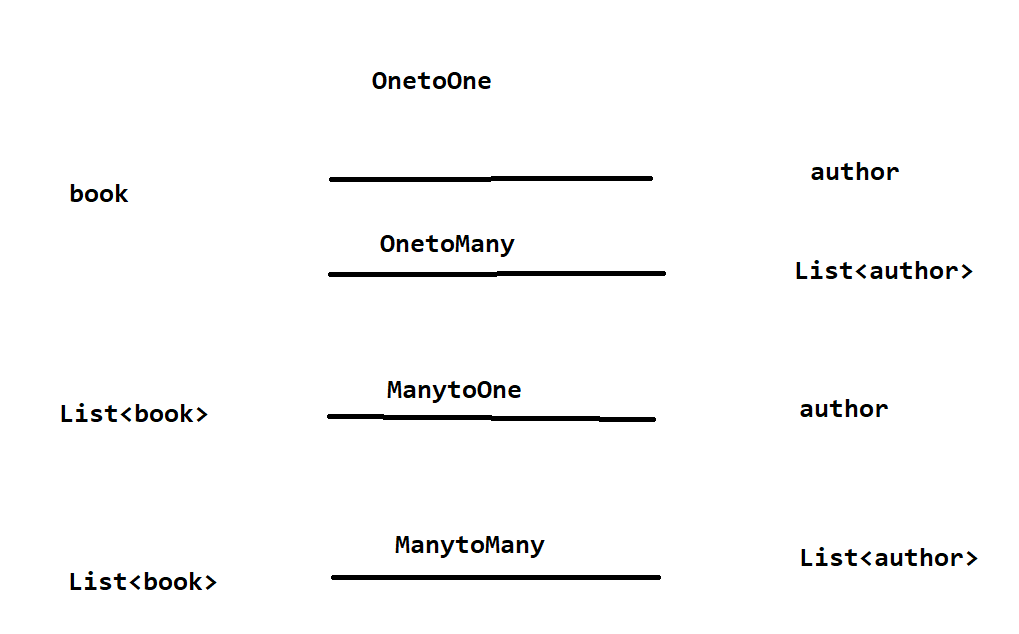
Note : if the id need to get auto generated from the sequence of 1 then we can use an @ called

@GenratedValue(strategy=GenerationType.AUTO)

int id;      //3   ->4  ⇒last seq where the work have been stopped from there it gets continued

@GenratedValue(strategy=GenerationType.IDENTITY)

int id;     //1 =>starts the id from sequence 1 only



package com.book;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Author {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

private String email;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

package com.book;

import javax.persistence.CascadeType;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.OneToOne;

@Entity

public class Book {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int bid;

private String title;

//all the data in the author table should be related [we can access all the data ]

@OneToOne(cascade = CascadeType.ALL)

@JoinColumn(name="id")

private Author author;

public int getBid() {

return bid;

}

public void setBid(int bid) {

this.bid = bid;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public Author getAuthor() {

return author;

}

public void setAuthor(Author author) {

this.author = author;

}

}

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

    <session-factory>

        <!--db driver , connection  -->

        <property name="hibernate.connection.driver\_class">com.mysql.jdbc.Driver</property>

        <property name="hibernate.connection.password">123456</property>

        <property name="hibernate.connection.url">jdbc:mysql://localhost/db2</property>

        <property name="hibernate.connection.username">root</property>

        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>

         <!--hibernate config  -->

       <property name="hbm2ddl.auto">update</property>

       <property name="show\_sql">true</property>

        <!--pojo -->

       <mapping class="com.book.Book"/>

       <mapping class="com.book.Author"/>

    </session-factory>

</hibernate-configuration>

package com.book.main;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

import org.hibernate.boot.Metadata;

import org.hibernate.boot.MetadataSources;

import org.hibernate.boot.registry.StandardServiceRegistry;

import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

import com.book.Author;

import com.book.Book;

public class BookMain {

public static void main(String[] args) {

//SSR is used to map the config file and execute it .

StandardServiceRegistry  ssr=new StandardServiceRegistryBuilder().configure("hibernate.cfg.xml").build();

//Metadata of the xml file is read by this object

Metadata md=new MetadataSources(ssr).getMetadataBuilder().build();

//session-factory- db

SessionFactory sf=md.getSessionFactoryBuilder().build();

//all the crud operations need to be done in Session

Session s=sf.openSession();

//Transaction- perform sql operations and commit it permenantly on the db

Transaction t=s.beginTransaction();

Author au=new Author();

au.setName("karthik");

au.setEmail("k@c.c");

s.save(au);

Book book=new Book();

book.setTitle("java");

book.setAuthor(au);

s.save(book);

//by using HQL find out the author name for the book java?

t.commit();

s.close();

sf.close();

}

}

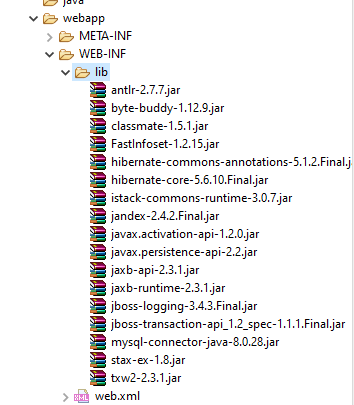
//JSP with hibernate

Step -1:



Step -2 :

Hibernate and the dB all the dependencies take it into the lib



1. Java code - pojo

package com.mphasis.pojo;

import java.util.Date;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Temporal;

import javax.persistence.TemporalType;

@Entity

public class Product {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int pid;

private String pname;

private float cost;

//when we have a util based date then we need to @ it with Temporal

@Temporal(TemporalType.DATE)

private Date orderdate;

public int getPid() {

return pid;

}

public void setPid(int pid) {

this.pid = pid;

}

public String getPname() {

return pname;

}

public void setPname(String pname) {

this.pname = pname;

}

public float getCost() {

return cost;

}

public void setCost(float cost) {

this.cost = cost;

}

public Date getOrderdate() {

return orderdate;

}

public void setOrderdate(Date orderdate) {

this.orderdate = orderdate;

}

}

//dbutil

package com.mphasis.dButil;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.boot.Metadata;

import org.hibernate.boot.MetadataSources;

import org.hibernate.boot.registry.StandardServiceRegistry;

import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

public class DbUtil {

StandardServiceRegistry  ssr=null;

Metadata md=null;

SessionFactory sf=null;

Session session=null;

public Session dbConn() {

//SSR is used to map the config file and execute it .

ssr=new StandardServiceRegistryBuilder().configure("hibernate.cfg1.xml").build();

//Metadata of the xml file is read by this object

md=new MetadataSources(ssr).getMetadataBuilder().build();

//session-factory- db

sf=md.getSessionFactoryBuilder().build();

//all the crud operations need to be done in Session

session=sf.openSession();

return session;

}

}

//DAO

package com.mphasis.dao;

import java.util.List;

import org.hibernate.Session;

import org.hibernate.Transaction;

import org.hibernate.query.Query;

import com.mphasis.dButil.DbUtil;

import com.mphasis.pojo.Product;

public class ProductDAO {

public int addProduct(Product product) {

DbUtil dbconn=new DbUtil();

Session session=dbconn.dbConn();

Transaction trans=session.beginTransaction();

int value=(Integer) session.save(product);

trans.commit();

session.close();

return value;

}

public List<Product> display(){

DbUtil dbconn=new DbUtil();

Session session=dbconn.dbConn();

Transaction trans=session.beginTransaction();

//HQL

Query query=session.createQuery("from Product");

List<Product> list=query.list();

trans.commit();

session.close();

return list;

}

public List<Product> delete(Product product) {

DbUtil dbconn=new DbUtil();

Session session=dbconn.dbConn();

Transaction trans=session.beginTransaction();

session.delete(product);

trans.commit();

session.close();

return display();

}

}

hibernate.cfg.xml

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

    <session-factory>

        <!--db driver , connection  -->

        <property name="hibernate.connection.driver\_class">com.mysql.jdbc.Driver</property>

        <property name="hibernate.connection.password">123456</property>

        <property name="hibernate.connection.url">jdbc:mysql://localhost/db2</property>

        <property name="hibernate.connection.username">root</property>

        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>

         <!--hibernate config  -->

       <property name="hbm2ddl.auto">update</property>

       <property name="show\_sql">true</property>

        <!--pojo -->

       <mapping class="com.mphasis.pojo.Product"/>

    </session-factory>

</hibernate-configuration>

//index.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

      <%@ page import="com.mphasis.dao.\*" %>

    <%@ page import="com.mphasis.pojo.\*" %>

    <%@page import="java.util.\*" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="addproduct.jsp">

<input type="submit" value="addproduct">

</form>

<h1><i>List of products </i></h1>

<table border="1">

<tr><th>Pid</th><th>Pname</th><th>PCost</th><th>POrderDate</th><th>Edit Action</th><th>Delete Action</th><tr>

<%

ProductDAO dao=new ProductDAO();

List<Product> list=dao.display();

for(Product p:list){

%>

<tr><td><%=p.getPid() %></td><td><%=p.getPname() %></td><td><%=p.getCost() %></td><td><%=p.getOrderdate() %></td><td><a href="edit.jsp?id=<%=p.getPid()%>">Edit</a></td><td><a href="delete.jsp?id=<%=p.getPid()%>">Delete</a></td></tr>

<%} %>

</table>

</body>

</html>

//addproduct.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1><i>Add a product</i></h1>

<form action="addController.jsp">

<table>

<tr><td>Pname</td><td><input type="text" name="pname"></td></tr>

<tr><td>Pcost</td><td><input type="text" name="pcost"></td></tr>

<tr><td>Orderdate</td><td><input type="date" name="pdoo"></td></tr>

<tr><td></td><td><input type="submit" value="add"></td></tr>

</table>

</form>

</body>

</html>

//addController.jsp

<%@page import="java.text.SimpleDateFormat"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

    <%@ page import="com.mphasis.dao.\*" %>

    <%@ page import="com.mphasis.pojo.\*" %>

    <%@page import="java.util.\*" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<%

ProductDAO dao=new ProductDAO();

Product product=new Product();

product.setPname(request.getParameter("pname"));

product.setCost(Float.parseFloat(request.getParameter("pcost")));

//mysql always accept the date in the format of  yyyy-MM-dd

//convert the java based util date to sql date

SimpleDateFormat dateFormat=new SimpleDateFormat("yyyy-MM-dd");

Date date=dateFormat.parse(request.getParameter("pdoo"));

product.setOrderdate(date);

int row=dao.addProduct(product);

if(row>0){

response.sendRedirect("success.jsp");

}

else{

response.sendRedirect("fail.jsp");

}

%>

</body>

</html>

//success.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1>the product got added successfully....</h1>

<%@ include file="index.jsp" %>

</body>

</html>

//delete.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

     <%@ page import="com.mphasis.dao.\*" %>

    <%@ page import="com.mphasis.pojo.\*" %>

     <%@page import="java.util.\*" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1>deleted the product successfully....</h1>

<%@include file="button.jsp" %>

<h1><i>List of products </i></h1>

<table border="1">

<tr><th>Pid</th><th>Pname</th><th>PCost</th><th>POrderDate</th><th>Edit Action</th><th>Delete Action</th><tr>

<%

ProductDAO dao=new ProductDAO();

Product product=new Product();

product.setPid(Integer.parseInt(request.getParameter("id")));

List<Product> list=dao.delete(product);

for(Product p:list){

%>

<tr><td><%=p.getPid() %></td><td><%=p.getPname() %></td><td><%=p.getCost() %></td><td><%=p.getOrderdate() %></td><td><a href="edit.jsp?id=<%=p.getPid()%>">Edit</a></td><td><a href="delete.jsp?id=<%=p.getPid()%>">Delete</a></td></tr>

<%} %>

</table>

</body>

</html>

//button.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="addproduct.jsp">

<input type="submit" value="addproduct">

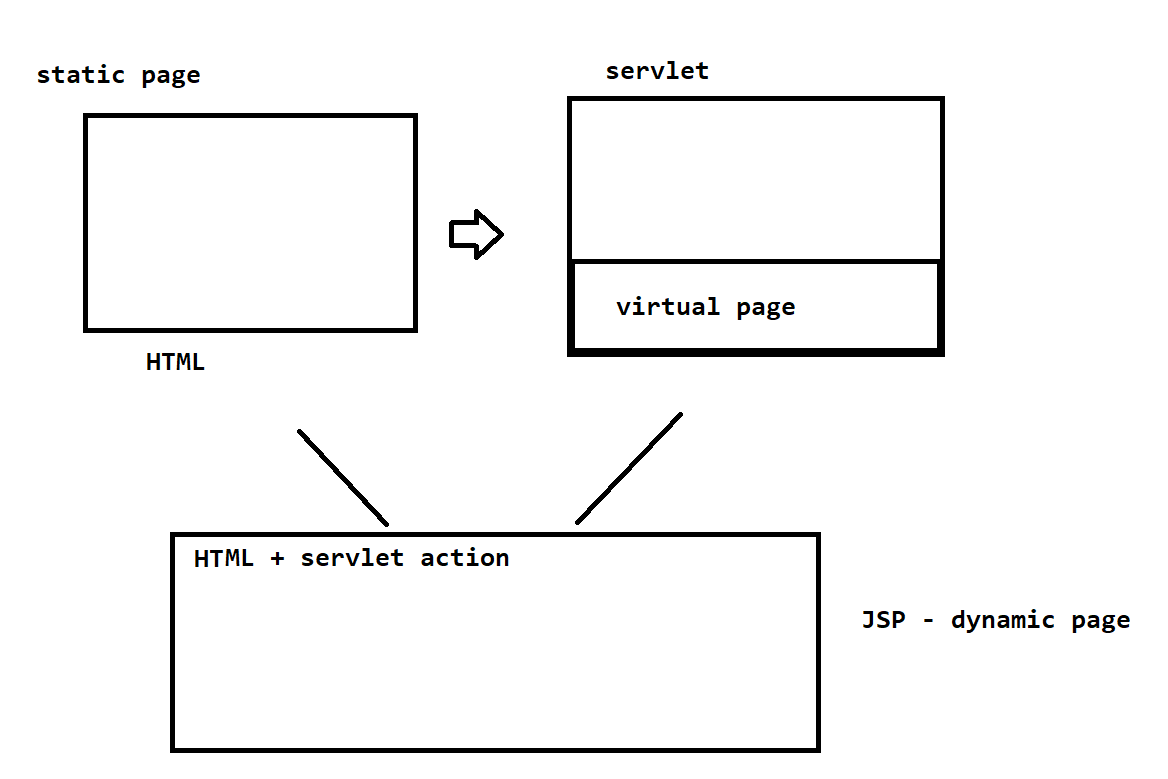
</form>

</body>

</html>



JSP - Java server pages

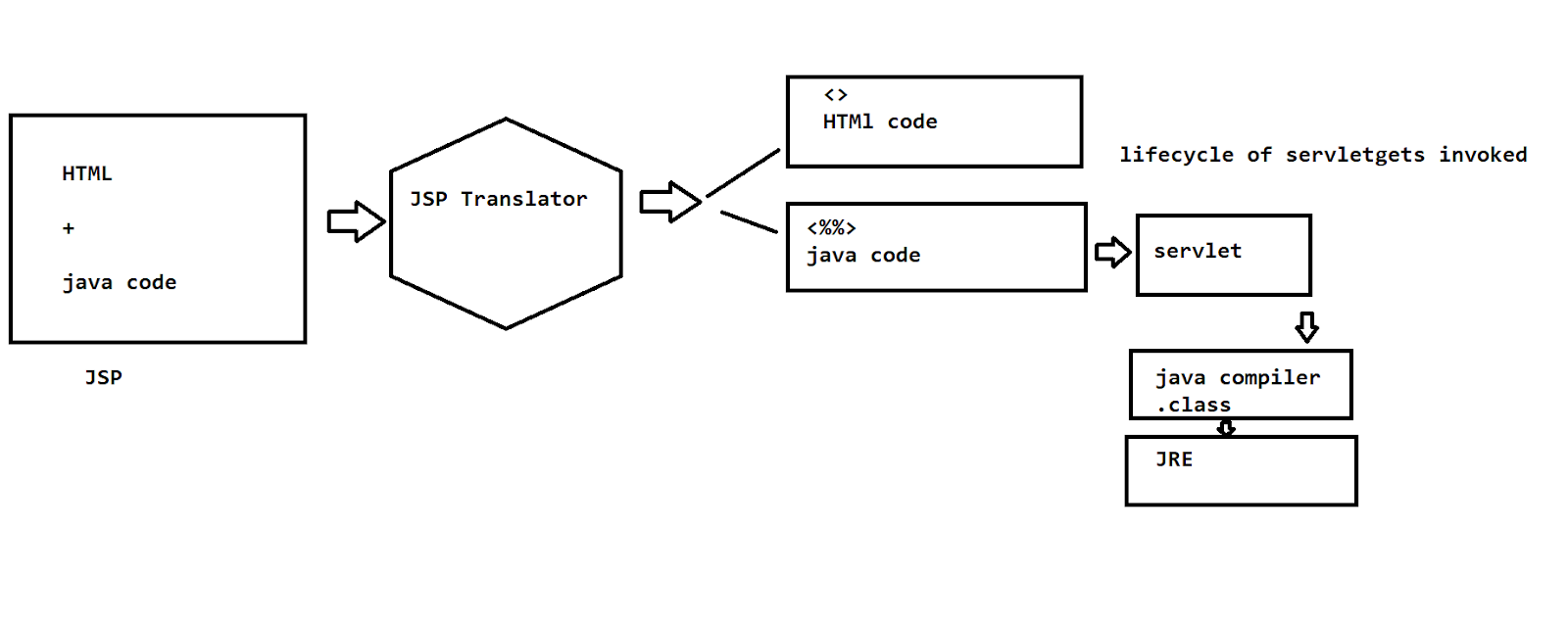


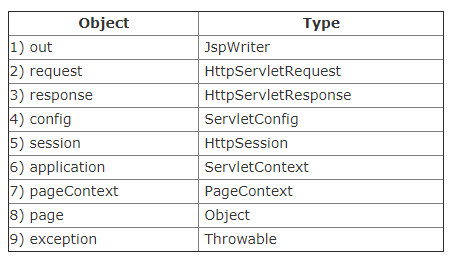
On servlet we have given the highest priority to the java component

Java code having HTML in it

On Jsp we have given the priority to the design of the page by maintaining the java action on it

HTML code having java code on it .





JSP -  HTML                + java code

          <> markups      + <%%>-scriptlets

Scripting elements :

1.scriptlet- <%%>

2.expression-<%=%>

3.declaration-<%!%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1>Hi Mphasis Learners </h1>

<%!

int counter=0;

int getusers(){

return ++counter;

}

%>

<%out.println("no of vistors on this page"+getusers()); %>

<%="no of vistors on this page"+getusers() %>

</body>

</html>

//application,config

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<%=application.getInitParameter("url") %>

<%=config.getInitParameter("password") %>

</body>

</html>

//web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/j2ee" xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee, http://java.sun.com/xml/ns/j2ee/web-app\_2\_5.xsd" id="WebApp\_ID" version="2.5">

  <display-name>jspdemo</display-name>

  <welcome-file-list>

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

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

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

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

    <welcome-file>default.jsp</welcome-file>

    <welcome-file>default.htm</welcome-file>

  </welcome-file-list>

  <context-param>

  <param-name>url</param-name>

  <param-value>jdbc:mysql://localhost:3306/sonoo</param-value>

  </context-param>

  <servlet>

  <description></description>

    <display-name>index</display-name>

    <servlet-name>index</servlet-name>

    <jsp-file>/index.jsp</jsp-file>

    <init-param>

      <param-name>password</param-name>

   <param-value>123456</param-value>

    </init-param>

  </servlet>

  <servlet-mapping>

    <servlet-name>index</servlet-name>

    <url-pattern>/index</url-pattern>

  </servlet-mapping>

  <servlet>

  <description></description>

    <display-name>display</display-name>

    <servlet-name>display</servlet-name>

    <jsp-file>/display.jsp</jsp-file>

  </servlet>

  <servlet-mapping>

    <servlet-name>display</servlet-name>

    <url-pattern>/display</url-pattern>

  </servlet-mapping>

</web-app>

//session

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="display.jsp">

user<input type="text" name="user">

<input type="submit">

</form>

</body>

</html>

//display.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<%

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

session.setAttribute("sessionid", user);

response.sendRedirect("show.jsp");

%>

</body>

</html>

//show.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<%="Welcome to "+session.getAttribute("sessionid") %>

</body>

</html>

//exception

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="display.jsp">

Enter a no1<input type="text" name="no1">

Enter a no2<input type="text" name="no2">

<input type="submit" value="divide">

</form>

</body>

</html>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<%int res=(Integer.parseInt(request.getParameter("no1"))/Integer.parseInt(request.getParameter("no2"))); %>

<%="The results is "+res%>

</body>

</html>



//updated display.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

    <%@page errorPage="show.jsp" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<%int res=(Integer.parseInt(request.getParameter("no1"))/Integer.parseInt(request.getParameter("no2"))); %>

<%="The results is "+res%>

</body>

</html>

//show.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<%@page isErrorPage="true" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<%="the exception is "+exception  %>

</body>

</html>

Directives :

<%@ %>

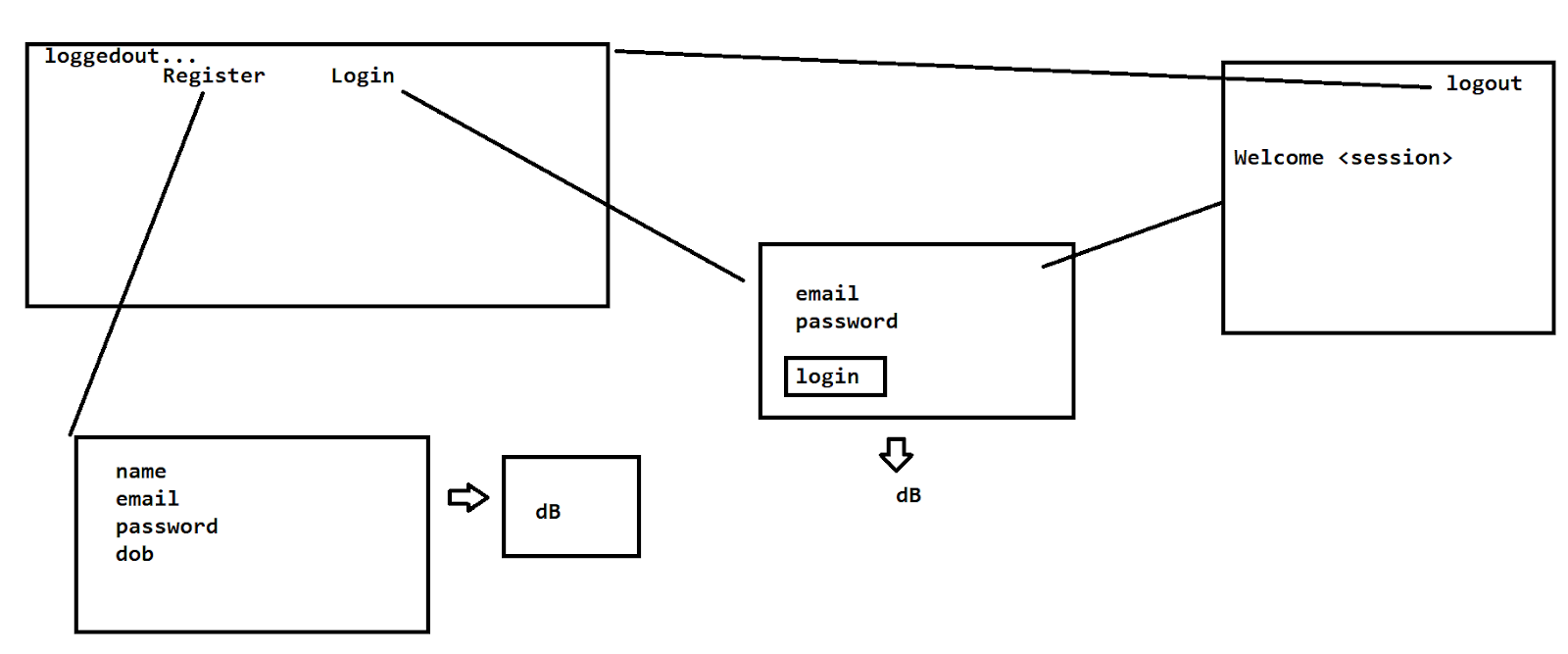
1. page - import the package / loads the error pages

<%@ page import="com.mphasis.dao.\*" %>

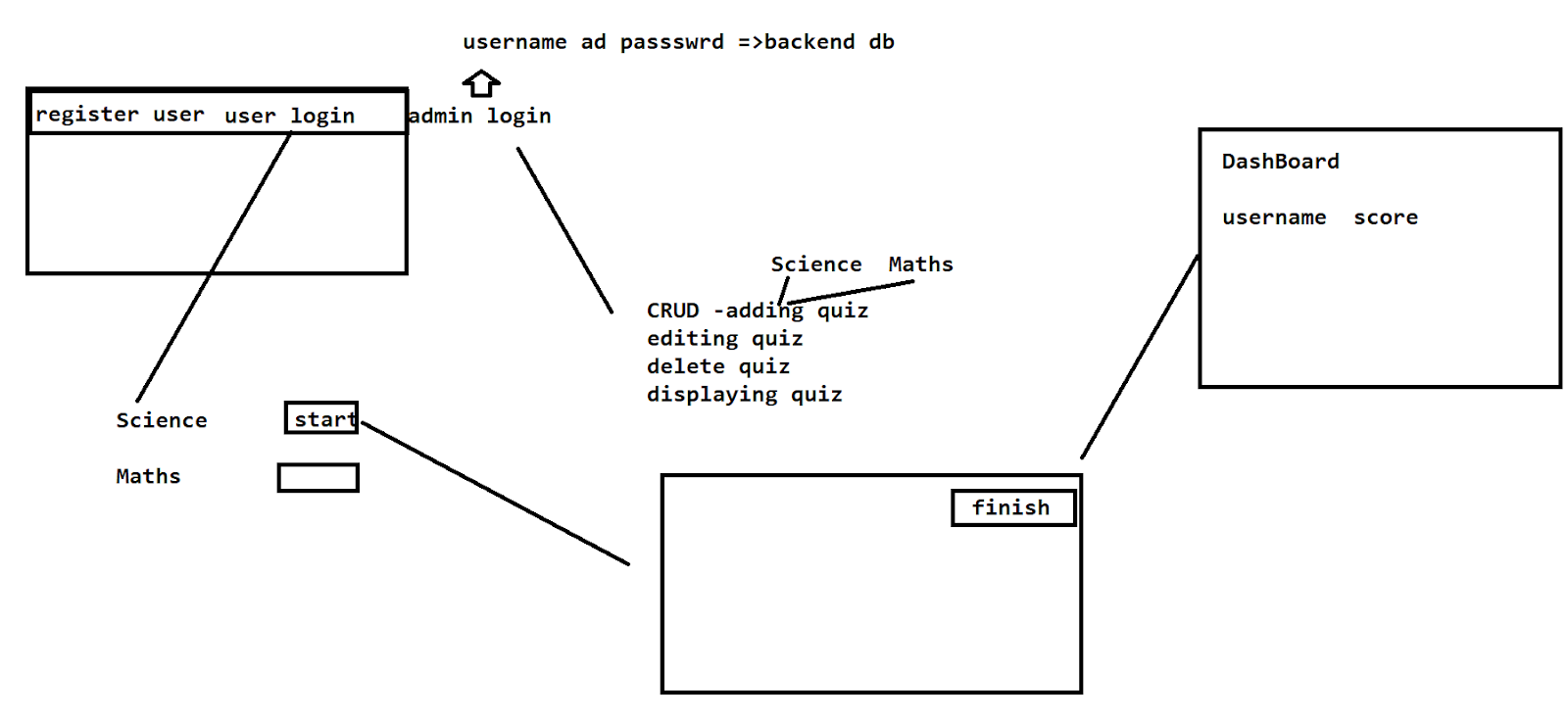
1. include

<%@ include file="index.jsp" %>

Task :



Functional flow of Quiz portal



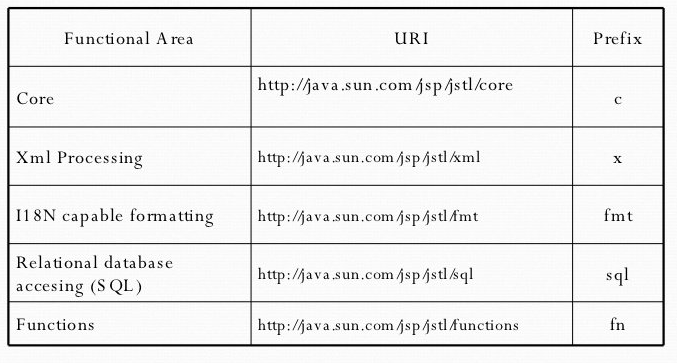


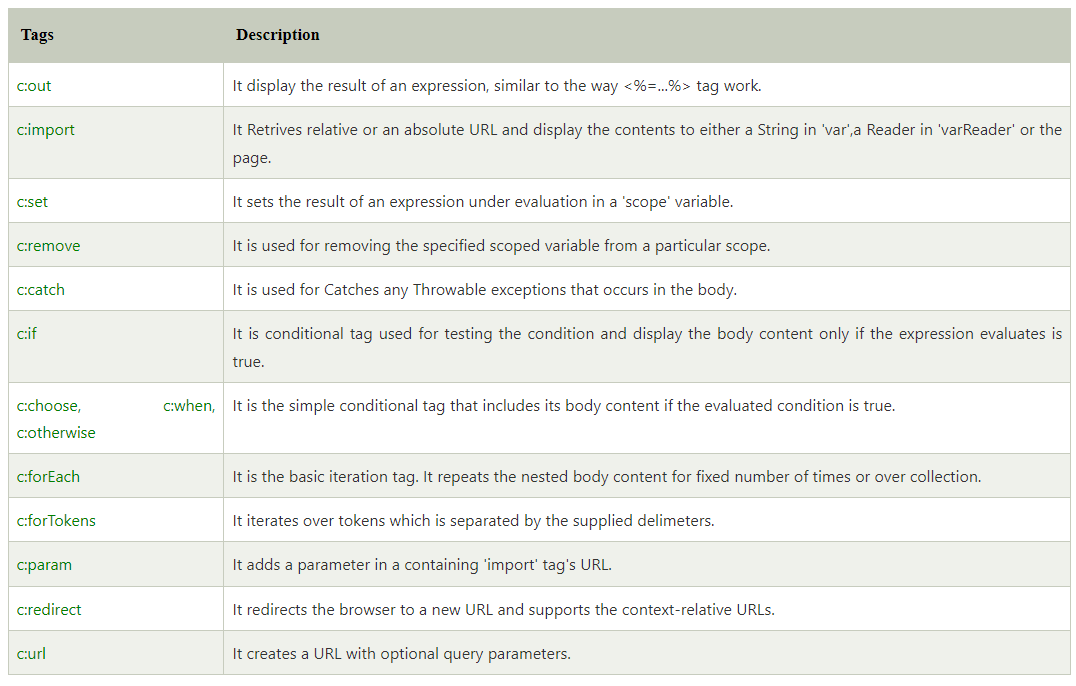
//JSTL

JSP standard tag library

>reduce the scriptlet code.

>JSTL elements access is much faster compared to scriptlet.

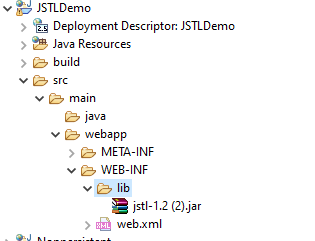




Core tag - declarations, flow control , url management

JSTL -

<https://mvnrepository.com/artifact/javax.servlet/jstl/1.2>



<%@taglib%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

    <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1>--------------core declarations--------------- </h1>

<h2>--declare and printing the value--</h2>

<c:set var="a" value="${101}"></c:set>

<c:out value="${a}"></c:out>

<h2>--remove the value --</h2>

<p>before the value</p>

<c:out value="${a+10}"></c:out>

<p>after remove the value</p>

<c:remove var="a"/>

<c:out value="${a+10}"></c:out>

<h1>--------------conditions--------------- </h1>

<h2>--if condition --</h2>

<c:set var="a1" value="${110}"></c:set>

<c:set var="b" value="${10}"></c:set>

<c:if test="${a1>b}">

<c:out value="${'a1 value is greater than b' }"></c:out>

</c:if>

<h2>--choose when otherwise --</h2>

<c:set var="a2" value="${-2}"></c:set>

<c:choose>

<c:when test="${a2>=100}">

<c:out value="${'a2 value is greater than 100' }"></c:out>

</c:when>

<c:when test="${a2>=0 && a2<100}">

<c:out value="${'a2 value is bw 0 to 99' }"></c:out>

</c:when>

<c:otherwise>

<c:out value="${'a2 is negative' }"></c:out>

</c:otherwise>

</c:choose>

<h2>--forEach --</h2>

<c:set var="a3" value="${0}"></c:set>

<c:forEach var="i" begin="1" end="7">

<c:out value="${a3+i}"></c:out>

</c:forEach>

</body>

</html>

UseBeans

POJO = bean

POJO - setter and getters

<jsp:setProperty>

<jsp:getProperty>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="show.jsp">

name<input type="text" name="user"><br>

password<input type="password" name="password"><br>

age<input type="text" name="age"><br>

<input type="submit">

</form>

</body>

</html>

//pojo

package com.example;

public class User {

private String user;

private String password;

private int age;

public String getUser() {

return user;

}

public void setUser(String user) {

this.user = user;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

}

//show.jsp

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<jsp:useBean id="userid" class="com.example.User"></jsp:useBean>

<jsp:setProperty property="\*" name="userid"/>

properties of user are

<jsp:getProperty property="user" name="userid"/>

<jsp:getProperty property="password" name="userid"/>

<jsp:getProperty property="age" name="userid"/>

</body>

</html>

—-URL Management—------

//index.jsp

<form action="display.jsp">

id<input type="text" name="id"><br>

name<input type="text" name="name"><br>

<input type="submit">

</form>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

    <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1>------------URL management--------------</h1>

<c:set var="id" value="${param.id}"/>

<c:set var="name" value="${param.name}"/>

<h2>----id and name is going to be passed as a parameter to another page---   index.jsp?id=14&name=abcd</h2>

<c:url var="url" value="index.jsp">

<c:param name="id" value="${id}"></c:param>

<c:param name="name" value="${name}"></c:param>

</c:url>

<c:out value="${url}"></c:out>

</body>

</html>

<c:redirect url="<https://www.google.com/>"></c:redirect>

Sql tags

—---------

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="crud.jsp">

Empno<input type="text" name="no"><br>

Empname<input type="text" name="name"><br>

<input type="submit">

</form>

</body>

</html>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

    <%@taglib prefix="sql"  uri="http://java.sun.com/jsp/jstl/sql"    %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<sql:setDataSource var="db" driver="com.mysql.jdbc.Driver" url="jdbc:mysql://localhost:3306/db2" user="root" password="123456"/>

<sql:update dataSource="${db}" var="rows">

insert into employee values(?,?);

<sql:param value="${param.no}"></sql:param>

<sql:param value="${param.name}"/>

</sql:update>

<c:if test="${rows>0}"></c:if>

insertion is successful

<form action="retr.jsp">

<input type="submit" value="getemployees">

</form>

</body>

</html>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

    <%@taglib prefix="sql"  uri="http://java.sun.com/jsp/jstl/sql"    %>

     <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<sql:setDataSource var="db" driver="com.mysql.jdbc.Driver" url="jdbc:mysql://localhost:3306/db2" user="root" password="123456"/>

<sql:query dataSource="${db}" var="rs">

select \* from employee;

</sql:query>

<c:forEach var="table" items="${rs.rows}">

${table.eid} ${table.ename}

</c:forEach>

</body>

</html>

Module 1:

Admin and User

Admin - Login in

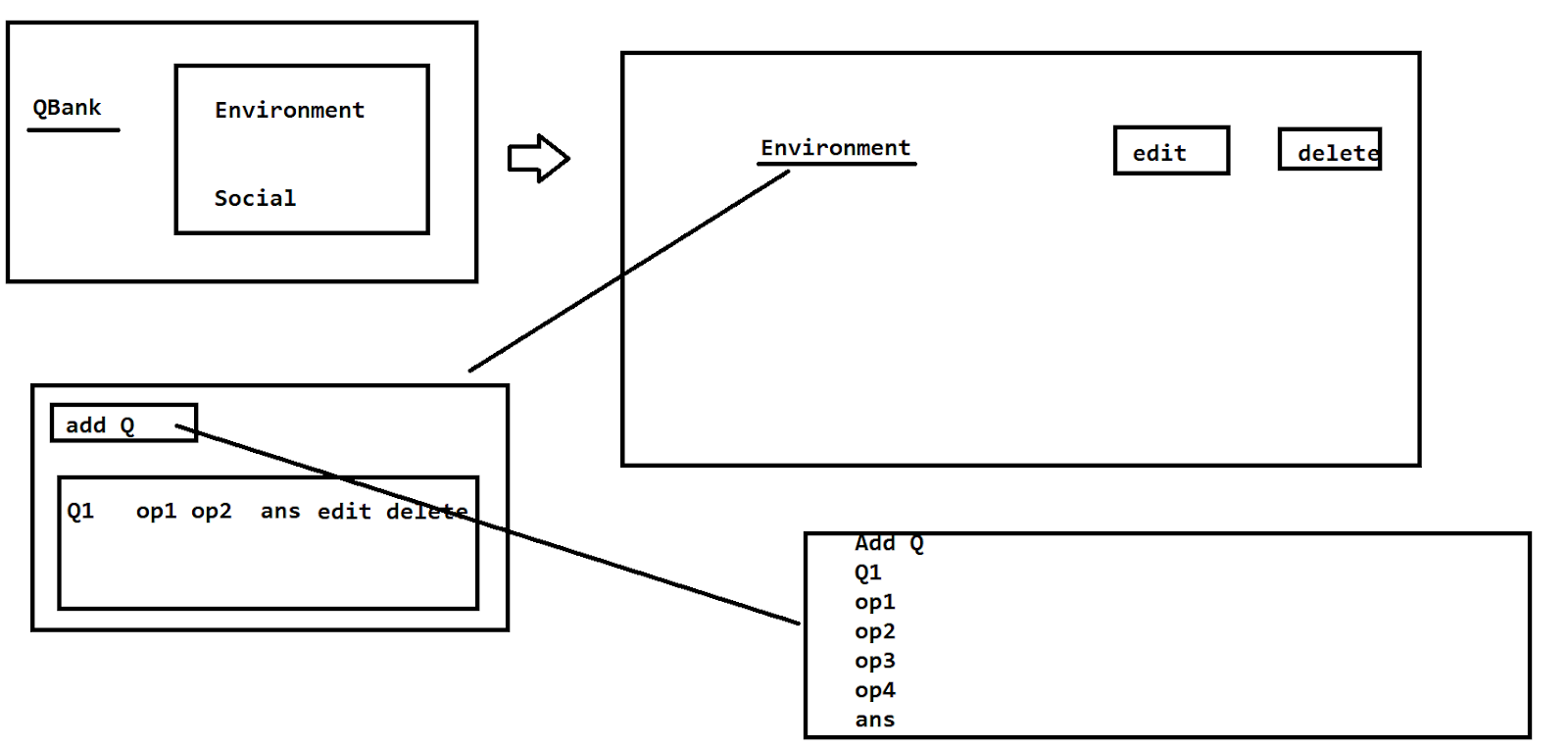
User - Registration and Login

Admin

1. Frame the qbank   [add]
2. Edit and delete of the qs in the qbank
3. Edit and delete the users from the db
4. View of qbank + View of the users
5. Dashboard of the qbank that was attempted by the users

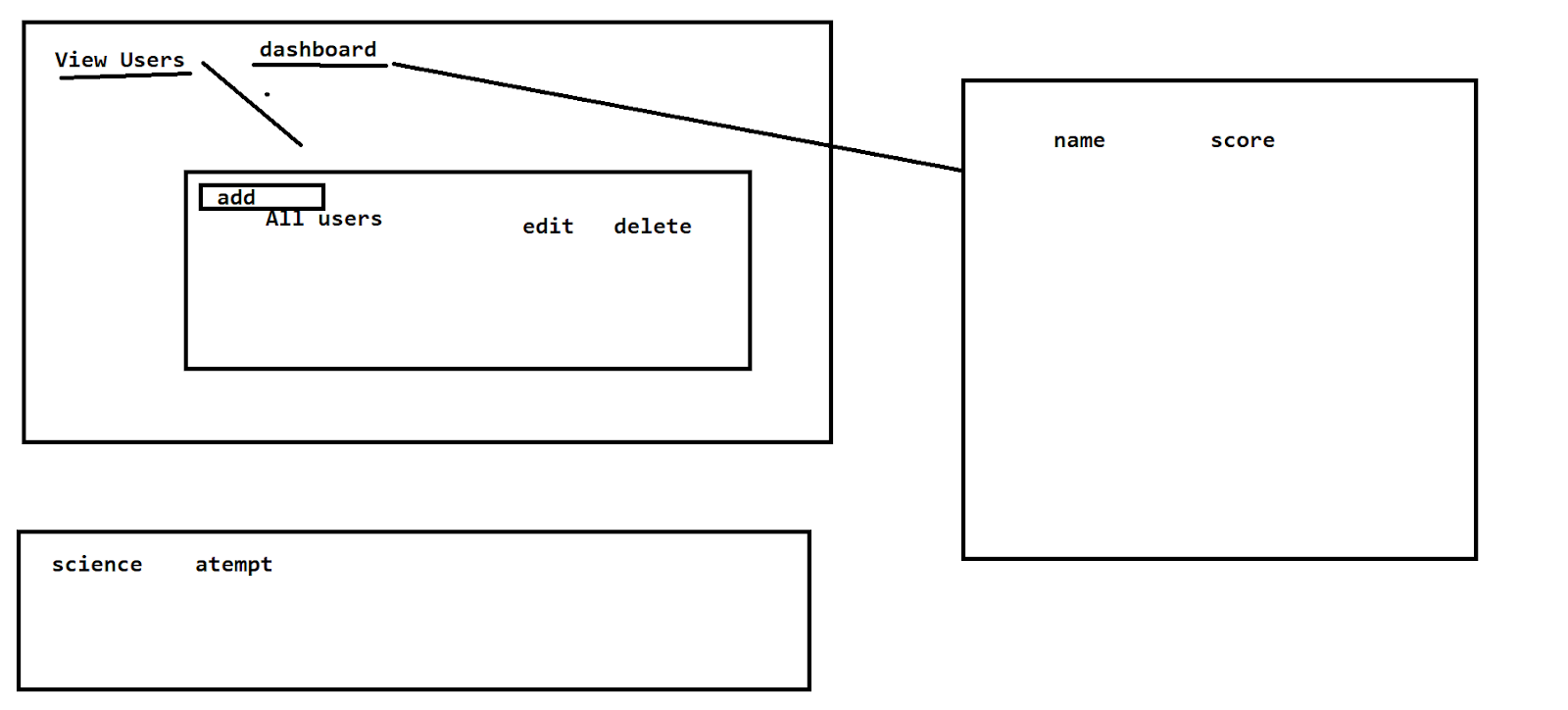
User

1. Qbank view - attempt
2. Viewing the questions -submit
3. Score on the dashboard



1. Add questions - add the questions to the table
2. Edit and delete options for questions
3. Add users
4. Edit and delete users

User - display the questions and calculate the score



Class Quiz{

Q1

Op1

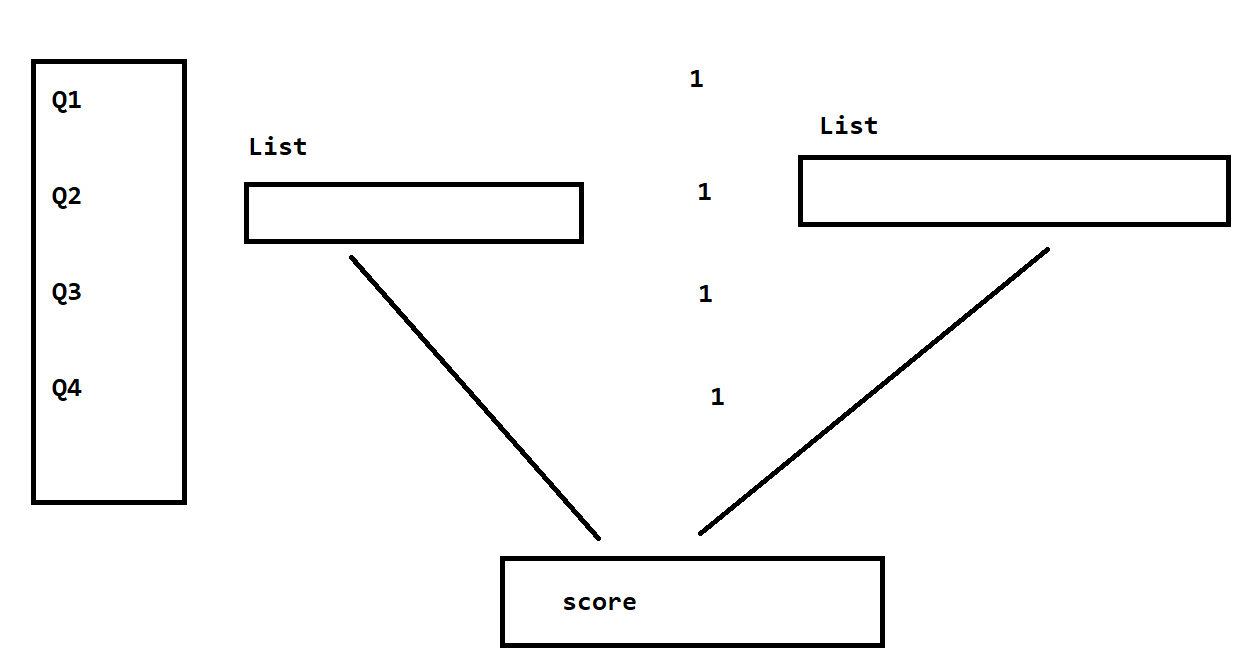
Op2

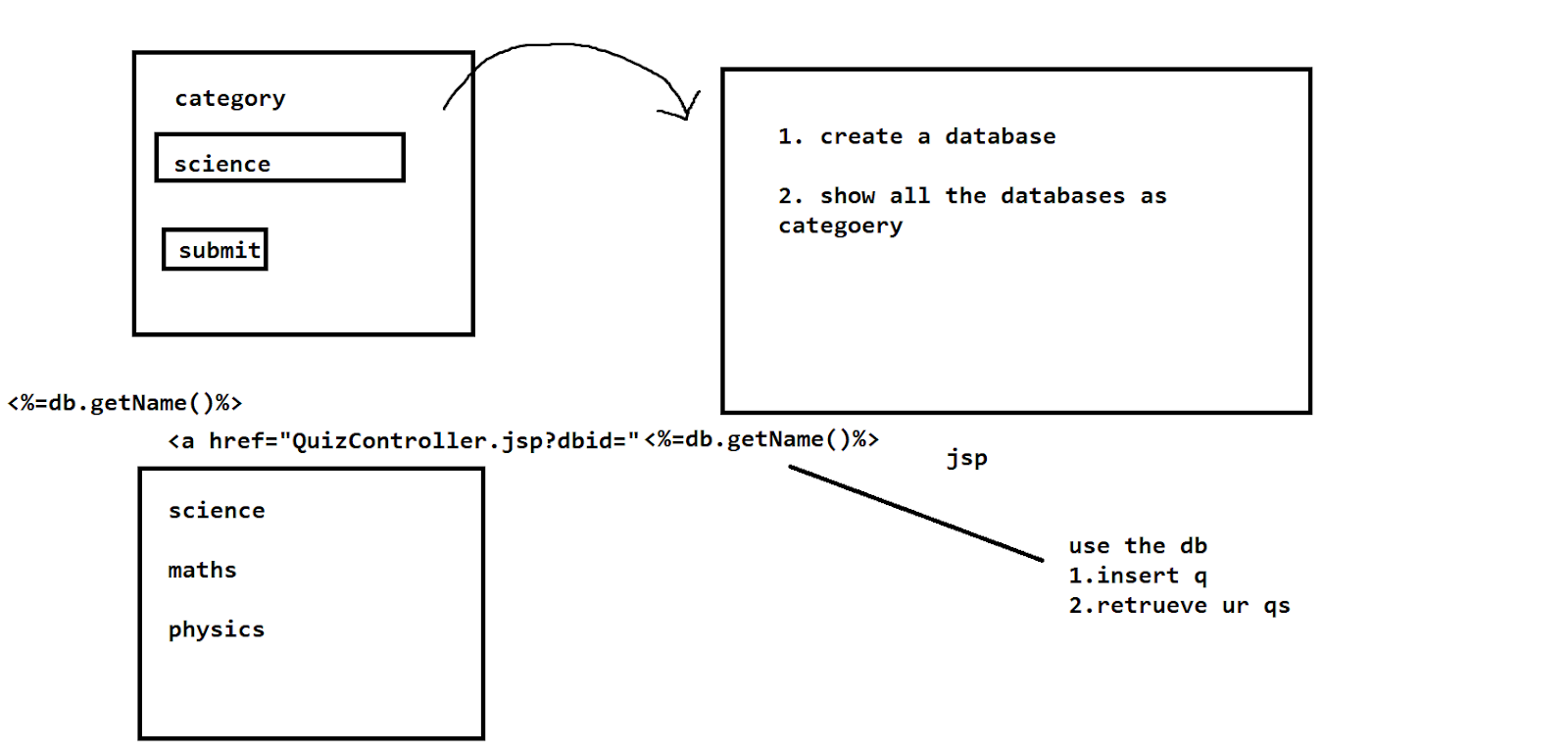
Op3

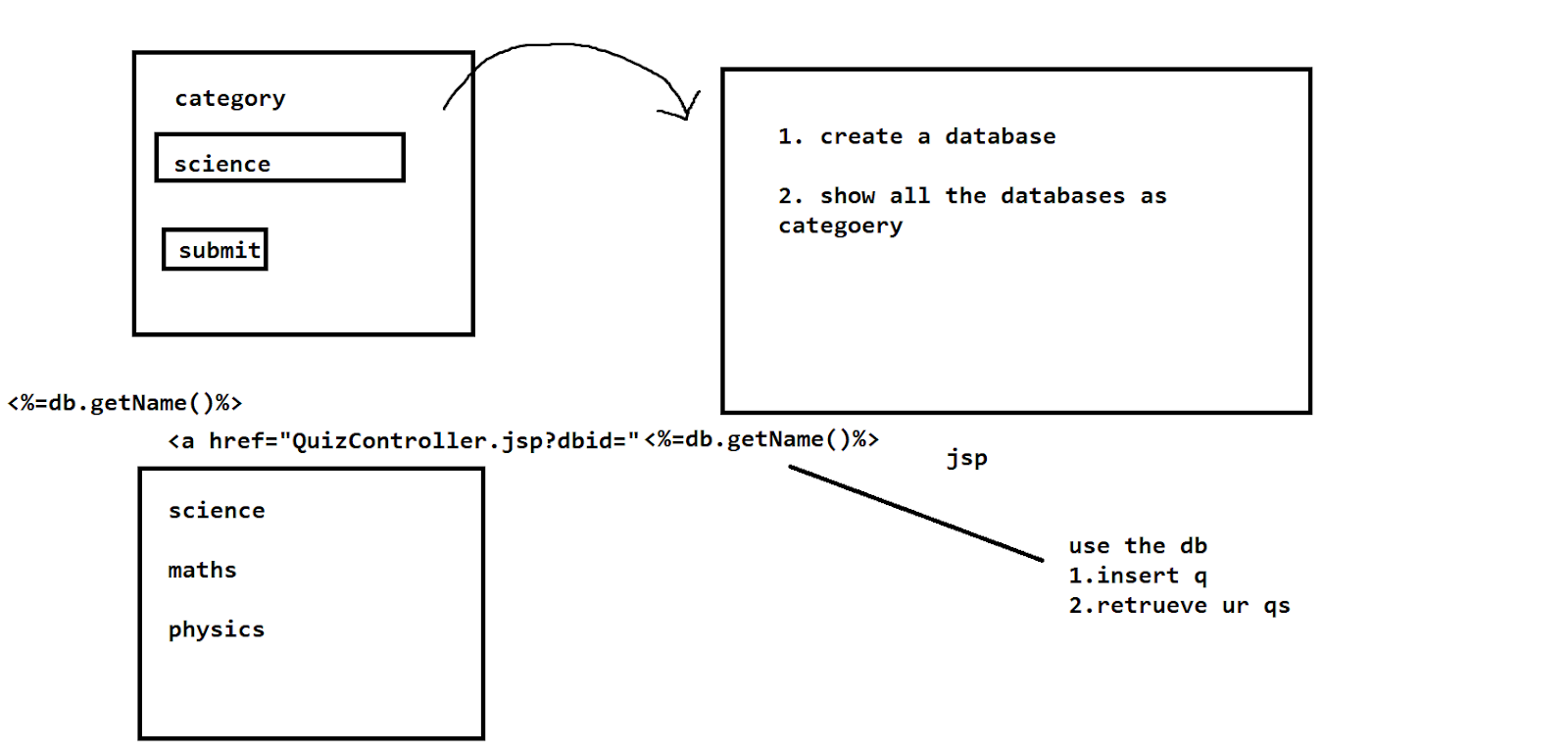
Op4

ans

}







At jdbc

package jdbcdBcreation;

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 dBcreation {

private static final String DRIVER\_CLASS="com.mysql.jdbc.Driver";

//by using jdbc driver - connect to mysql which is a local dB server on the port of 3306 which has db1 as dB

private static final String DB\_URL="jdbc:mysql://localhost:3306/db1";

private static final String USERNAME="root";

private static final String PASSWORD="123456";

public static void main(String[] args) throws ClassNotFoundException, SQLException {

Scanner sc=new Scanner(System.in);

System.out.println("enter the db name ");

String dbname=sc.next();

Class.forName(DRIVER\_CLASS);

//connection with the dB

Connection con=DriverManager.getConnection(DB\_URL,USERNAME,PASSWORD);

Statement st=con.createStatement();

st.executeUpdate("create database "+dbname);

System.out.println("database created !!");

ResultSet rs=st.executeQuery("show databases");

while(rs.next()) {

System.out.println(rs.getString(1));

}

System.out.println("enter the db to be used");

String dbnameuse=sc.next();

st.executeUpdate("use "+dbnameuse);

System.out.println("db selected ");

String sql="create table quiz(q1 varchar(10),op1 varchar(10))";

st.executeUpdate(sql);

System.out.println("table created");

}

}

//hibernate

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

    <session-factory>

        <!--db driver , connection  -->

        <property name="hibernate.connection.driver\_class">com.mysql.jdbc.Driver</property>

        <property name="hibernate.connection.password">123456</property>

        <property name="hibernate.connection.url">jdbc:mysql://localhost/db1</property>

        <property name="hibernate.connection.username">root</property>

        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>

         <!--hibernate config  -->

       <property name="hbm2ddl.auto">update</property>

       <property name="show\_sql">true</property>

        <!--pojo -->

       <mapping class="com.example.pojo.Quiz"/>

    </session-factory>

</hibernate-configuration>

package com.example.pojo;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Quiz {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private int qno;

private String question;

private String options;

public int getQno() {

return qno;

}

public void setQno(int qno) {

this.qno = qno;

}

public String getQuestion() {

return question;

}

public void setQuestion(String question) {

this.question = question;

}

public String getOptions() {

return options;

}

public void setOptions(String options) {

this.options = options;

}

}

package com.example.main;

import java.util.List;

import java.util.Scanner;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

import org.hibernate.boot.Metadata;

import org.hibernate.boot.MetadataSources;

import org.hibernate.boot.registry.StandardServiceRegistry;

import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

import org.hibernate.query.Query;

/\*

StandardServiceRegistry

Metadata

SessionFactory

Session

Transaction

Close the connections

 \* \*/

public class EmpMain {

public static void main(String[] args) {

//SSR is used to map the config file and execute it .

StandardServiceRegistry  ssr=new StandardServiceRegistryBuilder().configure("hibernate.cfg.xml").build();

//Metadata of the xml file is read by this object

Metadata md=new MetadataSources(ssr).getMetadataBuilder().build();

//session-factory- db

SessionFactory sf=md.getSessionFactoryBuilder().build();

//all the crud operations need to be done in Session

Session s=sf.openSession();

//Transaction- perform sql operations and commit it permenantly on the db

Transaction t=s.beginTransaction();

Scanner sc=new Scanner(System.in);

System.out.println("enter the dbname");

String dbname=sc.next();

s.createSQLQuery("create database "+dbname).executeUpdate();

System.out.println("db created");

t.commit();

String sql="show databases";

List<String> databases= s.createSQLQuery(sql).list();

for(String database:databases) {

String dbName=database;

System.out.println(dbName);

}

s.beginTransaction();

System.out.println("enter ur db");

String dbuse=sc.next();

s.createSQLQuery("use "+dbuse).executeUpdate();

//create a table

s.createSQLQuery("create table Quiz(qno int primary key auto\_increment,question varchar(255),options varchar(255))").executeUpdate();

s.getTransaction().commit();

System.out.println("table created");

}

}

–Monday

Spring framework - IoC + DI - JDBC Template , Hibernate Template