PRACTICA SERVLET

JSPs: son páginas web que, además de contener código HTML y hojas de estilo CSS, poseen elementos dinámicos (o scripts) en Java que pueden procesar peticiones Java proveniente de los Servlets para ser mostrados en la página HTML.

Servlets: son piezas de código Java que realiza el procesamiento de datos de lado servidor, y se conectan con los JSPs para entregarle datos que, a su vez, los JPSs mostrarán dentro de la página HTML.

1. Una aplicación JSP/Servlet debe tener un servidor Web y un motor Servlet/JSP para procesar los requerimientos HTTP , y retornar una respuesta HTTP. Como servidor Web se puede utilizar un servidor Tomcat.

2. La mayoría de las aplicaciones Servlet/JSP, utiliza una base de datos para almacenar la data que será utilizada por la aplicación. Por lo general se puede utilizar MySQL (MariaDB), o PostgreSQL.

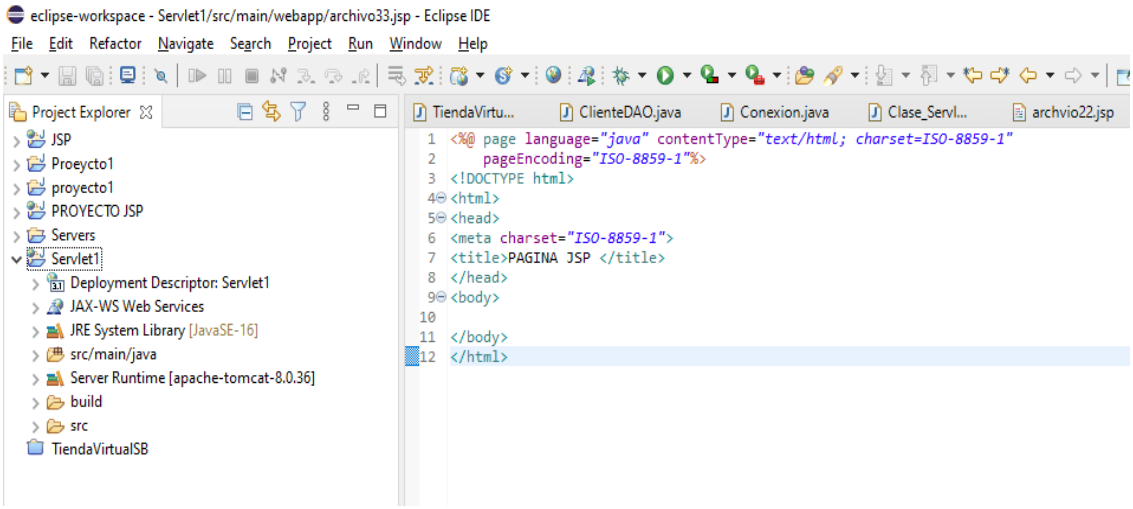
3. Para que el motor del Servlet/JSP opere, este accede al JDK (Java Development Kit), el cual es un software que permite que Java opere y ejecute programas Java.

1. La capa de presentación contiene páginas Web, tanto HTML, como JSPs.

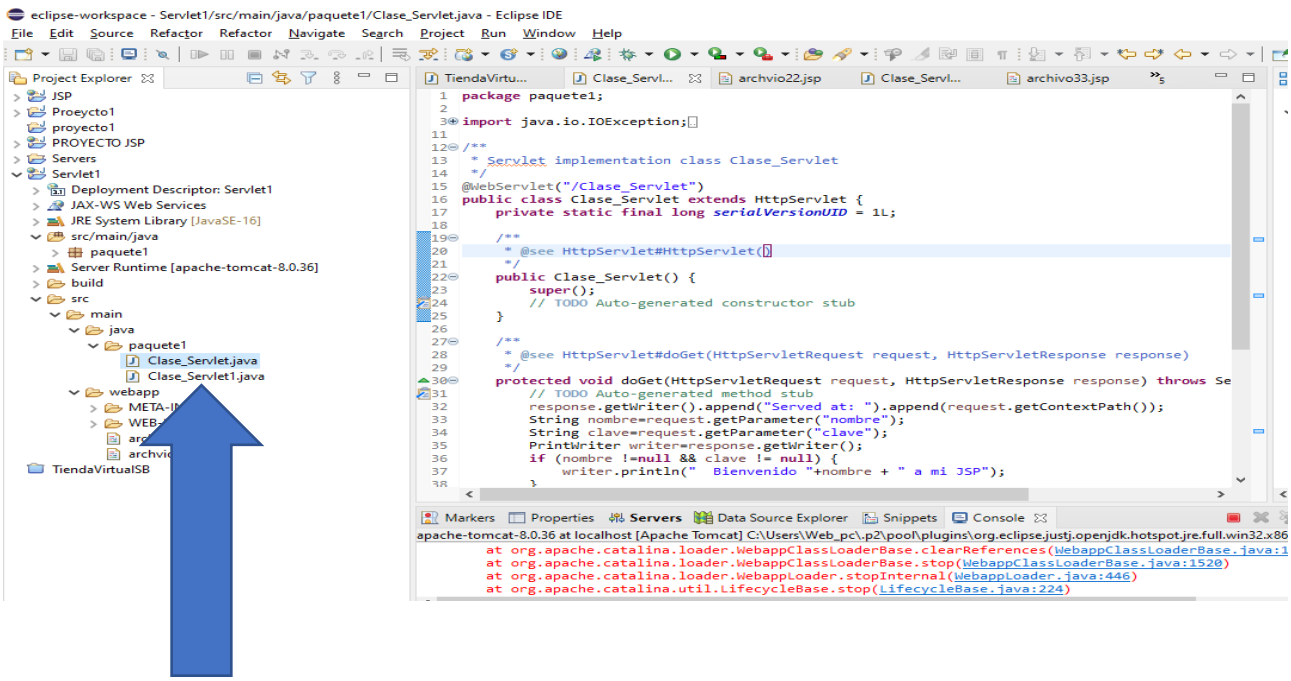
2. La capa de reglas de negocio consiste en Servlets, los cuales procesan la lógica de negocio de la aplicación, utilizando clases Java llamadas Javabeans, u otras clases.

3. La capa de acceso a los datos contiene las clases Java que leen y escriben data que se almacena en bases de datos, archivos, archivos XML, o JSON.

Creamos un Servlet (ver grafica):

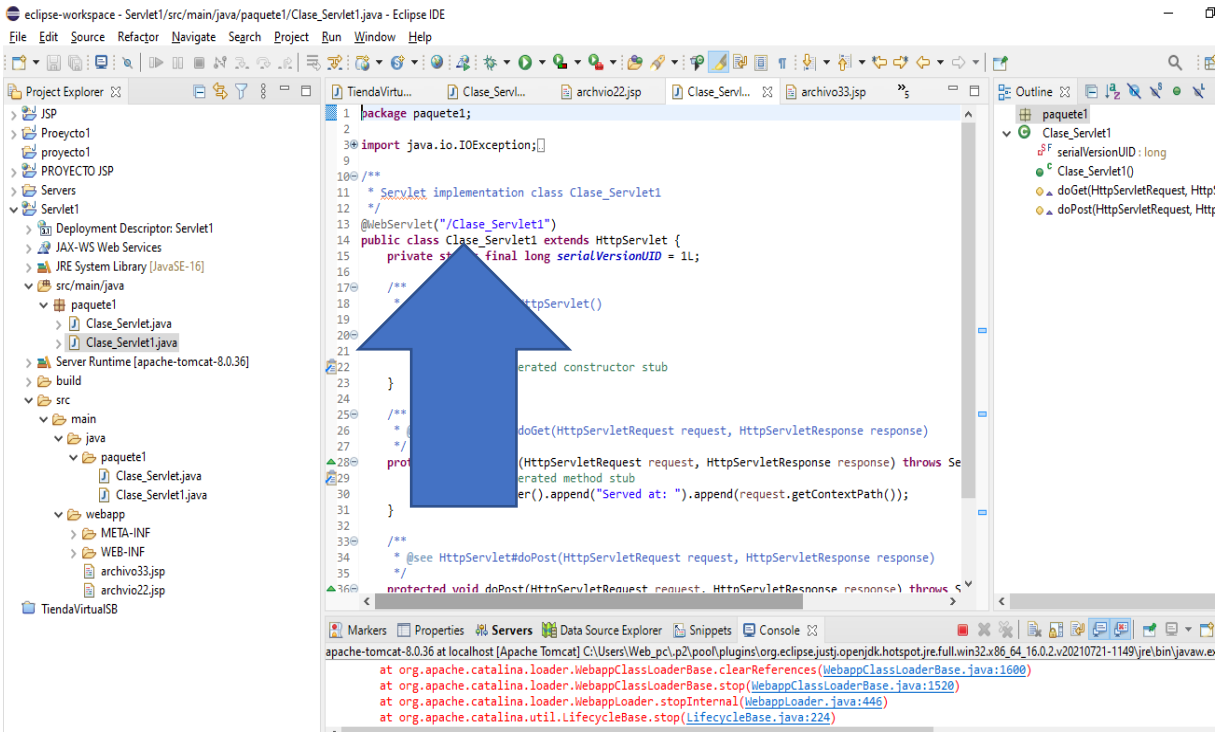


1. Creamos la Clase_Servlet dentro del paquete1



Clase_Servlet.java

Al crear Clase_Servlet.java nos crea el sigueinte codigo:



Vemos que en el codigo aparece un patron de comportamiento definido WebServlet (“/Clase_Servlet”) lo cual indica que es una clase que será un Servlet para las peticiones que se hagan dese cualquier pagina Web o JSP, para este caso desde la página archvio22.jsp.

```
@WebServlet("/Clase_Servlet")

public class Clase_Servlet extends HttpServlet {
	private static final long serialVersionUID = 1L;
```

El metodo (method) de acceso a la informacion del <form> en el archivo22.jsp, se ha colocado GET y al momento de presionar el boton “Enviar”, se pasaran al Servlet definido en la seccion (action) el cual es DemoServlet (tener en cuenta ./ para que se pueda procesar de la forma http://localhost:8080/Servlet/archvio22.jsp?nombre=Luisa Fernanda&cedula=123456), es decir estos son los datos del nombre y cedula visibles en el navegador.

Los metodos **doGet y doPost**, tienen como proposito el procesamineto de la accion que reciba del JSP. En este caso doGet es el que procese, para lo cual en este metodo escribiremos dos instrucciones que permiten obtener los parametros de nombre y cedula, haciendo un Request.GetParameter para procesarlos con un mensaje de vuelta por medio de PrintWriter

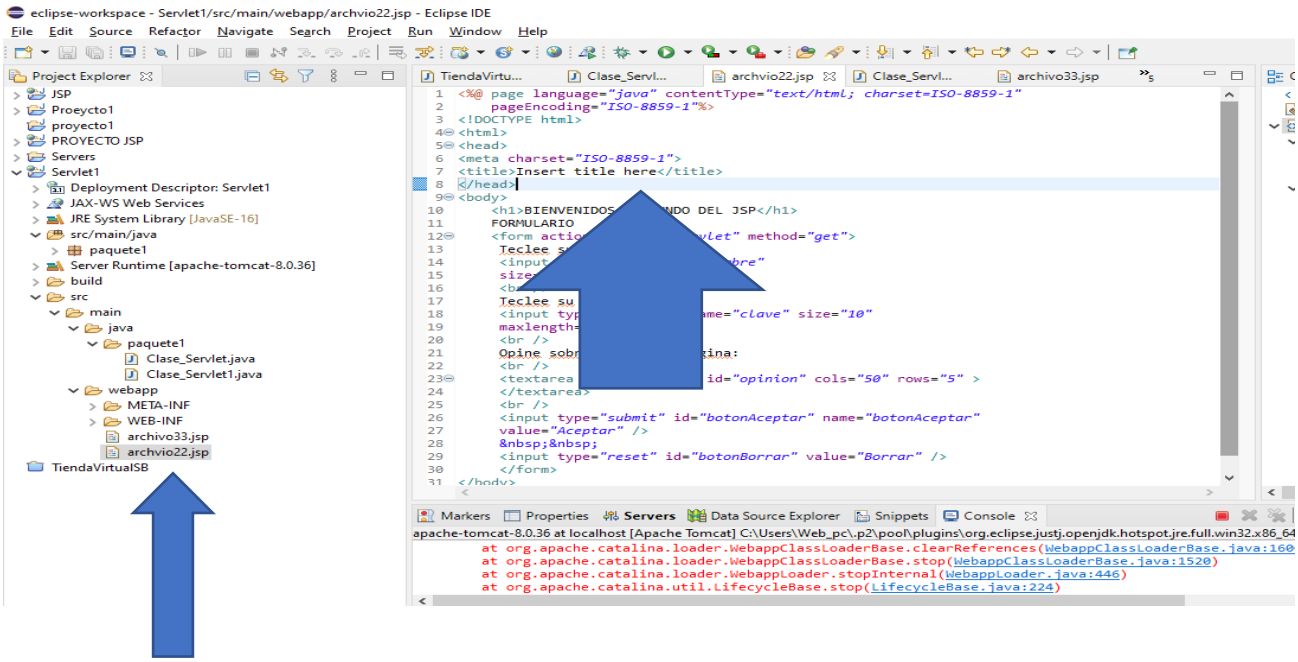
PrintWriter writer=response.getWriter();

Las sentencias:

String nombre= request.getParameter(“nombre”) y

String nombre= request.getParameter(“cedula”) reciben lo que el archivo22.jsp envian por GET.

2. Procedemos a crear el archivo jsp en la carpeta Webapp (Eclipse nos genera este codigo):




```
<%@ 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>BIENVENIDOS AL MUNDO DEL JSP</h1>
```

```
</body>
</html>
```

Guardamos como archivo22.jsp

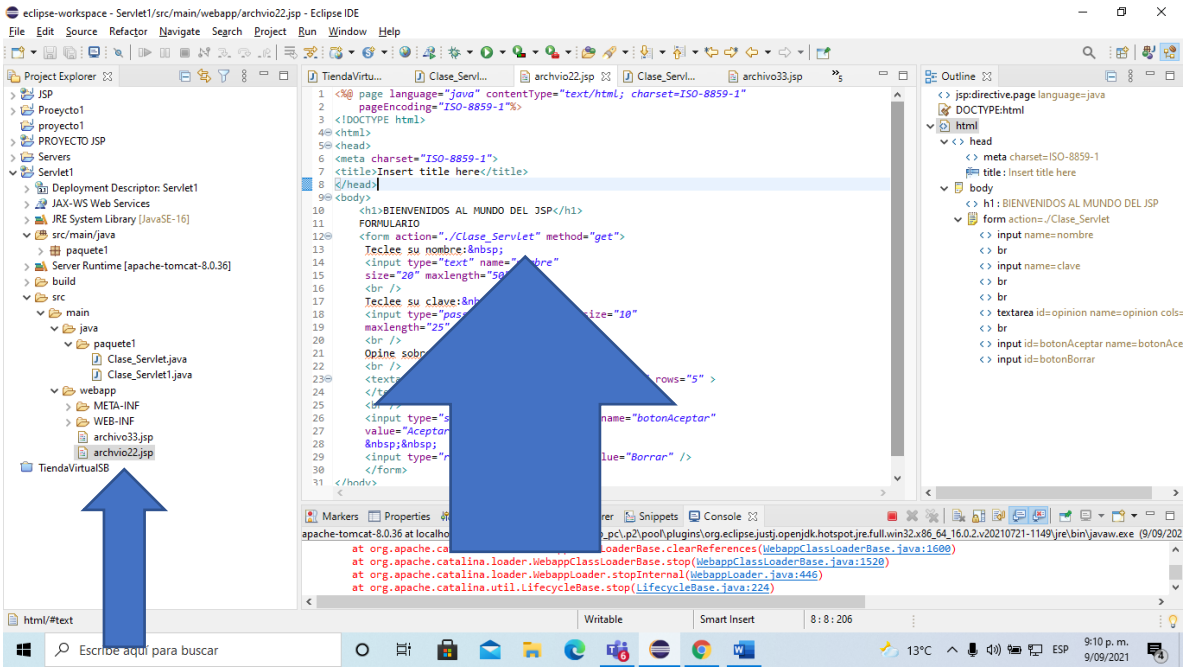
3. Procedemos a incrustar las siguientes líneas de código, que corresponden a un formulario HTML:

```
<form action="./Clase_Servlet" method="get">
 Teclee su nombre: 
 <input type="text" name="nombre"
 size="20" maxlength="50" />
 <br />
 Teclee su clave: 
 <input type="password" name="clave" size="10"
 maxlength="25" />
 <br />
 Opine sobre mi p&aacute;gina:
 <br />
 <textarea name="opinion" id="opinion" cols="50" rows="5" >
 </textarea>
 <br />
 <input type="submit" id="botonAceptar" name="botonAceptar"
 value="Aceptar" />
   
 <input type="reset" id="botonBorrar" value="Borrar" />
 </form>
```

NOTA: algo muy importante es en la etiqueta

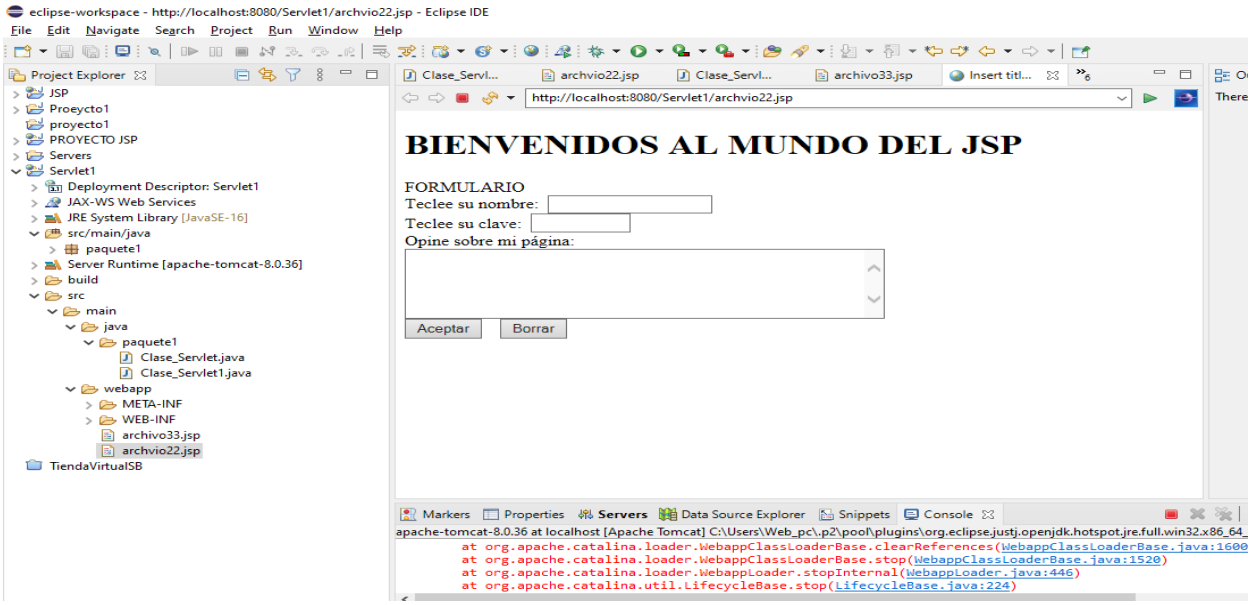
<form action="./Clase_Servlet" method="get"> colocar en action "./Clase_Servlet" que indica el Servlet que va a invocar.

Guardar como archivo22.jsp



archivo22.jsp

4. Lanzamos el archivo22.jsp y nos da el siguiente resultado:



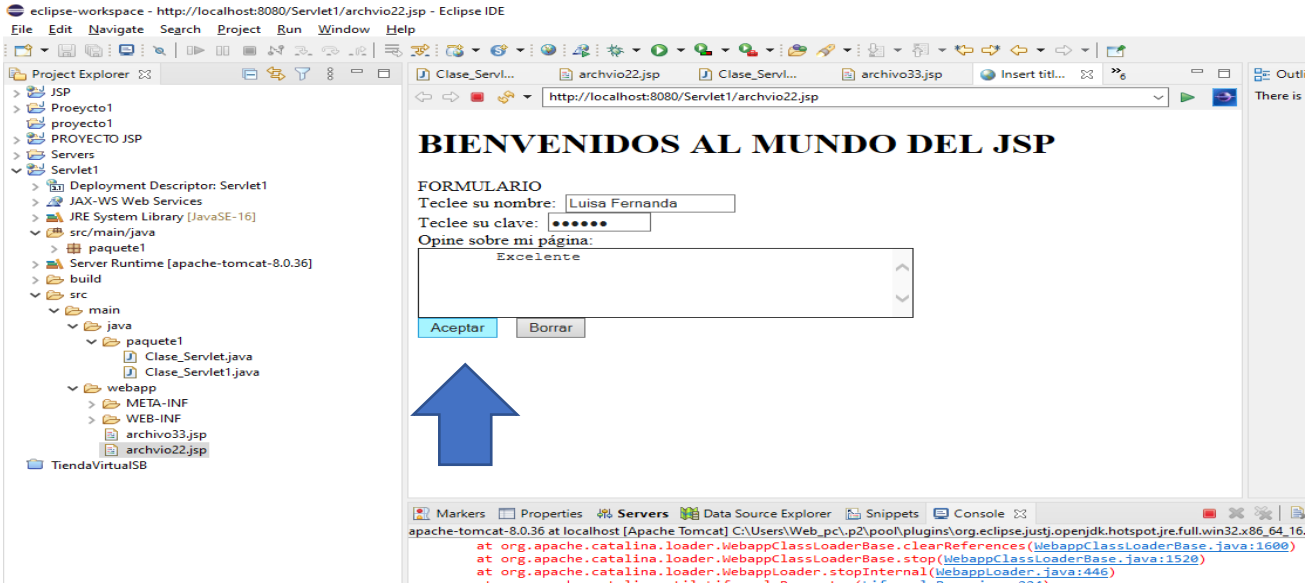
A lo cual procedemos a ingresar:

Teclee su nombre: Luisa Fernanda

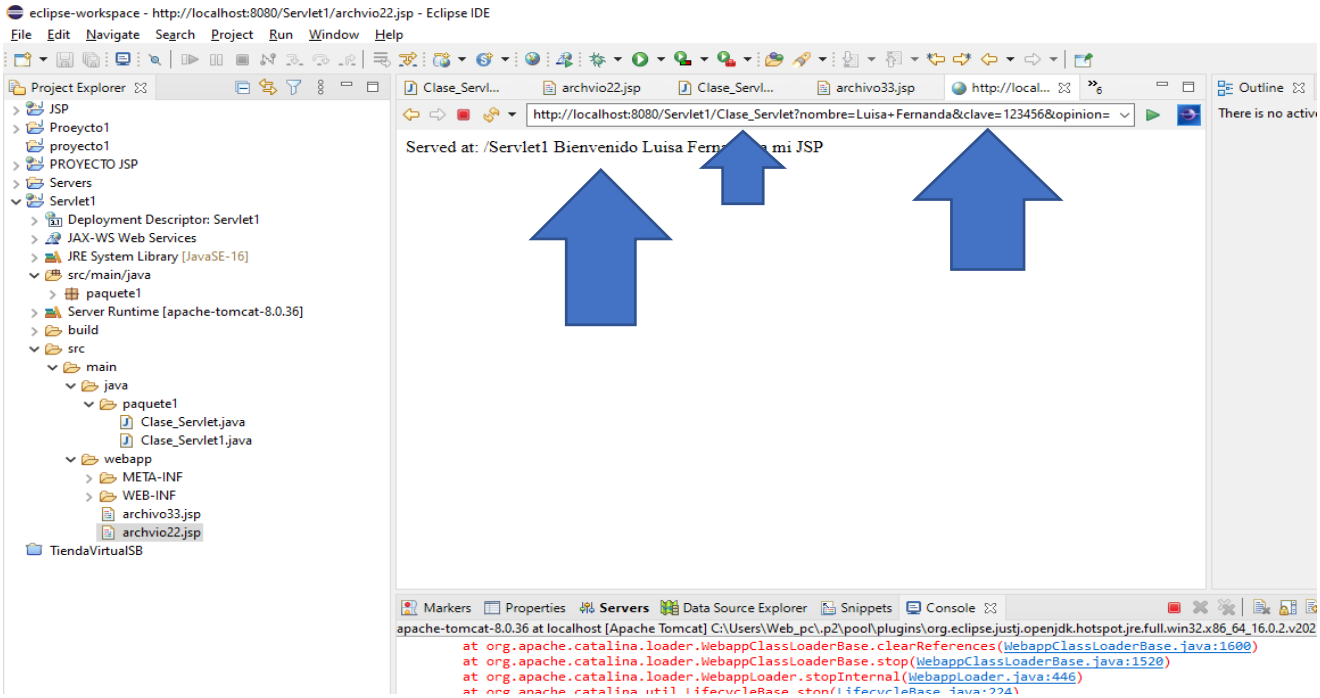
Teclee su clave: *******

Opine sobre mi pagina: Excelente

Boton **Aceptar**



El resultado es:


eclipse-workspace - http://localhost:8080/Servlet1/archvio22.jsp - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Project Explorer
JSP
Proeycto1
proyecto1
PROYECTO JSP
Servers
Servlet1
Deployment Descriptor: Servlet1
JAX-WS Web Services
JRE System Library [JavaSE-16]
src/main/java
paquete1
Server Runtime [apache-tomcat-8.0.36]
build
src
main
java
paquete1
Clase_Servlet.java
Clase_Servlet1.java
webapp
META-INF
WEB-INF
archivo33.jsp
archvio22.jsp
TiendaVirtualSB
Clase_Servl...
archvio22.jsp
Clase_Servl...
archivo33.jsp
http://local...
http://localhost:8080/Servlet1/Clase_Servlet?nombre=Luisa+Fernanda&clave=123456&opinion=
Served at: /Servlet1 Bienvenido Luisa Fern a mi JSP
Outline
There is no activ
Markers
Properties
Servers
Data Source Explorer
Snippets
Console
apache-tomcat-8.0.36 at localhost [Apache Tomcat] C:\Users\Web_pc\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v202
at org.apache.catalina.loader.WebappClassLoaderBase.clearReferences(WebappClassLoaderBase.java:1600)
at org.apache.catalina.loader.WebappClassLoaderBase.stop(WebappClassLoaderBase.java:1520)
at org.apache.catalina.loader.WebappLoader.stopInternal(WebappLoader.java:446)