

Informe de prácticas de ISDCM

Entrega 01

Benny Pérez

Victor Carhuairicra

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona



Contenido

Introducción	1
Decisiones de diseño	1
Repositorios de código consultados	9
Bibliografía consultada	9

Introducción

El informe muestra una descripción de la implementación de una aplicación web bajo el esquema de cliente – servidor, haciendo uso del patrón de diseño MVC (Modelo – Vista - Controlador).

La aplicación web consiste en un entorno que permite gestionar una base de datos de usuarios y videos, donde se pueden crear usuarios de acceso y cada uno de estos usuarios pueden registrar sus videos y consultar la lista de videos existente en la base de datos según el autor, título del video o fecha de creación.

La aplicación web fue desarrollada bajo dos lenguajes de programación: java y html mediante la herramienta Apache Netbeans 12 y la de base de datos fue implementada sobre Java Database, además se utilizó Glassfish5.1.0 como aplicación servidor dónde poder alojar la aplicación web.

Decisiones de diseño

Siguiendo el patrón de MVC la arquitectura de la aplicación web está organizada en páginas (JSP), controladores (servlets) y modelos, en la figura 01 se muestra un esquema de la arquitectura, mientras que en la figura 02 se puede observar la organización del proyecto en Netbeans.

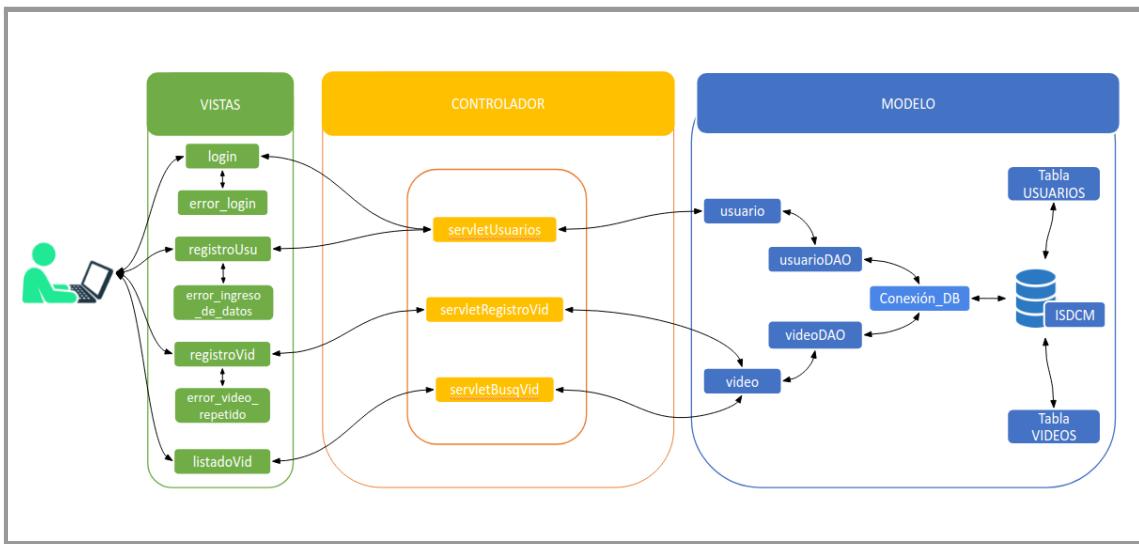


Figura 01

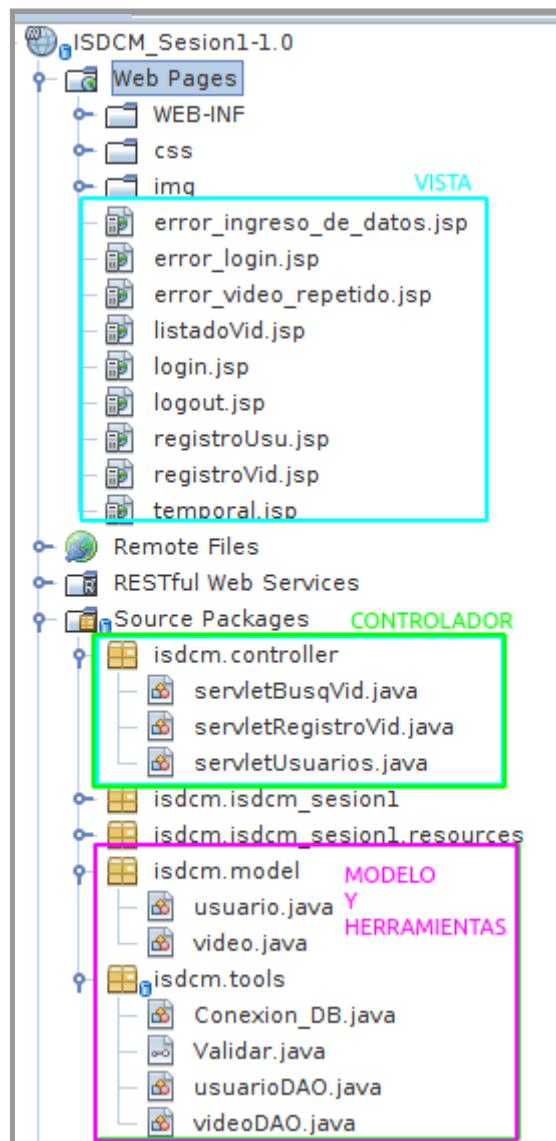


Figura 02

Adicionalmente, para dar formato de las vistas se ha utilizado ficheros CSS que se encuentran en el directorio CSS del proyecto, esto se muestra en la figura 03.

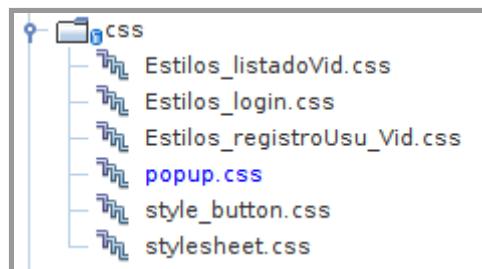


Figura 03

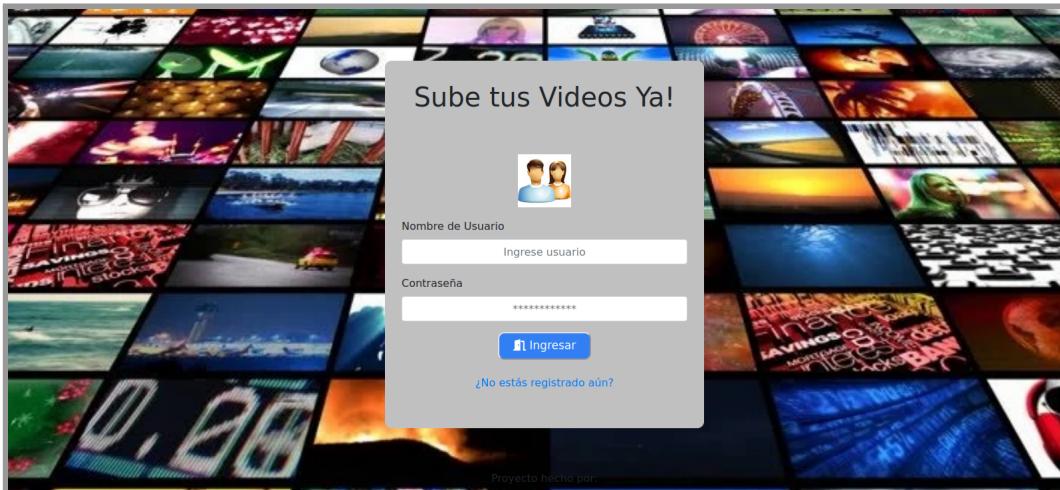
Vista login:

Figura 04

La aplicación web inicia en la vista login, donde esta interfaz permite realizar dos acciones: la primera es dar acceso a los usuarios que ya están registrados mediante la validación de un usuario y contraseña.

En esta vista se han tomado medidas de seguridad para validar el usuario y contraseña del usuario, el código de esta validación se muestra en la figura 05.

```

public int validar(usuario usu){
    r=0;
    String sql="Select * from usuarios where NOMBRE_DE_USUARIO=? and CONTRASENHA=?";
    try{
        con=cn.getConnection();
        ps=con.prepareStatement(sql);
        ps.setString(1,usu.getNombre_de_usuario());
        ps.setString(2,usu.getContraseña());
        rsps.executeQuery();

        while(rs.next()){
            r=r+1;
            usu.setNombre_de_usuario(rs.getString("NOMBRE_DE_USUARIO"));
            usu.setContraseña(rs.getString("CONTRASENHA"));
        }
        if(r==1){
            return 1;
        }else{
            return 0;
        }
    }catch (Exception e){
        System.out.println(e.getStackTrace());
        return 0;
    }
}

protected void processRequest(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
String accion=request.getParameter("accion");
if(accion.equals("Ingresar")){
    String nombre=request.getParameter("txtnom");
    String contraseña=request.getParameter("password");
    usu.setNombre_de_usuario(nombre);
    usu.setContraseña(contraseña);
    r=dao.validar(usu);
    if(r==1){
        request.getSession().setAttribute("userName",nombre);
        request.getRequestDispatcher("listadoVid.jsp").forward(request, response);
    }else{
        request.getRequestDispatcher("error_login.jsp").forward(request, response);
    }
} else {
    request.getRequestDispatcher("login.jsp").forward(request, response);
}
}

```

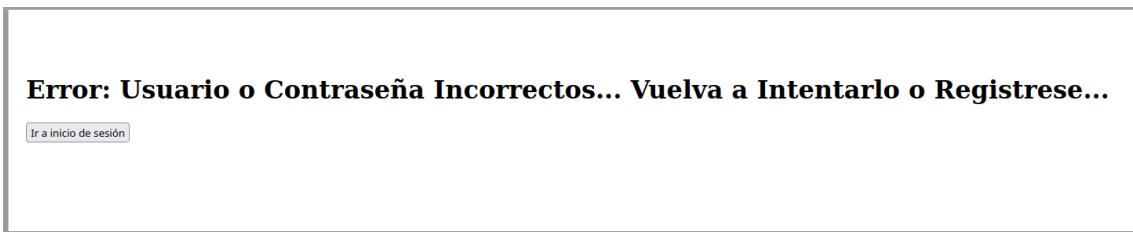


Figura 05

La segunda acción consiste en registrar un usuario nuevo mediante el redireccionamiento a la vista de registroUsu.

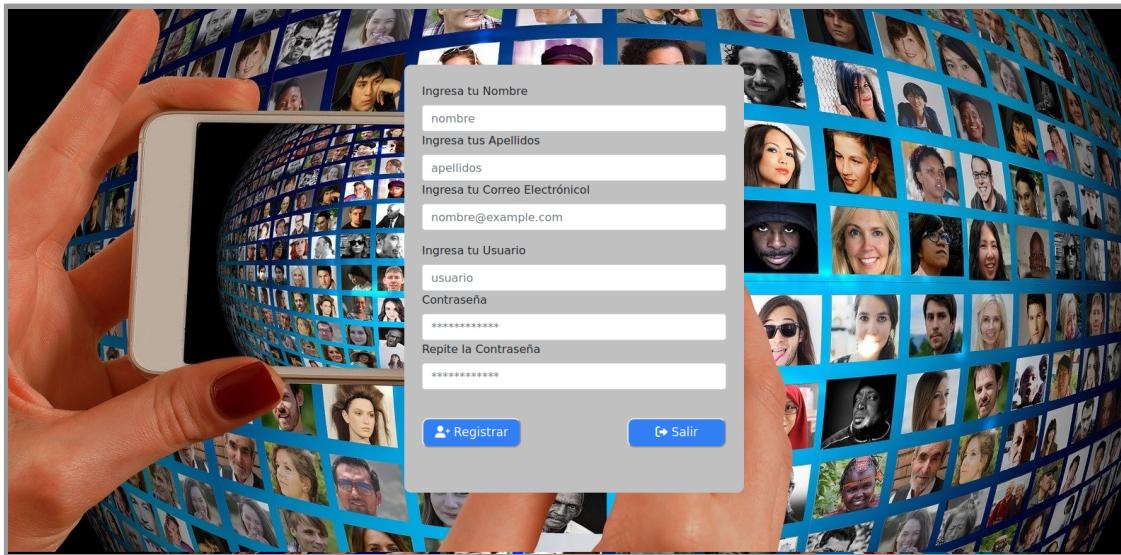
Vista registroUsu:

Figura 06

Esta interfaz permite al nuevo usuario registrarse en la aplicación web, direccionando los datos de los campos hacia la tabla USUARIO de la base de datos ISDCM.

El nombre de usuario que utiliza para identificarse dentro de la aplicación web debe ser único, es por ello que cuando un usuario nuevo se registra se debe hacer una validación del atributo “nombre_de_usuario”, las líneas de código para este control se muestran en la figura 07.

```

public int validar_reg(usuario usu){
    r=0;

    String sql="Select * from usuarios where NOMBRE_DE_USUARIO=?";

    try{
        con=cn.getConnection();
        ps=con.prepareStatement(sql);
        ps.setString(1,usu.getNombre_de_usuario());
        rs=ps.executeQuery();

        while(rs.next()){
            r=r+1;
            usu.setNombre_de_usuario(rs.getString("NOMBRE_DE_USUARIO"));

        }
        if(r==1){
            return 1;
        }else{
            return 0;
        }

    }catch (Exception e){
        System.out.println(e.getStackTrace());
        return 0;
    }
}

```

```

protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    String nombre=request.getParameter("name");
    String apellido=request.getParameter("surname");
    String correo_electronico=request.getParameter("email");
    String nombre_usuario=request.getParameter("username");
    String contrasenya=request.getParameter("password");
    String contrasenha2=request.getParameter("password2");
    String accion=request.getParameter("action");
    if(accion.equals("register")){
        usu_reg.setNombre_de_usuario(nombre_usuario);

        r=dao_reg.validar_reg(usu_reg);
        if(r==0){
            if(contrasenya.equals(contrasenha2)){

                usuario usul = new usuario();
                usul.setNombre(nombre);
                usul.setApellido(apellido);
                usul.setCorreo_electronico(correo_electronico);
                usul.setNombre_de_usuario(nombre_usuario);
                usul.setContrasenya(contrasenya);
                try {
                    dao.registerusu(usul);
                } catch (Exception e) {
                    //TODO: handle exception
                    e.printStackTrace();
                }
                response.sendRedirect("login.jsp");
            } else {
                RequestDispatcher rd = request.getRequestDispatcher("registroUsu.jsp");
                rd.forward(request, response);
            }
        }else{
            response.sendRedirect("error_ingreso_de_datos.jsp");
        }
    } else {
        response.sendRedirect("registroUsu.jsp");
    }
}

```

¡Usuario ya existe!

El nombre de usuario ya existe, vuelva a intentarlo con otro nombre de usuario...

[Registro de Usuario](#)

Figura 07

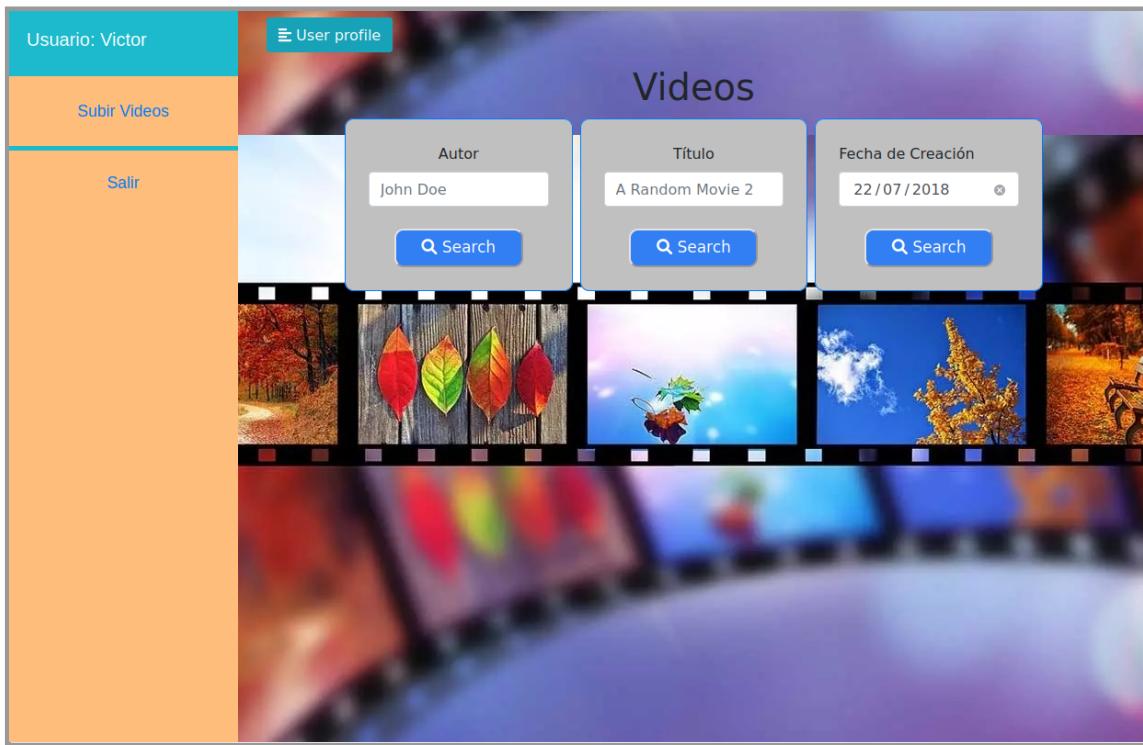
Vista listadoVid:

Figura 08

Después de validar los atributos de usuario y contraseña de un usuario que ya está registrado la aplicación web muestra una interfaz donde el usuario puede realizar búsquedas de vídeos en la base de datos. Para ello la aplicación le brinda al usuario tres opciones de búsqueda:

- Búsqueda por Autor: realiza una búsqueda en la tabla VIDEOS de la base datos tomando como parámetro de búsqueda el nombre del autor del video.
- Búsqueda por Título: realiza una búsqueda en la tabla VIDEOS de la base datos tomando como parámetro de búsqueda el nombre del título del video.
- Búsqueda por Fecha: realiza una búsqueda en la tabla VIDEOS de la base datos tomando como parámetro de búsqueda la fecha de registro del video.

En la figura 09 se muestra el código para estos tres tipos de búsqueda.

```

public void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    System.out.println(request);
    String action = request.getParameter("action");

    String parametro;
    String value;

    switch (action) {
        case "search-autor":
            parametro = "AUTOR";
            value = request.getParameter("author");
            search(request, response, parametro, value);
            break;
        case "search-titulo":
            parametro = "TITULO";
            value = request.getParameter("title");
            search(request, response, parametro, value);
            break;
        case "search-fecha":
            parametro = "FECHA_DE_CREACION";
            value = request.getParameter("date");
            search(request, response, parametro, value);
            break;
        case "change-search":
            log("change-search");
            request.getSession().removeAttribute("videos_list");
            response.sendRedirect("listadoVid.jsp");
            break;
        default:
            break;
    }
}

public static List<Video> getVideos(String parametro, String value) {
    ArrayList<Video> videos = new ArrayList<Video>();
    try {
        con = cn.getConnection();
        ps = con.prepareStatement("Select * from videos where " + parametro + " = ?");
        ps.setString(1, value);
        rs = ps.executeQuery();
        while (rs.next()) {
            Video video = new Video();
            video.setTitulo(rs.getString("TITULO"));
            video.setAutor(rs.getString("AUTOR"));
            video.setFecha_creacion(rs.getDate("FECHA_DE_CREACION"));
            video.setDuracion(rs.getTime("DURACION"));
            video.setReproducciones(rs.getInt("NUMERO_DE_REPRODUCCIONES"));
            video.setDescripcion(rs.getString("DESCRIPCION"));
            video.setFormato(rs.getString("FORMATO"));
            videos.add(video);
        }
    } catch (SQLException e) {
        printSQLException(e);
    }
    return videos;
}

```

Figura 09

Esta vista también le permite al usuario registrar un nuevo video en la aplicación redireccionando al usuario a la vista registroVid.

Vista registroVid

Mediante esta interfaz el usuario podrá registrar un nuevo video completando los campos que se muestran en la figura 10.

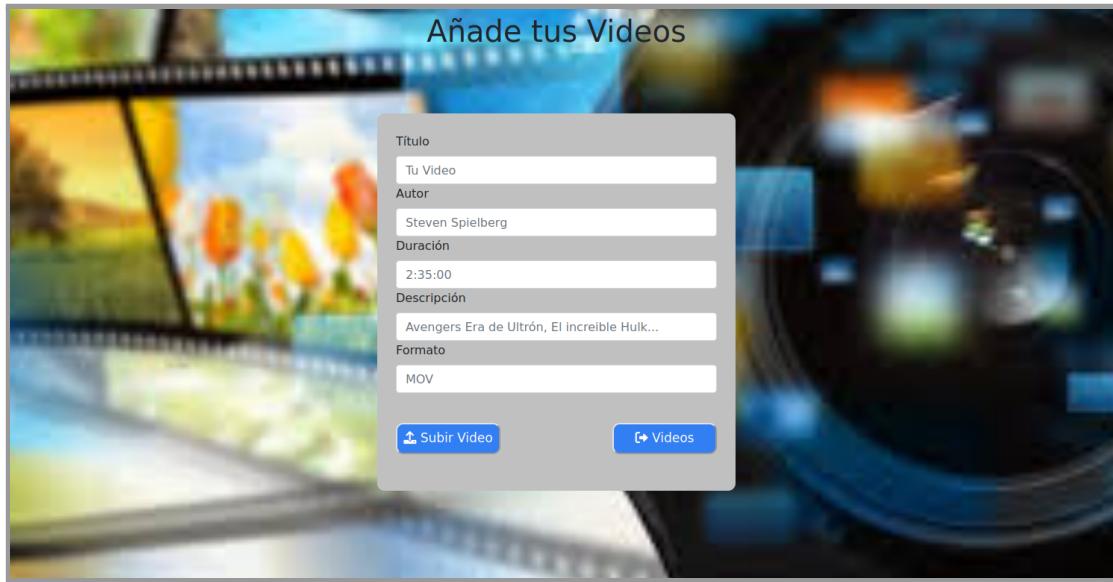


Figura 10

Aquí es importante garantizar que el nombre del video no se repita, para ello se ha implementado un control para validar que el Título de un nuevo video, no exista en la tabla VIDEO de la base de datos ISDCM. El código para este control se muestra en la figura 11.

```

protected void addVideo(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException{
    log("Anadiendo Video");
    String titulo = request.getParameter("titulo");
    String autor = request.getParameter("autor");
    //String fecha_cre = request.getParameter("fecha_de_creacion");
    //ate fecha = Date.valueOf(fecha_cre);
    String tiempo = request.getParameter("duracion");
    Time duracion = Time.valueOf(tiempo);
    String descripcion = request.getParameter("descripcion");
    String formato = request.getParameter("formato");

    String accion=request.getParameter("action");
    if(accion.equals("add-video")){
        video_reg.setTitulo(titulo);

        r=videoDAO_reg.validar_vid(video_reg);
        if(r==1){
            video1.setTitulo(titulo);
            video1.setAutor(autor);
            video1.setDuracion(duracion);
            video1.setDescripcion(descripcion);
            video1.setFormato(formato);
            try {
                videoDAO.registrovid(video1);
            } catch (ClassNotFoundException ex) {
                Logger.getLogger(servletRegistroVid.class.getName()).log(Level.SEVERE, null, ex);
            }
            List<video> videos = videoDAO.getVideos("TITULO",titulo);
            request.getSession().setAttribute("videos_list", videos);
            response.sendRedirect("listadoVid.jsp");
        }else{
            log("Video Repetido");
            request.getSession().removeAttribute("videos_list");
            response.sendRedirect("error_video_repetido.jsp");
        }
    } else {
        response.sendRedirect("listadoVid.jsp");
    }
}

public int validar_vid(video vid){
    r=0;

    String sql="Select * from videos where TITULO=?";

    try{
        con=cn.getConnection();
        ps=con.prepareStatement(sql);
        ps.setString(1,vid.getTitulo());
        rs=ps.executeQuery();

        while(rs.next()){
            r+=1;
            vid.setTitulo(rs.getString("TITULO"));
        }
        if(r==0){
            return 1;
        }else{
            return 0;
        }
    }catch (Exception e){
        System.out.println(e.getStackTrace());
        return 0;
    }
}

```

Error: Video ya existe... Vuelva a cargar el video...

[Ir a Registro de Video](#)

Figura 11

Repositorios de código consultados

- <https://github.com/danielniko/SimpleJspServletDB>
- <https://github.com/hlk-1135/Blog-jsp-servlet-mysql>-
- [https://github.com/RameshMF/servlet-tutorial/tree/master/jsp-servlet-jdbc-mysql-exa
mple](https://github.com/RameshMF/servlet-tutorial/tree/master/jsp-servlet-jdbc-mysql-example)

Bibliografía consultada

- https://www.youtube.com/channel/UCdulls-x_xRd1ezwJZR9ww
- <https://www.youtube.com/watch?v=OkJYtw1mPeU>
- <https://www.youtube.com/watch?v=c-4wFMGFuCg>
- <https://www.w3schools.com/html/default.asp>
- <https://www.w3schools.com/java/default.asp>
- <https://www.w3schools.com/css/default.asp>