



Facultad de Ingeniería

Desarrollo web Integrado - Sección 18186

TITULO

Aplicación Web para la Simulación de Financiamiento y Cotización de Automóviles

Toyota

Integrantes

Alarcon Oroncuy, Claudia Isabel U17200025

Bendezú Cucho, Abel Angel U22222206

Jaramillo Rojas Moisés Alonso U19220767

Melgarejo Vásquez, Maddiel Iroska U22240907

Camargo Pumahuacre, Aaron Gustavo U20307837

Docente

Chicana Aspajo Jorge

Lima, Perú – 2025

1. [Introducción](#)

2. [Capítulo 1](#): Aspectos generales

- 1.1 Nombre del negocio
- 1.2 Antecedente y breve historia
- 1.3 Rubro del negocio
- 1.4 Información cuantitativa (opcional)
- 1.5 Misión y visión
- 1.6 Problemática que afronta
- 1.7 Alternativas de solución
- 1.8 Solución elegida y justificación

3. [Capítulo 2](#): Alcance

- 2.1 Requerimientos funcionales
- 2.2 Requerimientos no funcionales
- 2.3 Funcionalidades adicionales exigidas

4. [Capítulo 3](#):

- 3.1 Implementación de patrón MVC
- 3.2 Funcionalidad del sistema
- 3.3 Diseño de la base de datos con al menos 6 tablas relacionadas
- 3.4 Diagramas de flujo del sistema

5. [Capítulo 4](#):

- 4.1 Realizar operaciones con la base de datos aplicando el patrón MVC.
- 4.2 Debe realizar operaciones CRUD haciendo uso de DataTable.
- 4.3 Debe aplicar al menos 4 operaciones haciendo uso de AJAX.
- 4.4 Manejo de sesiones.
- 4.5 Consumir un servicio web para lectura (consulta y/o listado).

- 4.6 Consumir un servicio web para escritura (agregar y/o modificar).

6. [Glosario](#)

7. [Bibliografía](#)

8. Anexos

Introducción

El presente informe describe el desarrollo de una aplicación web orientada a la simulación de financiamiento para la adquisición de vehículos Toyota.

La solución busca responder a la creciente necesidad de los clientes de contar con herramientas digitales que permitan calcular de manera rápida y confiable sus cuotas, considerando inicial, plazo y tasa de interés.

Este proyecto se enmarca en el curso de Diseño Web Integrado, donde se aplican principios de programación web, arquitectura MVC, bases de datos relacionales y generación de reportes

Capítulo 1: Aspectos generales

Global cars – Plataforma de Financiamiento.

En la actualidad, gran parte de los concesionarios de autos en el país operan de manera tradicional, mediante locales físicos donde los asesores gestionan cotizaciones y financiamientos. Sin embargo, la tendencia digital ha impulsado la creación de plataformas online que permiten al cliente informarse y realizar procesos de manera remota.

El negocio pertenece al rubro automotriz, específicamente en la venta y financiamiento de vehículos de la marca Toyota.

Misión y visión

- Misión: Brindar a los clientes una plataforma digital confiable para simular y obtener cotizaciones de financiamiento de vehículos Toyota, garantizando transparencia y accesibilidad.
- Visión: Convertirse en la principal plataforma web de financiamiento automotriz del país, ampliando sus servicios hacia una concesionaria digital completa.

Problemática que afronta

Los clientes no cuentan con herramientas digitales que les permitan calcular de manera clara y precisa las cuotas de financiamiento. Esto genera retrasos, sobrecarga de consultas al área de asesores y pérdida de potenciales ventas.

Alternativas de solución

- Desarrollar una plataforma web con simulador de financiamiento y generación de cotizaciones en PDF.

1.7 Solución elegida y justificación del proyecto

Se optó por la tercera alternativa: una plataforma web con simulador.

La decisión se justifica porque permite:

- Brindar información inmediata y transparente al cliente.
- Reducir la carga de trabajo en asesores.
- Incrementar la tasa de conversión de consultas en ventas.
- Posicionar a la empresa en el mercado digital.

Capítulo 2: Alcance

2.1 Requerimientos funcionales

- Simular cuotas en función de precio, inicial, plazo y tasa.
- Mostrar catálogo básico de vehículos Toyota.
- Generar pre-cotización en formato PDF.
- Permitir gestión de modelos y tasas por parte del asesor.
- Registrar simulaciones en la base de datos.

2.2 Requerimientos no funcionales

- La aplicación debe ser responsive (usuable en PC, tablet y móvil).
- El sistema debe utilizar el patrón MVC.
- La base de datos debe ser MySQL.
- El código debe ser mantenable y documentado.

2.3 Funcionalidades adicionales exigidas

- Layout del sitio web: cabecera, menú de navegación, pie de página.

localhost:8081/ProyDesWeb2/

GLOBAL CARS Administrador

Tu carro ideal, con el plan perfecto

Financia tu vehículo con cuotas accesibles, asesoría personalizada y aprobación rápida.

Solicita tu financiamiento

¿Por qué elegir GLOBAL CARS?

	Cuotas Flexibles Planes adaptados a tu presupuesto, sin sorpresas ocultas.
	Aprobación Rápida Evaluación en minutos y aprobación personalizada para cada cliente.
	Variedad de Modelos Elige entre sedanes, SUV, pickups y más, todos disponibles para financiamiento.

Modelos de Carros Disponibles

Toyota
Carro con buen modelo
[S/. 569210.0](#)
[Ver más](#)

localhost:8081/ProyDesWeb2/

Simula tu financiamiento

Calculadora de cuotas

Precio del carro (S/.)

Número de cuotas

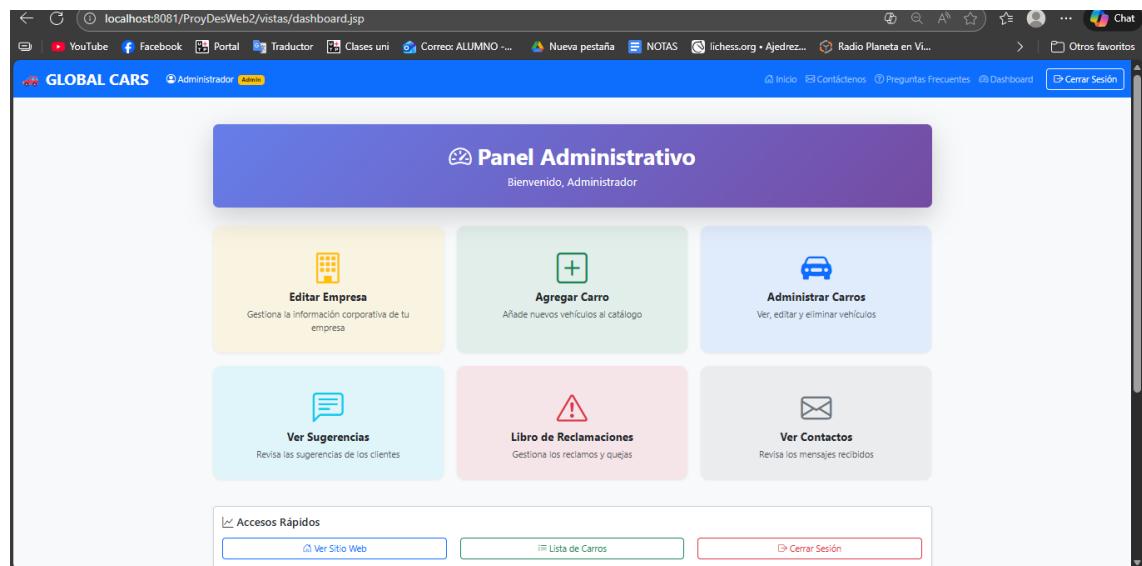
\$ Calcular cuota mensual

Sistema Web Plataforma integral para gestión de procesos empresariales con tecnología moderna y segura. 	Enlaces Rápidos Inicio Preguntas Frecuentes Contáctenos Enviar Sugerencias Libro de Reclamaciones	Contacto Dirección: Av. Universitaria 1801 San Miguel, Lima - Perú Teléfono: (01) 626-2000 Email: info@empresa.com	Horario de Atención Lunes a Viernes: 8:00 AM - 6:00 PM Sábados: 9:00 AM - 1:00 PM Domingos: Cerrado
---	---	--	---

© 2025 Sistema Web. Todos los derechos reservados.
Desarrollado con tecnología Java - JSP | Versión 1.0

[Política de Privacidad](#) | [Términos y Condiciones](#) | [Soporte](#)

- **Dashboard Administrador:**



- **Formulario para libro de reclamaciones:**

The screenshot shows the 'Libro de Reclamaciones' (Complaint Book) form. The title 'Libro de Reclamaciones' is at the top, followed by a note: 'Este formulario está destinado a presentar reclamos y quejas conforme a la normativa vigente.' The form contains several input fields: 'Nombre completo' (Name), 'Correo electrónico' (Email), 'Teléfono' (Phone), 'Dirección' (Address), 'DNI / RUC' (ID/RC), 'Tipo de solicitud' (Type of request) with a dropdown menu, 'Producto o servicio involucrado' (Product or service involved), 'Descripción' (Description), and 'Pedido del consumidor' (Consumer request). A red button at the bottom right is labeled 'Registrar Reclamo' (Register Complaint).

- **Formulario de sugerencias:**

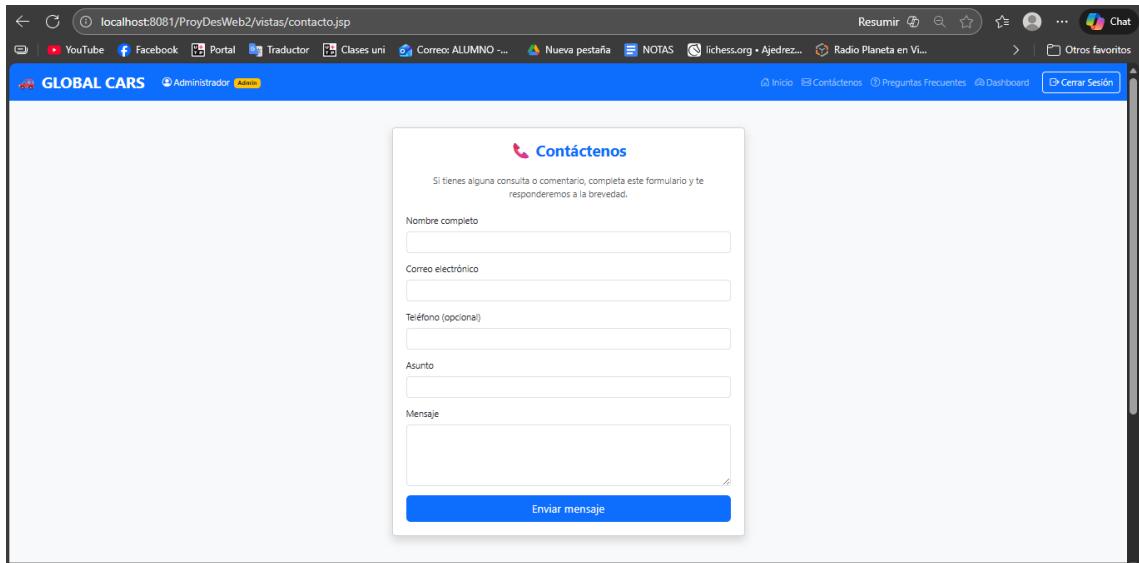
The screenshot shows a web page with a blue header bar. The header contains the text 'GLOBAL CARS' and 'Administrador'. On the right side of the header are links for 'Inicio', 'Contáctenos', 'Preguntas Frecuentes', 'Dashboard', and a 'Cerrar Sesión' button. Below the header is a white form box with a title 'Envíanos tu sugerencia'. Inside the form, there is a message: 'Queremos mejorar nuestros servicios. Tu opinión es muy importante para nosotros.' followed by four input fields: 'Nombre completo' (with placeholder 'Ej: Juan Pérez'), 'Correo electrónico' (with placeholder 'correo@ejemplo.com'), 'Asunto' (with placeholder 'Ej: Mejorar la atención'), and a large 'Mensaje' text area with placeholder 'Escribe aquí tu sugerencia...'. At the bottom of the form is a blue button labeled 'Enviar sugerencia'.

- **Preguntas frecuentes (FAQ)**

The screenshot shows a web page with a blue header bar. The header contains the text 'GLOBAL CARS' and 'Administrador'. On the right side of the header are links for 'Inicio', 'Contáctenos', 'Preguntas Frecuentes', 'Dashboard', and a 'Cerrar Sesión' button. Below the header is a section titled 'Preguntas Frecuentes' containing five expandable questions:

- ¿Cómo puedo cotizar un auto? (With expanded content: 'Para cotizar un auto, solo necesitas iniciar sesión, seleccionar el modelo que te interesa y hacer clic en "Cotizar". Nuestro equipo se pondrá en contacto contigo para brindarte más detalles.')
- ¿Qué documentos necesito para comprar un auto?
- ¿Puedo financiar mi auto?
- ¿Puedo probar el auto antes de comprarlo?
- ¿Qué métodos de pago aceptan?

- **Contáctenos / Ubícanos: formulario**



Capítulo 3:

3.1 Implementación de patrón MVC

```

<!-- ACTUALIZAR -->
<h2>Modificar Carro</h2>
<form id="formActualizar" enctype="multipart/form-data">
    <input name="id" type="number" placeholder="ID" required>
    <input name="nombre" placeholder="Nuevo nombre" required>
    <input name="descripcion" placeholder="Nueva descripción">
    <input name="precio" type="number" placeholder="Nuevo precio" required>
    <input type="file" name="archivoImagen" accept="image/*">
    <button type="submit">Modificar</button>
</form>
<div id="resultadoActualizar"></div>

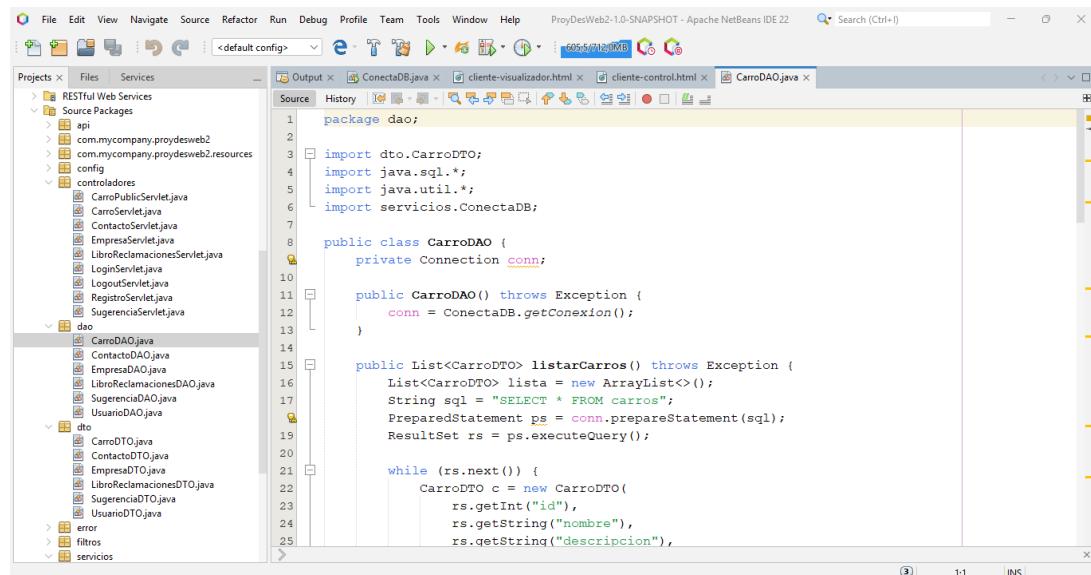
<script>
    const baseUrl = "http://localhost:8081/ProyDesWeb2/carro",
        // CREAR
        document.getElementById("formCrear").addEventListener("submit", e => {
            e.preventDefault();
            const formData = new FormData(e.target);
            formData.append("accion", "insertar");

            fetch(baseUrl, {
                method: "POST",
                body: formData,
            })
        })
</script>

```

3.2 Funcionalidad del sistema

• CarroDAO



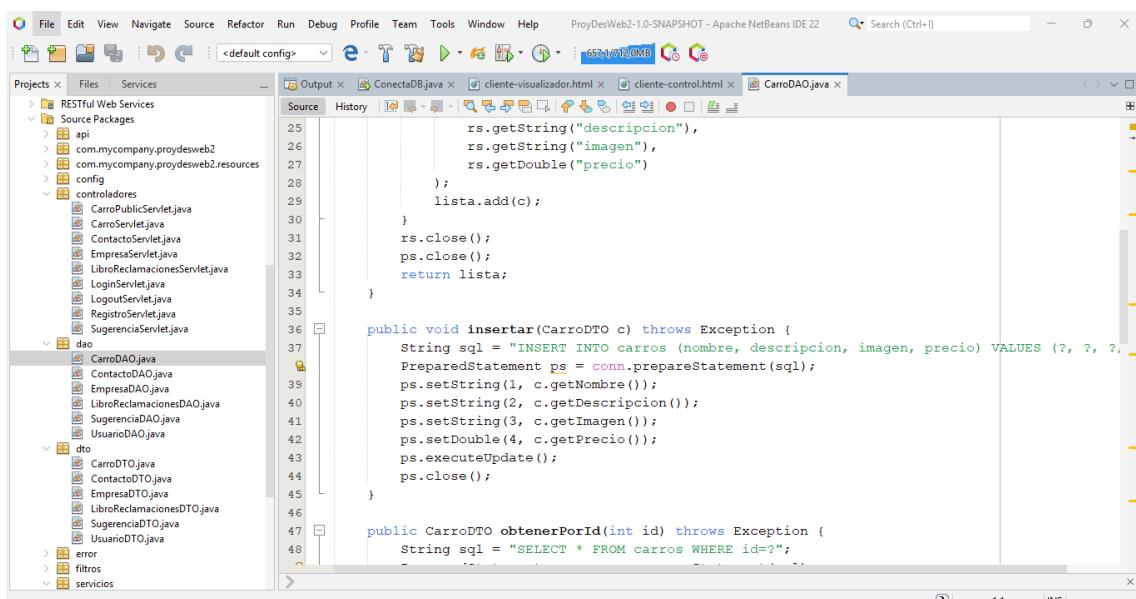
The screenshot shows the Apache NetBeans IDE interface with the following details:

- Projects Tab:** Shows the project structure under "RESTful Web Services". The "dao" package contains several Java files: CarroDAO.java, ContactoDAO.java, EmpresaDAO.java, LibroReclamacionesDAO.java, SugerenciaDAO.java, and UsuarioDAO.java.
- Output Tab:** Shows the current file being edited: CarroDAO.java.
- Source Tab:** Displays the Java code for the CarroDAO class. The code implements a DAO interface for managing CarroDTO objects. It includes methods for listing cars, inserting a car, and getting a car by ID. The code uses JDBC to interact with a database.

```

1 package dao;
2
3 import dto.CarroDTO;
4 import java.sql.*;
5 import java.util.*;
6 import servicios.ConectaDB;
7
8 public class CarroDAO {
9     private Connection conn;
10
11     public CarroDAO() throws Exception {
12         conn = ConectaDB.getConexion();
13     }
14
15     public List<CarroDTO> listarCarros() throws Exception {
16         List<CarroDTO> lista = new ArrayList<>();
17         String sql = "SELECT * FROM carros";
18         PreparedStatement ps = conn.prepareStatement(sql);
19         ResultSet rs = ps.executeQuery();
20
21         while (rs.next()) {
22             CarroDTO c = new CarroDTO(
23                 rs.getInt("id"),
24                 rs.getString("nombre"),
25                 rs.getString("descripcion"),
26                 rs.getString("imagen"),
27                 rs.getDouble("precio")
28             );
29             lista.add(c);
30         }
31         rs.close();
32         ps.close();
33         return lista;
34     }
35
36     public void insertar(CarroDTO c) throws Exception {
37         String sql = "INSERT INTO carros (nombre, descripcion, imagen, precio) VALUES (?, ?, ?, ?)";
38         PreparedStatement ps = conn.prepareStatement(sql);
39         ps.setString(1, c.getNombre());
40         ps.setString(2, c.getDescripcion());
41         ps.setString(3, c.getImagen());
42         ps.setDouble(4, c.getPrecio());
43         ps.executeUpdate();
44         ps.close();
45     }
46
47     public CarroDTO obtenerPorId(int id) throws Exception {
48         String sql = "SELECT * FROM carros WHERE id=?";
49
50         ...
51     }
52 }

```



This screenshot continues the view of the CarroDAO.java code from the previous screenshot, showing the remaining methods and their implementations:

- Projects Tab:** Same project structure as the first screenshot.
- Output Tab:** Still showing CarroDAO.java.
- Source Tab:** Continues the implementation of the CarroDAO class. It includes a constructor that initializes the connection, a method to list cars, a method to insert a car, and a method to get a car by its ID. The code uses JDBC to interact with the database.

```

26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22 Search (Ctrl+)

Projects Files Services

RESTful Web Services
Source Packages
api
com.mycompany.proydesweb2
com.mycompany.proydesweb2.resources
config
controladores
CarroPublicServlet.java
CarroServlet.java
ContactoServlet.java
EmpresaServlet.java
LibroReclamacionesServlet.java
LoginServlet.java
LogoutServlet.java
RegistroServlet.java
SugerenciaServlet.java
dao
CarroDAO.java
ContactoDAO.java
EmpresaDAO.java
LibroReclamacionesDAO.java
SugerenciaDAO.java
UsuarioDAO.java
dto
CarroDTO.java
ContactoDTO.java
EmpresaDTO.java
LibroReclamacionesDTO.java
SugerenciaDTO.java
UsuarioDTO.java
error
filtros
servicios

Output ConnectDB.java cliente-visualizador.html cliente-control.html CarroDAO.java

```
public CarroDTO obtenerPorId(int id) throws Exception {
    String sql = "SELECT * FROM carros WHERE id=?";
    PreparedStatement ps = conn.prepareStatement(sql);
    ps.setInt(1, id);
    ResultSet rs = ps.executeQuery();
    CarroDTO c = null;
    if (rs.next()) {
        c = new CarroDTO(
            rs.getInt("id"),
            rs.getString("nombre"),
            rs.getString("descripcion"),
            rs.getString("imagen"),
            rs.getDouble("precio")
        );
    }
    rs.close();
    ps.close();
    return c;
}

public void actualizar(CarroDTO c) throws Exception {
    String sql = "UPDATE carros SET nombre=?, descripcion=?, imagen=?, precio=? WHERE id=?";
    PreparedStatement ps = conn.prepareStatement(sql);
    ps.setString(1, c.getNombre());
    ps.setString(2, c.getDescripcion());
    ps.setString(3, c.getImagen());
    ps.setDouble(4, c.getPrecio());
    ps.setInt(5, c.getId());
    ps.executeUpdate();
    ps.close();
}
```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22 Search (Ctrl+)

Projects Files Services

RESTful Web Services
Source Packages
api
com.mycompany.proydesweb2
com.mycompany.proydesweb2.resources
config
controladores
CarroPublicServlet.java
CarroServlet.java
ContactoServlet.java
EmpresaServlet.java
LibroReclamacionesServlet.java
LoginServlet.java
LogoutServlet.java
RegistroServlet.java
SugerenciaServlet.java
dao
CarroDAO.java
ContactoDAO.java
EmpresaDAO.java
LibroReclamacionesDAO.java
SugerenciaDAO.java
UsuarioDAO.java
dto
CarroDTO.java
ContactoDTO.java
EmpresaDTO.java
LibroReclamacionesDTO.java
SugerenciaDTO.java
UsuarioDTO.java
error
filtros
servicios

Output ConnectDB.java cliente-visualizador.html cliente-control.html CarroDAO.java

```
public void actualizar(CarroDTO c) throws Exception {
    String sql = "UPDATE carros SET nombre=?, descripcion=?, imagen=?, precio=? WHERE id=?";
    PreparedStatement ps = conn.prepareStatement(sql);
    ps.setString(1, c.getNombre());
    ps.setString(2, c.getDescripcion());
    ps.setString(3, c.getImagen());
    ps.setDouble(4, c.getPrecio());
    ps.setInt(5, c.getId());
    ps.executeUpdate();
    ps.close();
}

public void eliminar(int id) throws Exception {
    String sql = "DELETE FROM carros WHERE id=?";
    PreparedStatement ps = conn.prepareStatement(sql);
    ps.setInt(1, id);
    ps.executeUpdate();
    ps.close();
}
```

- EmpresaDAO

The screenshot shows the Apache NetBeans IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help.
- Toolbar:** Includes icons for file operations like Open, Save, Find, and Run.
- Project Explorer:** Shows the project structure under "Projects".
 - RESTful Web Services
 - Source Packages
 - api
 - com.mycompany.proydesweb2
 - com.mycompany.proydesweb2.resources
 - config
 - controladores
 - CarroPublicServlet.java
 - CarroServlet.java
 - ContactoServlet.java
 - EmpresaServlet.java
 - LibroReclamacionesServlet.java
 - LoginServlet.java
 - LogoutServlet.java
 - RegistroServlet.java
 - SugerenciaServlet.java
 - dao
 - CarroDAO.java
 - ContactoDAO.java
 - EmpresaDAO.java
 - LibroReclamacionesDAO.java
 - SugerenciaDAO.java
 - UsuarioDAO.java
 - dto
 - CarroDTO.java
 - ContactoDTO.java
 - EmpresaDTO.java
 - LibroReclamacionesDTO.java
 - SugerenciaDTO.java
 - UsuarioDTO.java
 - error
 - filtros
 - servicios
- Output View:** Shows "5551/17800MB" and a progress bar.
- Code Editor:** Displays the content of "EmpresaDAO.java".

```
package dao;

import dto.EmpresaDTO;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import servicios.ConectaDB;

public class EmpresaDAO {

    // Obtener la empresa registrada (asumiendo que solo hay una)
    public EmpresaTO obtenerEmpresa() throws SQLException {
        EmpresaTO empresa = null;
        String sql = "SELECT * FROM empresa LIMIT 1";

        try (Connection cn = ConectaDB.getConexion();
             PreparedStatement ps = cn.prepareStatement(sql);
             ResultSet rs = ps.executeQuery()) {

            if (rs.next()) {
                empresa = new EmpresaDTO();
                empresa.setId(rs.getInt("id"));
                empresa.setRuc(rs.getString("ruc"));
                empresa.setRazonSocial(rs.getString("razon_social"));
            }
        }
    }
}
```

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ProjDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22 Search (Ctrl+I) <default config> 6661/730.0MB Projects x Files Services RESTful Web Services Source Packages api com.mycompany.proydesweb2 com.mycompany.proydesweb2.resources config controladores CarroPublicServlet.java CarroServlet.java ContactoServlet.java EmpresaServlet.java LibroReclamacionesServlet.java LoginServlet.java LogoutServlet.java RegistroServlet.java SugerenciaServlet.java dao CarroDAO.java ContactoDAO.java EmpresaDAO.java LibroReclamacionesDAO.java SugerenciaDAO.java UsuarioDAO.java dto CarroDTO.java ContactoDTO.java EmpresaDTO.java LibroReclamacionesDTO.java SugerenciaDTO.java UsuarioDTO.java error filtros servicios <default config> Output x ConectaDB.java x cliente-visualizador.html x cliente-control.html x CarroDAO.java x EmpresaDAO.java x Source History &nbsp; 25 empresa.setRazonSocial(rs.getString("razon_social")); 26 empresa.setNombreComercial(rs.getString("nombre_comercial")); 27 empresa.setDireccion(rs.getString("direccion")); 28 empresa.setTelefono(rs.getString("telefono")); 29 empresa.setCorreo(rs.getString("correo")); 30 31 } 32 } 33 return empresa; 34 } 35 36 // Insertar nueva empresa 37 public boolean insertar(EmpresaDTO empresa) throws SQLException { 38     String sql = "INSERT INTO empresa (ruc, razon_social, nombre_comercial, direccion, telefono) 39     + \"VALUES (?, ?, ?, ?, ?, ?)\"; 40 41     try (Connection cn = ConectaDB.getConexion(); 42          PreparedStatement ps = cn.prepareStatement(sql)) { 43 44         ps.setString(1, empresa.getRuc()); 45         ps.setString(2, empresa.getRazonSocial()); 46         ps.setString(3, empresa.getNombreComercial()); 47         ps.setString(4, empresa.getDireccion()); 48         ps.setString(5, empresa.getTelefono());
```

Two screenshots of the Apache NetBeans IDE 22 interface showing Java code for DAO classes.

Screenshot 1: The code for `EmpresaDAO.java`. It contains methods for updating an existing company and checking if a company exists.

```

46     ps.setString(3, empresa.getNombreComercial());
47     ps.setString(4, empresa.getDireccion());
48     ps.setString(5, empresa.getTelefono());
49     ps.setString(6, empresa.getCorreo());

50
51     return ps.executeUpdate() > 0;
52 }

53 }

54 // Actualizar empresa existente
55 public boolean actualizar(EmpresaDTO empresa) throws SQLException {
56     String sql = "UPDATE empresa SET ruc=?, razon_social=?, nombre_comercial=?
57         + WHERE id=?";

58     try (Connection cn = ConectaDB.getConexion();
59          PreparedStatement ps = cn.prepareStatement(sql)) {

60         ps.setString(1, empresa.getRuc());
61         ps.setString(2, empresa.getRazonSocial());
62         ps.setString(3, empresa.getNombreComercial());
63         ps.setString(4, empresa.getDireccion());
64         ps.setString(5, empresa.getTelefono());
65         ps.setString(6, empresa.getCorreo());
66     }
67 }
68
69 }
```

Screenshot 2: The code for `CarroDAO.java`. It contains methods for updating a car and checking if a car exists.

```

65     ps.setString(2, empresa.getRazonSocial());
66     ps.setString(3, empresa.getNombreComercial());
67     ps.setString(4, empresa.getDireccion());
68     ps.setString(5, empresa.getTelefono());
69     ps.setString(6, empresa.getCorreo());
70     ps.setInt(7, empresa.getId());

71     return ps.executeUpdate() > 0;
72 }

73 }

74 }

75 // Verificar si existe empresa registrada
76 public boolean existeEmpresa() throws SQLException {
77     String sql = "SELECT COUNT(*) FROM empresa";
78     try (Connection cn = ConectaDB.getConexion();
79          PreparedStatement ps = cn.prepareStatement(sql);
80          ResultSet rs = ps.executeQuery()) {

81         if (rs.next()) {
82             return rs.getInt(1) > 0;
83         }
84     }
85     return false;
86 }
87 }
```

3.3 Diseño de la base de datos con al menos 6 tablas relacionadas

- **Tabla carros**

Screenshot of the MySQL Workbench interface showing the database schema and a query result grid.

Navigator: Shows the database structure with schemas `db_boutique`, `sakila`, and `simcardb`. The `simcardb` schema contains tables `carros`, `contacto`, `empresa`, `libro_reclamaciones`, `sugerencias`, and `usuarios`.

Query 1: A query to select all columns from the `carros` table.

```
1 • SELECT * FROM simcardb.carros;
```

Result Grid: Displays the results of the query.

ID	nombre	descripcion	imagen	precio
1	test	test	toyota.jpg	5000.00
2	Mazda CX-3	Hermoso diseño, llamativo	MAZDA-CX-3.jpg	68000.00
3	Toyota	Carro con buen modelo	toyota1.jpg	56920.00
4	carro	asdsda	Captura de pantalla 2025-08-28 191559.png	123.00
5	DFSK Glory 560	Carro amplio con 3 filas de asiento	glory.jpeg	60000.00
6	Chevrolet Spark	Carro confortable motor 1.0	spark.webp	50000.00
*	NULL	NULL	NULL	NULL

● Tabla Contacto

The screenshot shows the MySQL Workbench interface. The left pane displays the Navigator with the 'simcardb' schema selected. The right pane shows the 'Query 1' tab with the query `SELECT * FROM simcardb.contacto;`. The results grid shows one row of data:

id	nombre	correo	telefono	asunto	mensaje	fecha
1	sdadas	saasd@fassa.com	dsadas	saddsa	adssda	2025-10-13 14:30:10

● Tabla Empresa

The screenshot shows the MySQL Workbench interface. The left pane displays the Navigator with the 'simcardb' schema selected. The right pane shows the 'Query 1' tab with the query `SELECT * FROM simcardb.empresa;`. The results grid shows one row of data:

id	ruc	razon_social	nombre_comercial	direccion	telefono	correo
1	555555	huhuh	lq	kjjkp	jiojiojo	SADDASAS@GMAIL.omcoads

● Tabla Libro_reclamaciones

The screenshot shows the MySQL Workbench interface. The left pane displays the Navigator with the 'simcardb' schema selected. The right pane shows the 'Query 1' tab with the query `SELECT * FROM simcardb.libro_reclamaciones;`. The results grid shows one row of data:

id	nombre	correo	telefono	direccion	documento	tipo	producto	descripcion	pedido	fecha
1	Toyota	admin@cotizador.com	jiojiojo	tu casa	46522581	Reclamo	carro	todo	quiero mejorar	2025-10-13 14:49:44

● Tabla Sugerencias

Navigator: Schemas

Query 1: SELECT * FROM simcardb.sugerencias;

	id	nombre	correo	asunto	mensaje	fecha
1	1	ASDDASAD	SADDASAS@GMAIL.COM.CO	sadssda	dsadsadasd	2025-10-13 14:29:12
*	NULL	NULL	NULL	NULL	NULL	NULL

● Tabla Usuarios

Navigator: Schemas

Query 1: SELECT * FROM simcardb.usuarios;

	id	correo	contraseña	nombre	rol
1	1	admin@admin.com	admin	Administrador	Admin
*	NULL	NULL	NULL	NULL	NULL

● Conexión ConectaDB

File Edt View Navigate Source Refactor Run Debug Profile Team Tools Window Help

ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 21

Projects x Files Services

Ejemplo1 ProyDesWeb2-1.0-SNAPSHOT [master]

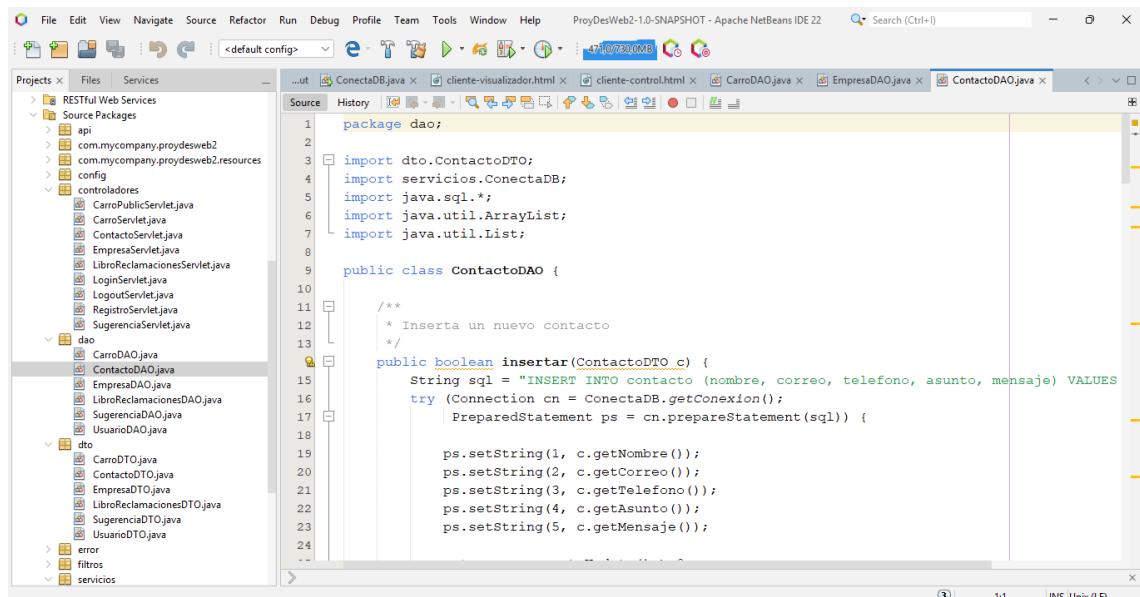
Start Page x ConectaDB.java x

```

1  /*
2   * Click nbfs://nhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5  package servicios;
6
7  import java.sql.Connection;
8  import java.sql.DriverManager;
9  import java.sql.SQLException;
10 import java.util.logging.Level;
11 import java.util.logging.Logger;
12
13 /**
14  * @author USUARIO
15  */
16
17 public class ConectaDB {
18     public static Connection getConexion(){
19         Connection cnx = null;
20
21         String url = "jdbc:mysql://localhost:3306/simcardb?useTimezone=true&" +
22                     "&serverTimezone=UTC&autoReconnect=true";
23
24         String user = "root";
25         String password = "root";
26         String Driver = "com.mysql.cj.jdbc.Driver";
27
28         try {
29             Class.forName(Driver);
30             cnx = DriverManager.getConnection(url, user, password);
31         } catch (ClassNotFoundException | SQLException ex) {
32             Logger.getLogger(ConectaDB.class.getName()).log(Level.SEVERE, null, ex);
33         }
34
35         return cnx;
36     }
37 }
38

```

• ContactoDAO



The screenshot shows the Apache NetBeans IDE interface with the following details:

- Project Structure:** RESTful Web Services > Source Packages > com.mycompany.proydesweb2 > config > controladores > ContactoDAO.java.
- Code Editor:** The main window displays the `ContactoDAO.java` file. The code implements a DAO interface for managing contacts. It includes methods for inserting a new contact and listing all contacts.
- Code Snippet (Insertar):**

```

1 package dao;
2
3 import dto.ContactoDTO;
4 import servicios.ConectaDB;
5 import java.sql.*;
6 import java.util.ArrayList;
7 import java.util.List;
8
9 public class ContactoDAO {
10
11     /**
12      * Inserta un nuevo contacto
13      */
14     public boolean insertar(ContactoDTO c) {
15         String sql = "INSERT INTO contacto (nombre, correo, telefono, asunto, mensaje) VALUES";
16         try (Connection cn = ConectaDB.getConexion();
17              PreparedStatement ps = cn.prepareStatement(sql)) {
18
19             ps.setString(1, c.getNombre());
20             ps.setString(2, c.getCorreo());
21             ps.setString(3, c.getTelefono());
22             ps.setString(4, c.getAsunto());
23             ps.setString(5, c.getMensaje());
24
25             return ps.executeUpdate() > 0;
26         } catch (Exception e) {
27             e.printStackTrace();
28             return false;
29         }
30     }
31
32     /**
33      * Lista todos los contactos
34      */
35     public List<ContactoDTO> listarTodos() throws Exception {
36         Connection cnx = null;
37         PreparedStatement ps = null;
38         ResultSet rs = null;
39         List<ContactoDTO> lista = new ArrayList<>();
40
41         try {
42             cnx = ConectaDB.getConexion();
43             String sql = "SELECT * FROM contacto ORDER BY fecha DESC";
44             ps = cnx.prepareStatement(sql);
45
46             ...
        
```

ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22

```

    cnx = ConectaDB.getConexion();
    String sql = "SELECT * FROM contacto ORDER BY fecha DESC";
    ps = cnx.prepareStatement(sql);
    rs = ps.executeQuery();

    while (rs.next()) {
        ContactoDTO contacto = new ContactoDTO();
        contacto.setId(rs.getInt("id"));
        contacto.setNombre(rs.getString("nombre"));
        contacto.setCorreo(rs.getString("correo"));
        contacto.setTelefono(rs.getString("telefono"));
        contacto.setAsunto(rs.getString("asunto"));
        contacto.setMensaje(rs.getString("mensaje"));
        contacto.setFecha(rs.getTimestamp("fecha"));
        lista.add(contacto);
    }
} finally {
    if (rs != null) rs.close();
    if (ps != null) ps.close();
    if (cnx != null) cnx.close();
}

return lista;
}

```

ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22

```

    return lista;

    /**
     * Busca un contacto por ID
     */
    public ContactoDTO buscarPorId(int id) throws Exception {
        Connection cnx = null;
        PreparedStatement ps = null;
        ResultSet rs = null;
        ContactoDTO contacto = null;

        try {
            cnx = ConectaDB.getConexion();
            String sql = "SELECT * FROM contacto WHERE id = ?";
            ps = cnx.prepareStatement(sql);
            ps.setInt(1, id);
            rs = ps.executeQuery();

            if (rs.next()) {
                contacto = new ContactoDTO();
                contacto.setId(rs.getInt("id"));
                contacto.setNombre(rs.getString("nombre"));
            }
        } finally {
            if (rs != null) rs.close();
            if (ps != null) ps.close();
            if (cnx != null) cnx.close();
        }

        return contacto;
    }

```

ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22

```

    contacto = new ContactoDTO();
    contacto.setId(rs.getInt("id"));
    contacto.setNombre(rs.getString("nombre"));
    contacto.setCorreo(rs.getString("correo"));
    contacto.setTelefono(rs.getString("telefono"));
    contacto.setAsunto(rs.getString("asunto"));
    contacto.setMensaje(rs.getString("mensaje"));
    contacto.setFecha(rs.getTimestamp("fecha"));

} finally {
    if (rs != null) rs.close();
    if (ps != null) ps.close();
    if (cnx != null) cnx.close();
}

return contacto;
}

    /**
     * Elimina un contacto
     */
    public boolean eliminar(int id) throws Exception {
        Connection cnx = null;
        PreparedStatement ps = null;

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22 Search (Ctrl+F) <default config> 448/730.0MB

Projects > RESTful Web Services > Source Packages > com.mycompany.proydesweb2 > config > controladores > CarroPublicServlet.java, CarroServlet.java, ContactoServlet.java, EmpresaServlet.java, LibroReclamacionesServlet.java, LoginServlet.java, LogoutServlet.java, RegistroServlet.java, SugerenciaServlet.java > dao > CarroDAO.java, ContactoDAO.java, EmpresaDAO.java, LibroReclamacionesDAO.java, SugerenciaDAO.java, UsuarioDAO.java > dto > CarroDTO.java, ContactoDTO.java, EmpresaDTO.java, LibroReclamacionesDTO.java, SugerenciaDTO.java, UsuarioDTO.java > error > filtros > servicios

Source History ...ut ConectaDB.java cliente-visualizador.html cliente-control.html CarroDAO.java EmpresaDAO.java ContactoDAO.java

```
108     PreparedStatement ps = null;
109
110     try {
111         cnx = ConectaDB.getConexion();
112         String sql = "DELETE FROM contacto WHERE id = ?";
113         ps = cnx.prepareStatement(sql);
114         ps.setInt(1, id);
115
116         return ps.executeUpdate() > 0;
117     } finally {
118         if (ps != null) ps.close();
119         if (cnx != null) cnx.close();
120     }
121
122 /**
123 * Cuenta el total de contactos
124 */
125 public int contarTodos() throws Exception {
126     Connection cnx = null;
127     PreparedStatement ps = null;
128     ResultSet rs = null;
129
130     try {
131         cnx = ConectaDB.getConexion();
132         String sql = "SELECT COUNT(*) as total FROM contacto";
133         ps = cnx.prepareStatement(sql);
134         rs = ps.executeQuery();
135
136         if (rs.next()) {
137             return rs.getInt("total");
138         }
139         return 0;
140     } finally {
141         if (rs != null) rs.close();
142         if (ps != null) ps.close();
143         if (cnx != null) cnx.close();
144     }
145
146 }
147 }
```

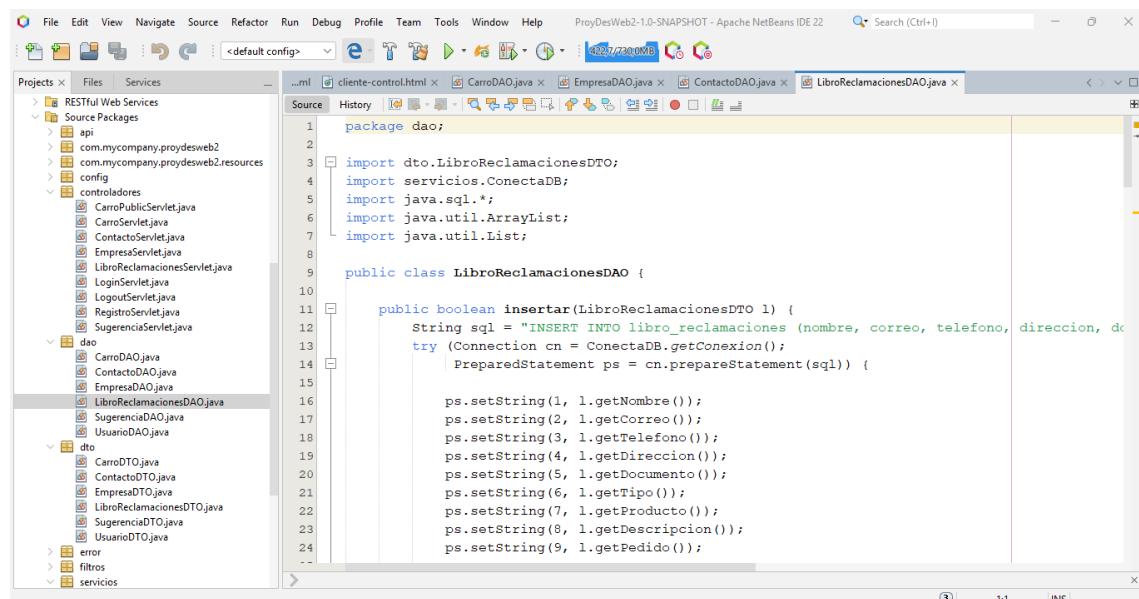
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22 Search (Ctrl+F) <default config> 492/730.0MB

Projects > RESTful Web Services > Source Packages > com.mycompany.proydesweb2 > config > controladores > CarroPublicServlet.java, CarroServlet.java, ContactoServlet.java, EmpresaServlet.java, LibroReclamacionesServlet.java, LoginServlet.java, LogoutServlet.java, RegistroServlet.java, SugerenciaServlet.java > dao > CarroDAO.java, ContactoDAO.java, EmpresaDAO.java, LibroReclamacionesDAO.java, SugerenciaDAO.java, UsuarioDAO.java > dto > CarroDTO.java, ContactoDTO.java, EmpresaDTO.java, LibroReclamacionesDTO.java, SugerenciaDTO.java, UsuarioDTO.java > error > filtros > servicios

Source History ...ut ConectaDB.java cliente-visualizador.html cliente-control.html CarroDAO.java EmpresaDAO.java ContactoDAO.java

```
129     ResultSet rs = null;
130
131     try {
132         cnx = ConectaDB.getConexion();
133         String sql = "SELECT COUNT(*) as total FROM contacto";
134         ps = cnx.prepareStatement(sql);
135         rs = ps.executeQuery();
136
137         if (rs.next()) {
138             return rs.getInt("total");
139         }
140         return 0;
141     } finally {
142         if (rs != null) rs.close();
143         if (ps != null) ps.close();
144         if (cnx != null) cnx.close();
145     }
146
147 }
```

- **LibroReclamacionesDAO**



The screenshot shows the Apache NetBeans IDE 22 interface. The title bar reads "ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and a search bar "Search (Ctrl+F)". The Projects tab is selected, showing a tree view of the project structure:

- RESTful Web Services
- Source Packages
 - api
 - com.mycompany.proydesweb2
 - com.mycompany.proydesweb2.resources
 - config
 - controladores
 - CarroPublicServlet.java
 - CarroServlet.java
 - ContactoServlet.java
 - EmpresServlet.java
 - LibroReclamacionesServlet.java
 - LoginServlet.java
 - LogoutServlet.java
 - RegistroServlet.java
 - SugerenciasServlet.java
 - dao
 - CarroDAO.java
 - ContactoDAO.java
 - EmpresDAO.java
 - LibroReclamacionesDAO.java
 - SugerenciaDAO.java
 - UsuarioDAO.java
 - dto
 - CarroDTO.java
 - ContactoDTO.java
 - EmpresDTO.java
 - LibroReclamacionesDTO.java
 - SugerenciaDTO.java
 - UsuarioDTO.java
 - error
 - filtros
 - servicios

The main editor area displays the code for `LibroReclamacionesDAO.java`:

```
package dao;  
import dto.LibroReclamacionesDTO;  
import servicios.ConectaDB;  
import java.sql.*;  
import java.util.ArrayList;  
import java.util.List;  
  
public class LibroReclamacionesDAO {  
  
    public boolean insertar(LibroReclamacionesDTO l) {  
        String sql = "INSERT INTO libro_reclamaciones (nombre, correo, telefono, direccion, dc  
        try (Connection cn = ConectaDB.getConexion()) {  
            PreparedStatement ps = cn.prepareStatement(sql) {  
  
                ps.setString(1, l.getNombre());  
                ps.setString(2, l.getCorreo());  
                ps.setString(3, l.getTelefono());  
                ps.setString(4, l.getDireccion());  
                ps.setString(5, l.getDocumento());  
                ps.setString(6, l.getTipo());  
                ps.setString(7, l.getProducto());  
                ps.setString(8, l.getDescripcion());  
                ps.setString(9, l.getPedido());  
            }  
        } catch (SQLException ex) {  
            ex.printStackTrace();  
        }  
        return true;  
    }  
}
```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22 Search (Ctrl+I) <default config> 533.7/730.0MB

Projects x Files Services

```

    > RESTful Web Services
    > Source Packages
    > api
    > com.mycompany.proydesweb2
    > com.mycompany.proydesweb2.resources
    > config
    > controladores
        CarroPublicServlet.java
        CarroServlet.java
        ContactoServlet.java
        EmpresaServlet.java
        LibroReclamacionesServlet.java
        LoginServlet.java
        LogoutServlet.java
        RegistroServlet.java
        SugerenciaServlet.java
    > dao
        CarroDAO.java
        ContactoDAO.java
        EmpresaDAO.java
        LibroReclamacionesDAO.java
        SugerenciaDAO.java
        UsuarioDAO.java
    > dto
        CarroDTO.java
        ContactoDTO.java
        EmpresaDTO.java
        LibroReclamacionesDTO.java
        SugerenciaDTO.java
        UsuarioDTO.java
    > error
    > filtros
    > servicios

```

...ml cliente-control.html x CarroDAO.java x EmpresaDAO.java x ContactoDAO.java x LibroReclamacionesDAO.java

Source History

```

23     ps.setString(8, l.getDescripcion());
24     ps.setString(9, l.getPedido());
25
26     return ps.executeUpdate() > 0;
27 }
28
29 } catch (Exception e) {
30     e.printStackTrace();
31     return false;
32 }
33
34 public List<LibroReclamacionesDTO> listarTodos() throws Exception {
35     Connection cnx = null;
36     PreparedStatement ps = null;
37     ResultSet rs = null;
38     List<LibroReclamacionesDTO> lista = new ArrayList<>();
39
40     try {
41         cnx = ConectaDB.getConexion();
42         String sql = "SELECT * FROM libro_reclamaciones ORDER BY fecha DESC";
43         ps = cnx.prepareStatement(sql);
44         rs = ps.executeQuery();
45
46         while (rs.next()) {
47             LibroReclamacionesDTO reclamo = new LibroReclamacionesDTO();
48             reclamo.setId(rs.getInt("id"));
49             reclamo.setNombre(rs.getString("nombre"));
50             reclamo.setCorreo(rs.getString("correo"));
51             reclamo.setTelefonos(rs.getString("telefono"));
52             reclamo.setDireccion(rs.getString("direccion"));
53
54             reclamo.setDireccion(rs.getString("direccion"));
55             reclamo.setDocumento(rs.getString("documento"));
56             reclamo.setTipo(rs.getString("tipo"));
57             reclamo.setProducto(rs.getString("producto"));
58             reclamo.setDescripcion(rs.getString("descripcion"));
59             reclamo.setPedido(rs.getString("pedido"));
60             reclamo.setFecha(rs.getTimestamp("fecha"));
61             lista.add(reclamo);
62         }
63     } finally {
64         if (rs != null) rs.close();
65         if (ps != null) ps.close();
66         if (cnx != null) cnx.close();
67     }
68
69     return lista;
70 }
71
72 public LibroReclamacionesDTO buscarPorId(int id) throws Exception {
73     Connection cnx = null;
74     PreparedStatement ps = null;
75     ResultSet rs = null;
76     LibroReclamacionesDTO reclamo = null;
77
78     try {
79         cnx = ConectaDB.getConexion();
80         String sql = "SELECT * FROM libro_reclamaciones WHERE id = ?";
81         ps = cnx.prepareStatement(sql);
82         ps.setInt(1, id);
83         rs = ps.executeQuery();

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22 Search (Ctrl+I) <default config> 626.0/730.0MB

Projects x Files Services

```

    > RESTful Web Services
    > Source Packages
    > api
    > com.mycompany.proydesweb2
    > com.mycompany.proydesweb2.resources
    > config
    > controladores
        CarroPublicServlet.java
        CarroServlet.java
        ContactoServlet.java
        EmpresaServlet.java
        LibroReclamacionesServlet.java
        LoginServlet.java
        LogoutServlet.java
        RegistroServlet.java
        SugerenciaServlet.java
    > dao
        CarroDAO.java
        ContactoDAO.java
        EmpresaDAO.java
        LibroReclamacionesDAO.java
        SugerenciaDAO.java
        UsuarioDAO.java
    > dto
        CarroDTO.java
        ContactoDTO.java
        EmpresaDTO.java
        LibroReclamacionesDTO.java
        SugerenciaDTO.java
        UsuarioDTO.java
    > error
    > filtros
    > servicios

```

...ml cliente-control.html x CarroDAO.java x EmpresaDAO.java x ContactoDAO.java x LibroReclamacionesDAO.java

Source History

```

52     reclamo.setDireccion(rs.getString("direccion"));
53     reclamo.setDocumento(rs.getString("documento"));
54     reclamo.setTipo(rs.getString("tipo"));
55     reclamo.setProducto(rs.getString("producto"));
56     reclamo.setDescripcion(rs.getString("descripcion"));
57     reclamo.setPedido(rs.getString("pedido"));
58     reclamo.setFecha(rs.getTimestamp("fecha"));
59     lista.add(reclamo);
60 }
61 } finally {
62     if (rs != null) rs.close();
63     if (ps != null) ps.close();
64     if (cnx != null) cnx.close();
65 }
66
67     return lista;
68 }
69
70 public LibroReclamacionesDTO buscarPorId(int id) throws Exception {
71     Connection cnx = null;
72     PreparedStatement ps = null;
73     ResultSet rs = null;
74     LibroReclamacionesDTO reclamo = null;
75
76     try {
77         cnx = ConectaDB.getConexion();
78         String sql = "SELECT * FROM libro_reclamaciones WHERE id = ?";
79         ps = cnx.prepareStatement(sql);
80         ps.setInt(1, id);
81         rs = ps.executeQuery();

```

Screenshot of Apache NetBeans IDE 22 showing the code editor for LibroReclamacionesDAO.java. The code implements methods for creating, updating, and deleting reclamations from a database.

```
81 rs = ps.executeQuery();
82
83     if (rs.next()) {
84         reclamo = new LibroReclamacionesDTO();
85         reclamo.setId(rs.getInt("id"));
86         reclamo.setNombre(rs.getString("nombre"));
87         reclamo.setCorreo(rs.getString("correo"));
88         reclamo.setTelefono(rs.getString("telefono"));
89         reclamo.setDireccion(rs.getString("direccion"));
90         reclamo.setDocumento(rs.getString("documento"));
91         reclamo.setTipo(rs.getString("tipo"));
92         reclamo.setProducto(rs.getString("producto"));
93         reclamo.setDescripcion(rs.getString("descripcion"));
94         reclamo.setPedido(rs.getString("pedido"));
95         reclamo.setFecha(rs.getTimestamp("fecha"));
96     }
97 } finally {
98     if (rs != null) rs.close();
99     if (ps != null) ps.close();
100    if (cnx != null) cnx.close();
101}
102
103 return reclamo;
104}
105
106 public boolean eliminar(int id) throws Exception {
107     Connection cnx = null;
108     PreparedStatement ps = null;
109
110     try {
111         cnx = ConectaDB.getConexion();
112         String sql = "DELETE FROM libro_reclamaciones WHERE id = ?";
113         ps = cnx.prepareStatement(sql);
114         ps.setInt(1, id);
115
116         return ps.executeUpdate() > 0;
117     } finally {
118         if (ps != null) ps.close();
119         if (cnx != null) cnx.close();
120     }
121 }
122
123 public int contarTodos() throws Exception {
124     Connection cnx = null;
125     PreparedStatement ps = null;
126     ResultSet rs = null;
127
128     try {
129         cnx = ConectaDB.getConexion();
130         String sql = "SELECT COUNT(*) as total FROM libro_reclamaciones";
131         ps = cnx.prepareStatement(sql);
132         rs = ps.executeQuery();
133
134         if (rs.next()) {
135             return rs.getInt("total");
136         }
137     } finally {
138         if (rs != null) rs.close();
139         if (ps != null) ps.close();
140         if (cnx != null) cnx.close();
141     }
142 }
143
144 }
```

Screenshot of Apache NetBeans IDE 22 showing the code editor for LibroReclamacionesDAO.java. The code implements methods for creating, updating, and deleting reclamations from a database.

```
107 Connection cnx = null;
108 PreparedStatement ps = null;
109
110 try {
111     cnx = ConectaDB.getConexion();
112     String sql = "DELETE FROM libro_reclamaciones WHERE id = ?";
113     ps = cnx.prepareStatement(sql);
114     ps.setInt(1, id);
115
116     return ps.executeUpdate() > 0;
117 } finally {
118     if (ps != null) ps.close();
119     if (cnx != null) cnx.close();
120 }
121
122 public int contarTodos() throws Exception {
123     Connection cnx = null;
124     PreparedStatement ps = null;
125     ResultSet rs = null;
126
127     try {
128         cnx = ConectaDB.getConexion();
129         String sql = "SELECT COUNT(*) as total FROM libro_reclamaciones";
130         ps = cnx.prepareStatement(sql);
131         rs = ps.executeQuery();
132
133         if (rs.next()) {
134             return rs.getInt("total");
135         }
136     } finally {
137         if (rs != null) rs.close();
138         if (ps != null) ps.close();
139         if (cnx != null) cnx.close();
140     }
141 }
142
143 }
```

Screenshot of Apache NetBeans IDE 22 showing the code editor for LibroReclamacionesDAO.java. The code implements methods for creating, updating, and deleting reclamations from a database.

```
125 PreparedStatement ps = null;
126 ResultSet rs = null;
127
128 try {
129     cnx = ConectaDB.getConexion();
130     String sql = "SELECT COUNT(*) as total FROM libro_reclamaciones";
131     ps = cnx.prepareStatement(sql);
132     rs = ps.executeQuery();
133
134     if (rs.next()) {
135         return rs.getInt("total");
136     }
137     return 0;
138 } finally {
139     if (rs != null) rs.close();
140     if (ps != null) ps.close();
141     if (cnx != null) cnx.close();
142 }
143
144 }
```

- **SugerenciaDAO**

```

package dao;

import dto.SugerenciaDTO;
import servicios.ConectaDB;
import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class SugerenciaDAO {

    public boolean insertar(SugerenciaDTO s) {
        String sql = "INSERT INTO sugerencias (nombre, correo, asunto, mensaje) VALUES (?, ?, ?, ?)";
        try (Connection cn = ConectaDB.getConexion();
             PreparedStatement ps = cn.prepareStatement(sql)) {
            ps.setString(1, s.getNombre());
            ps.setString(2, s.getCorreo());
            ps.setString(3, s.getAsunto());
            ps.setString(4, s.getMensaje());
            return ps.executeUpdate() > 0;
        } catch (Exception e) {
            e.printStackTrace();
            return false;
        }
    }

    public List<SugerenciaDTO> listarTodos() throws Exception {
        Connection cnx = null;
    }
}

```

```

package dao;

import dto.SugerenciaDTO;
import servicios.ConectaDB;
import java.sql.*;
import java.util.ArrayList;
import java.util.List;

try {
    cnx = ConectaDB.getConexion();
    String sql = "SELECT * FROM sugerencias ORDER BY fecha DESC";
    ps = cnx.prepareStatement(sql);
    rs = ps.executeQuery();

    while (rs.next()) {
        SugerenciaDTO sugerencia = new SugerenciaDTO();
        sugerencia.setId(rs.getInt("id"));
        sugerencia.setNombre(rs.getString("nombre"));
        sugerencia.setCorreo(rs.getString("correo"));
        sugerencia.setAsunto(rs.getString("asunto"));
        sugerencia.setMensaje(rs.getString("mensaje"));
        sugerencia.setFecha(rs.getTimestamp("fecha"));
        lista.add(sugerencia);
    }
} finally {
    if (rs != null) rs.close();
    if (ps != null) ps.close();
    if (cnx != null) cnx.close();
}

return lista;
}

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22 Search (Ctrl+)

Projects Files Services

```

    > RESTful Web Services
      > Source Packages
        > api
        > com.mycompany.proydesweb2
        > com.mycompany.proydesweb2.resources
        > config
          > controladores
            CarroPublicServlet.java
            CarroServlet.java
            ContactoServlet.java
            EmpresaServlet.java
            LibroReclamacionesServlet.java
            LoginServlet.java
            LogoutServlet.java
            RegistroServlet.java
            SugerenciaServlet.java
          > dao
            CarroDAO.java
            ContactoDAO.java
            EmpresaDAO.java
            LibroReclamacionesDAO.java
            SugerenciaDAO.java
            UsuarioDAO.java
          > dto
            CarroDTO.java
            ContactoDTO.java
            EmpresaDTO.java
            LibroReclamacionesDTO.java
            SugerenciaDTO.java
            UsuarioDTO.java
        > error
        > filtros
        > servicios

```

Source History

```

56     return lista;
57   }
58 }
59
60 public SugerenciaDTO buscarPorId(int id) throws Exception {
61   Connection cnx = null;
62   PreparedStatement ps = null;
63   ResultSet rs = null;
64   SugerenciaDTO sugerencia = null;
65
66   try {
67     cnx = ConectaDB.getConexion();
68     String sql = "SELECT * FROM sugerencias WHERE id = ?";
69     ps = cnx.prepareStatement(sql);
70     ps.setInt(1, id);
71     rs = ps.executeQuery();
72
73     if (rs.next()) {
74       sugerencia = new SugerenciaDTO();
75       sugerencia.setId(rs.getInt("id"));
76       sugerencia.setNombre(rs.getString("nombre"));
77       sugerencia.setCorreo(rs.getString("correo"));
78       sugerencia.setAunto(rs.getString("auento"));
79       sugerencia.setMensaje(rs.getString("mensaje"));
80       sugerencia.setFecha(rs.getTimestamp("fecha"));
81     }
82   } finally {
83     if (rs != null) rs.close();
84     if (ps != null) ps.close();
85     if (cnx != null) cnx.close();

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22 Search (Ctrl+)

Projects Files Services

```

    > RESTful Web Services
      > Source Packages
        > api
        > com.mycompany.proydesweb2
        > com.mycompany.proydesweb2.resources
        > config
          > controladores
            CarroPublicServlet.java
            CarroServlet.java
            ContactoServlet.java
            EmpresaServlet.java
            LibroReclamacionesServlet.java
            LoginServlet.java
            LogoutServlet.java
            RegistroServlet.java
            SugerenciaServlet.java
          > dao
            CarroDAO.java
            ContactoDAO.java
            EmpresaDAO.java
            LibroReclamacionesDAO.java
            SugerenciaDAO.java
            UsuarioDAO.java
          > dto
            CarroDTO.java
            ContactoDTO.java
            EmpresaDTO.java
            LibroReclamacionesDTO.java
            SugerenciaDTO.java
            UsuarioDTO.java
        > error
        > filtros
        > servicios

```

Source History

```

80       sugerencia.setFecha(rs.getTimestamp("fecha"));
81     }
82   } finally {
83     if (rs != null) rs.close();
84     if (ps != null) ps.close();
85     if (cnx != null) cnx.close();
86   }
87
88   return sugerencia;
89 }
90
91 public boolean eliminar(int id) throws Exception {
92   Connection cnx = null;
93   PreparedStatement ps = null;
94
95   try {
96     cnx = ConectaDB.getConexion();
97     String sql = "DELETE FROM sugerencias WHERE id = ?";
98     ps = cnx.prepareStatement(sql);
99     ps.setInt(1, id);
100
101   return ps.executeUpdate() > 0;
102 } finally {
103   if (ps != null) ps.close();
104   if (cnx != null) cnx.close();
105 }
106
107 public int contarTodos() throws Exception {
108   Connection cnx = null;
109   PreparedStatement ps = null;
110   ResultSet rs = null;
111
112   try {
113     cnx = ConectaDB.getConexion();
114     String sql = "SELECT COUNT(*) as total FROM sugerencias";
115     ps = cnx.prepareStatement(sql);
116     rs = ps.executeQuery();
117
118     if (rs.next()) {
119       return rs.getInt("total");
120     }
121   } finally {
122     if (rs != null) rs.close();
123     if (ps != null) ps.close();
124     if (cnx != null) cnx.close();
125   }
126
127 }
128
129 }

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22 Search (Ctrl+)

Projects Files Services

```

    > RESTful Web Services
      > Source Packages
        > api
        > com.mycompany.proydesweb2
        > com.mycompany.proydesweb2.resources
        > config
          > controladores
            CarroPublicServlet.java
            CarroServlet.java
            ContactoServlet.java
            EmpresaServlet.java
            LibroReclamacionesServlet.java
            LoginServlet.java
            LogoutServlet.java
            RegistroServlet.java
            SugerenciaServlet.java
          > dao
            CarroDAO.java
            ContactoDAO.java
            EmpresaDAO.java
            LibroReclamacionesDAO.java
            SugerenciaDAO.java
            UsuarioDAO.java
          > dto
            CarroDTO.java
            ContactoDTO.java
            EmpresaDTO.java
            LibroReclamacionesDTO.java
            SugerenciaDTO.java
            UsuarioDTO.java
        > error
        > filtros
        > servicios

```

Source History

```

106   }
107 }
108
109 public int contarTodos() throws Exception {
110   Connection cnx = null;
111   PreparedStatement ps = null;
112   ResultSet rs = null;
113
114   try {
115     cnx = ConectaDB.getConexion();
116     String sql = "SELECT COUNT(*) as total FROM sugerencias";
117     ps = cnx.prepareStatement(sql);
118     rs = ps.executeQuery();
119
120     if (rs.next()) {
121       return rs.getInt("total");
122     }
123   } finally {
124     if (rs != null) rs.close();
125     if (ps != null) ps.close();
126     if (cnx != null) cnx.close();
127   }
128
129 }

```

• UsuarioDAO

```

    package dao;

    import dto.UsuarioDTO;
    import servicios.ConectaDB;
    import java.sql.*;

    public class UsuarioDAO {

        /**
         * Valida las credenciales del usuario
         */
        public UsuarioDTO validarLogin(String correo, String contraseña) throws Exception {
            Connection cnx = null;
            PreparedStatement ps = null;
            ResultSet rs = null;
            UsuarioDTO usuario = null;

            try {
                // Obtener conexión
                cnx = ConectaDB.getConexion();

                // Log para debug
                System.out.println("==== DEBUG UsuarioDAO.validarLogin ===");
                System.out.println("Correo buscado: [" + correo + "]");
                System.out.println("Contraseña buscada: [" + contraseña + "]");

                // Primero: verificar si el correo existe
                String sqlCheck = "SELECT id, contraseña, nombre, rol FROM usuarios WHERE correo = ?";
                ps = cnx.prepareStatement(sqlCheck);
                ps.setString(1, correo);
                rs = ps.executeQuery();

                if (rs.next()) {
                    Cuenta usuarioDB = new UsuarioDAO().obtenerUsuario(rs);
                    String passwordDB = rs.getString("contraseña");
                    System.out.println("Correoooo encontrado en BD");
                    System.out.println("Contraseña en BD: [" + passwordDB + "]");
                    System.out.println("Contraseña recibida: [" + contraseña + "]");
                    System.out.println("Coincidencia? " + passwordDB.equals(contraseña));

                    // Verificar si las contraseñas coinciden
                    if (passwordDB.equals(contraseña)) {
                        usuario = new UsuarioDTO();
                        usuario.setId(rs.getInt("id"));
                        usuario.setCorreo(rs.getString("correo"));
                        usuario.setContrasena(rs.getString("contraseña"));
                        usuario.setNombre(rs.getString("nombre"));
                        usuario.setRol(rs.getString("rol"));

                        System.out.println("LOGIN EXITOSO");
                        System.out.println(" Usuario: " + usuario.getNombre());
                        System.out.println(" Rol: " + usuario.getRol());
                    } else {
                        System.err.println("X Contraseña incorrecta");
                    }
                } else {
                    System.err.println("X Correo no encontrado en la base de datos");
                }
            } catch (SQLException e) {
                System.err.println("Error SQL en validarLogin: " + e.getMessage());
                e.printStackTrace();
                throw new Exception("Error al validar login: " + e.getMessage());
            } finally {
                // Cerrar recursos
                try {

```

```

                    usuario = new UsuarioDTO();
                    usuario.setId(rs.getInt("id"));
                    usuario.setCorreo(rs.getString("correo"));
                    usuario.setContrasena(rs.getString("contraseña"));
                    usuario.setNombre(rs.getString("nombre"));
                    usuario.setRol(rs.getString("rol"));

                    System.out.println("LOGIN EXITOSO");
                    System.out.println(" Usuario: " + usuario.getNombre());
                    System.out.println(" Rol: " + usuario.getRol());
                } else {
                    System.err.println("X Contraseña incorrecta");
                }
            } catch (SQLException e) {
                System.err.println("Error SQL en validarLogin: " + e.getMessage());
                e.printStackTrace();
                throw new Exception("Error al validar login: " + e.getMessage());
            } finally {
                // Cerrar recursos
                try {

```

ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Search (Ctrl+I) 6663/730MB
Projects Files Services <default config> E T D G B P S
Source History ...va EmpresaDAO.java ContactoDAO.java LibroReclamacionesDAO.java SugerenciaDAO.java UsuarioDAO.java
Source Packages
  RESTful Web Services
    Source Packages
      api
      com.mycompany.proydesweb2
      config
        controllers
          CarroPublicService.java
          CarroServlet.java
          ContactoServlet.java
          EmpresaServlet.java
          LibroReclamacionesServlet.java
          LoginServlet.java
          LogoutServlet.java
          RegistroServlet.java
          SugerenciaServlet.java
      dao
        CarroDAO.java
        ContactoDAO.java
        EmpresaDAO.java
        LibroReclamacionesDAO.java
        SugerenciaDAO.java
        UsuarioDAO.java
      dto
        CarroDTO.java
        ContactoDTO.java
        EmpresaDTO.java
        LibroReclamacionesDTO.java
        SugerenciaDTO.java
        UsuarioDTO.java
      error
      filtros
      servicios
try {
    if (rs != null) rs.close();
    if (ps != null) ps.close();
    if (cnx != null) cnx.close();
} catch (SQLException e) {
    System.err.println("Error al cerrar recursos: " + e.getMessage());
}

return usuario;
}

/**
 * Busca un usuario por correo
 */
public UsuarioDTO buscarPorCorreo(String correo) throws Exception {
    Connection cnx = null;
    PreparedStatement ps = null;
    ResultSet rs = null;
    UsuarioDTO usuario = null;

    try {
        cnx = ConectaDB.getConnection();
        String sql = "SELECT id, correo, nombre, rol FROM usuarios WHERE correo = ?";
        ps = cnx.prepareStatement(sql);
        ps.setString(1, correo);
        rs = ps.executeQuery();

        if (rs.next()) {
            usuario = new UsuarioDTO();
            usuario.setId(rs.getInt("id"));
            usuario.setCorreo(rs.getString("correo"));
            usuario.setNombre(rs.getString("nombre"));
            usuario.setRol(rs.getString("rol"));
        }
    } catch (SQLException e) {
        System.err.println("Error en buscarPorCorreo: " + e.getMessage());
        throw new Exception("Error al buscar usuario: " + e.getMessage());
    } finally {
        try {
            if (rs != null) rs.close();
            if (ps != null) ps.close();
            if (cnx != null) cnx.close();
        } catch (SQLException e) {
            System.err.println("Error al cerrar recursos: " + e.getMessage());
        }
    }
}

return usuario;
}

/*
 * Registra un nuevo usuario
 */
public boolean registrarUsuario(UsuarioDTO usuario) throws Exception {
    Connection cnx = null;
    PreparedStatement ps = null;

    try {
        cnx = ConectaDB.getConnection();
        String sql = "INSERT INTO usuarios (correo, contraseña, nombre, rol) VALUES (?, ?, ?, ?)";
        ps = cnx.prepareStatement(sql);
        ps.setString(1, usuario.getCorreo());
        ps.setString(2, usuario.getContraseña());
        ps.setString(3, usuario.getNombre());
        ps.setString(4, usuario.getRol());
    } catch (SQLException e) {
        System.err.println("Error al registrar usuario: " + e.getMessage());
        throw new Exception("Error al registrar usuario: " + e.getMessage());
    }
}

```

ProyDesWeb2-1.0-SNAPSHOT - Apache NetBeans IDE 22

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Search (Ctrl+I) 6338/730MB
Projects Files Services <default config> E T D G B P S
Source History ...va EmpresaDAO.java ContactoDAO.java LibroReclamacionesDAO.java SugerenciaDAO.java UsuarioDAO.java
Source Packages
  RESTful Web Services
    Source Packages
      api
      com.mycompany.proydesweb2
      config
        controllers
          CarroPublicService.java
          CarroServlet.java
          ContactoServlet.java
          EmpresaServlet.java
          LibroReclamacionesServlet.java
          LoginServlet.java
          LogoutServlet.java
          RegistroServlet.java
          SugerenciaServlet.java
      dao
        CarroDAO.java
        ContactoDAO.java
        EmpresaDAO.java
        LibroReclamacionesDAO.java
        SugerenciaDAO.java
        UsuarioDAO.java
      dto
        CarroDTO.java
        ContactoDTO.java
        EmpresaDTO.java
        LibroReclamacionesDTO.java
        SugerenciaDTO.java
        UsuarioDTO.java
      error
      filtros
      servicios
        usuari.usuario.setNombre(usuario.getNombre());
        usuari.usuario.setRol(usuario.getRol());
    } catch (SQLException e) {
        System.err.println("Error en buscarPorCorreo: " + e.getMessage());
        throw new Exception("Error al buscar usuario: " + e.getMessage());
    } finally {
        try {
            if (rs != null) rs.close();
            if (ps != null) ps.close();
            if (cnx != null) cnx.close();
        } catch (SQLException e) {
            System.err.println("Error al cerrar recursos: " + e.getMessage());
        }
    }
}

return usuario;
}

/*
 * Registra un nuevo usuario
 */
public boolean registrarUsuario(UsuarioDTO usuario) throws Exception {
    Connection cnx = null;
    PreparedStatement ps = null;

    try {
        cnx = ConectaDB.getConnection();
        String sql = "INSERT INTO usuarios (correo, contraseña, nombre, rol) VALUES (?, ?, ?, ?)";
        ps = cnx.prepareStatement(sql);
        ps.setString(1, usuario.getCorreo());
        ps.setString(2, usuario.getContraseña());
        ps.setString(3, usuario.getNombre());
        ps.setString(4, usuario.getRol());
    } catch (SQLException e) {
        System.err.println("Error al registrar usuario: " + e.getMessage());
        throw new Exception("Error al registrar usuario: " + e.getMessage());
    }
}

```

```

120 Connection cnx = null;
121 PreparedStatement ps = null;
122
123 try {
124     cnx = ConectaDB.getConexion();
125     String sql = "INSERT INTO usuarios (correo, contraseña, nombre, rol) VALUES (?, ?, ?, ?)";
126     ps = cnx.prepareStatement(sql);
127     ps.setString(1, usuario.getCorreo());
128     ps.setString(2, usuario.getContraseña());
129     ps.setString(3, usuario.getNombre());
130     ps.setString(4, usuario.getRol());
131
132     int filasAfectadas = ps.executeUpdate();
133     System.out.println("Usuario registrado: " + usuario.getCorreo());
134
135     return filasAfectadas > 0;
136 } catch (SQLException e) {
137     System.err.println("Error en registrar usuario: " + e.getMessage());
138     throw new Exception("Error al registrar usuario: " + e.getMessage());
139 } finally {
140     try {
141         if (ps != null) ps.close();
142         if (cnx != null) cnx.close();
143     } catch (SQLException e) {
144         System.err.println("Error al cerrar recursos: " + e.getMessage());
145     }
146 }
147
148 /**
149 * Actualiza la contraseña de un usuario
150 */
151 public boolean actualizarContrasena(String correo, String nuevaContrasena) throws Exception {
152     Connection cnx = null;
153     PreparedStatement ps = null;
154
155     try {
156         cnx = ConectaDB.getConexion();
157         String sql = "UPDATE usuarios SET contraseña = ? WHERE correo = ?";
158         ps = cnx.prepareStatement(sql);
159         ps.setString(1, nuevaContrasena);
160         ps.setString(2, correo);
161
162         return ps.executeUpdate() > 0;
163     } catch (SQLException e) {
164         System.err.println("Error en actualizar contraseña: " + e.getMessage());
165         throw new Exception("Error al actualizar contraseña: " + e.getMessage());
166     } finally {
167         try {
168             if (ps != null) ps.close();
169             if (cnx != null) cnx.close();
170         } catch (SQLException e) {
171             System.err.println("Error al cerrar recursos: " + e.getMessage());
172         }
173     }
174 }
175
176

```

```

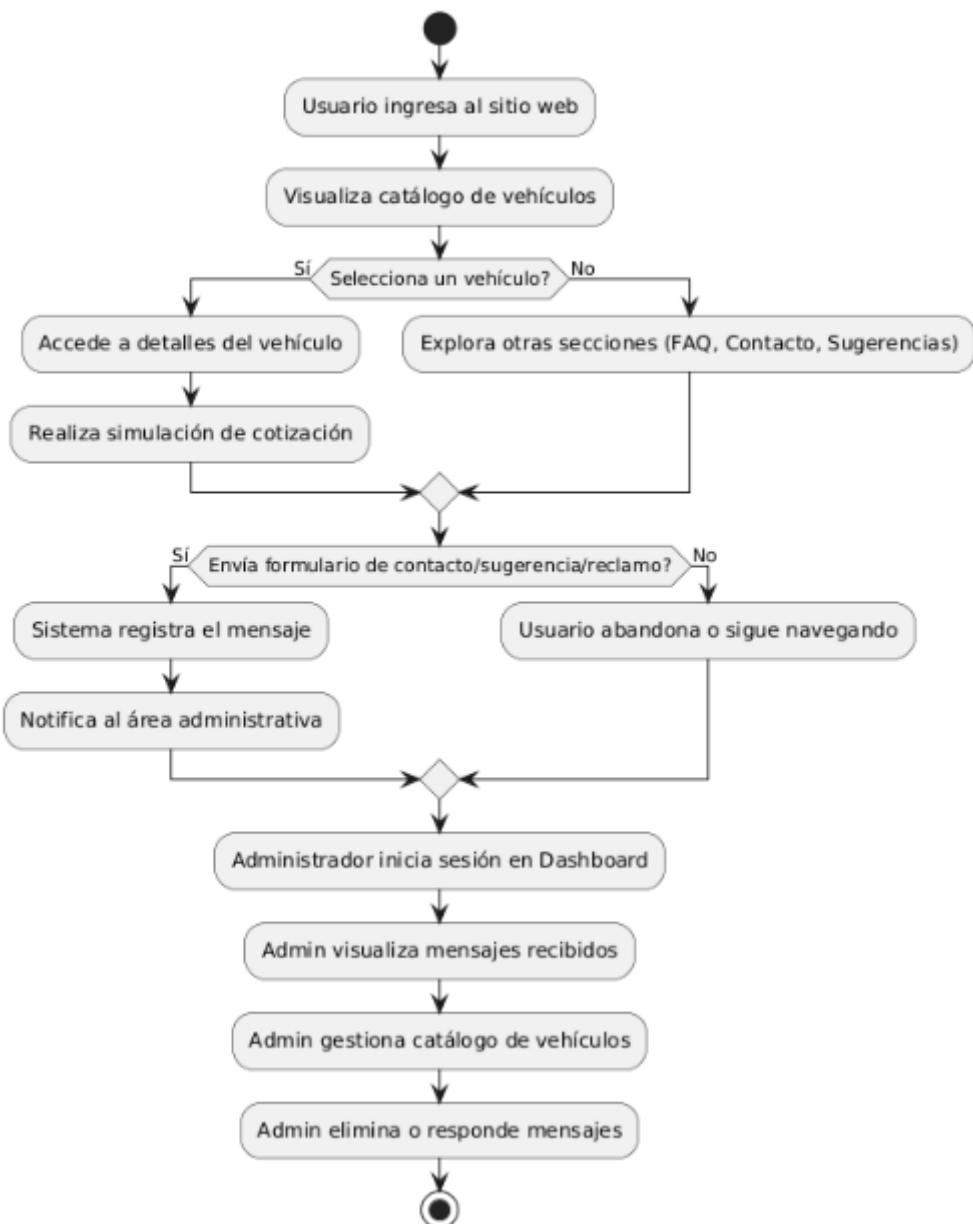
149 /**
150 * Actualiza la contraseña de un usuario
151 */
152 public boolean actualizarContrasena(String correo, String nuevaContrasena) throws Exception {
153     Connection cnx = null;
154     PreparedStatement ps = null;
155
156     try {
157         cnx = ConectaDB.getConexion();
158         String sql = "UPDATE usuarios SET contraseña = ? WHERE correo = ?";
159         ps = cnx.prepareStatement(sql);
160         ps.setString(1, nuevaContrasena);
161         ps.setString(2, correo);
162
163         return ps.executeUpdate() > 0;
164     } catch (SQLException e) {
165         System.err.println("Error en actualizar contraseña: " + e.getMessage());
166         throw new Exception("Error al actualizar contraseña: " + e.getMessage());
167     } finally {
168         try {
169             if (ps != null) ps.close();
170             if (cnx != null) cnx.close();
171         } catch (SQLException e) {
172             System.err.println("Error al cerrar recursos: " + e.getMessage());
173         }
174     }
175 }

```

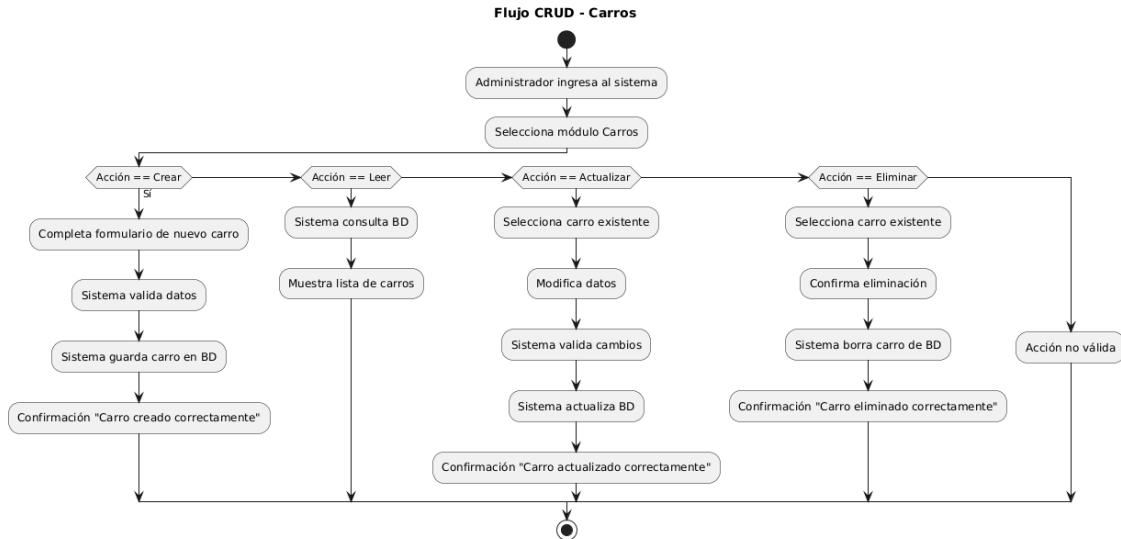
3.4 Diagramas de flujo del sistema:

- Diagrama general del sistema:

Flujo General del Sistema - Proyecto Desarrollo Web 2.0



- **Diagrama del CRUD de carros:**



Capítulo 4:

Operaciones CRUD con Datatable y AJAX:

- "listarAjax"

```

@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    HttpSession session = request.getSession(false);
    UsuarioDTO usuario = (session != null) ? (UsuarioDTO) session.getAttribute("usuario") : null;

    if (usuario == null) {
        response.sendRedirect("vistas/login.jsp");
        return;
    }

    String accion = request.getParameter("accion");
    try {
        if (accion == null) {
            List<CarroDTO> lista = dao.listarCarros();
            request.setAttribute("listaCarros", lista);
            request.getRequestDispatcher("vistas/carro/listar.jsp").forward(request, response);

        } else if ("listarAjax".equals(accion)) {
            List<CarroDTO> lista = dao.listarCarros();
            JSONArray arr = new JSONArray();
            for (CarroDTO c : lista) {
                JSONObject obj = new JSONObject();
                obj.put("id", c.getId());
                obj.put("nombre", c.getNombre());
                obj.put("descripcion", c.getDescripcion());
                obj.put("imagen", c.getImagen());
                obj.put("precio", c.getPrecio());
                arr.put(obj);
            }
            response.setContentType("application/json");
            response.setCharacterEncoding("UTF-8");
            response.getWriter().print(arr.toString());
        }
    }
}

```

- “Insertar”

```
if ("insertar".equals(accion)) {  
    dao.insertar(c);  
    ok = true;  
    json.put("success", ok);  
    json.put("message", "Carro registrado correctamente.");  
} else if ("actualizar".equals(accion)) {  
    c.setId(Integer.parseInt(request.getParameter("id")));  
    dao.actualizar(c);  
    ok = true;  
    json.put("success", ok);  
    json.put("message", "Carro actualizado correctamente.");  
} else {  
    json.put("success", false);  
    json.put("message", "Acción inválida.");  
}
```

- “Editar”

```
} else if ("editar".equals(accion)) {  
    int id = Integer.parseInt(request.getParameter("id"));  
    CarroDTO carro = dao.obtenerPorId(id);  
    request.setAttribute("carroEditar", carro);  
    request.getRequestDispatcher("vistas/carro/carroForm.jsp").forward(request, response);
```

- “Eliminar”

```
request.getRequestDispatcher("vistas/carro/carroList.jsp").forward(request, response);  
} else if ("eliminar".equals(accion)) {  
    int id = Integer.parseInt(request.getParameter("id"));  
    dao.eliminar(id);  
    response.sendRedirect("carro");  
}
```

Consumiendo Servicios Web

- Cliente Visualizador (lectura)



Cliente Visualizador para API de Carros

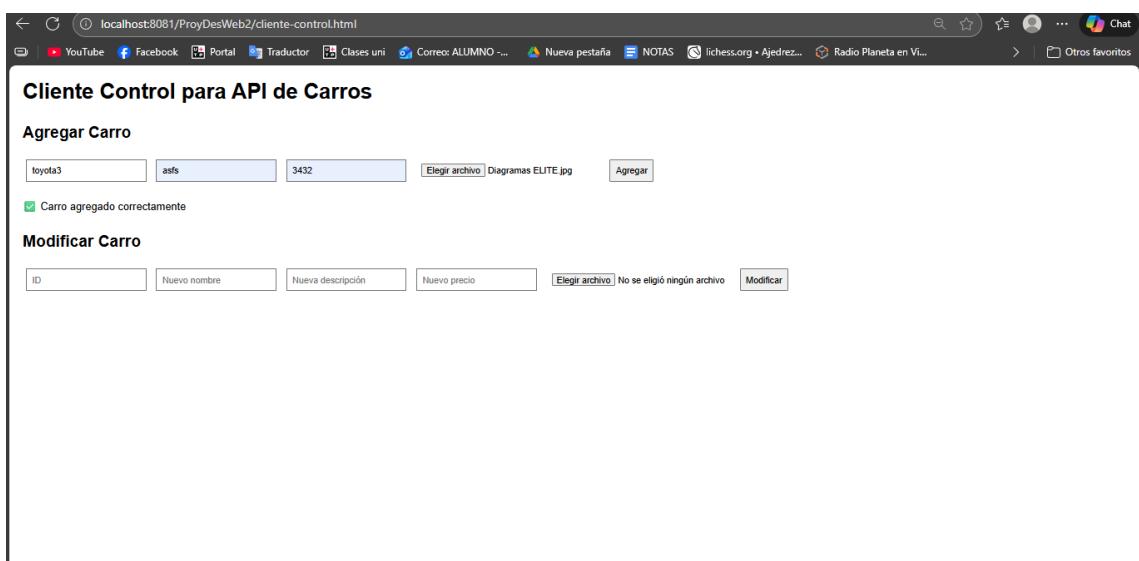
Listar todos los carros

[Listar Carros](#)

- 1 - Toyota Rush - \$5000
- 2 - Toyota Fortuner - \$68000
- 3 - Toyota Yaris - \$569200
- 4 - Toyota Hilux - \$56000
- 5 - Toyota Carola - \$60000
- 6 - Toyota Agya - \$50000
- 7 - Hilux 4x4 - \$96500
- 8 - test - \$5000
- 9 - Mazda CX-3 - \$68000
- 10 - Toyota - \$569200
- 11 - DFSK Glory 560 - \$60000
- 12 - Chevrolet Spark - \$50000
- 13 - Toyota Hilux e21 - \$131233
- 14 - 3234243 - \$324324
- 15 - 131233 - \$12332
- 16 - Kiosko - \$12312
- 19 - llanta - \$132132

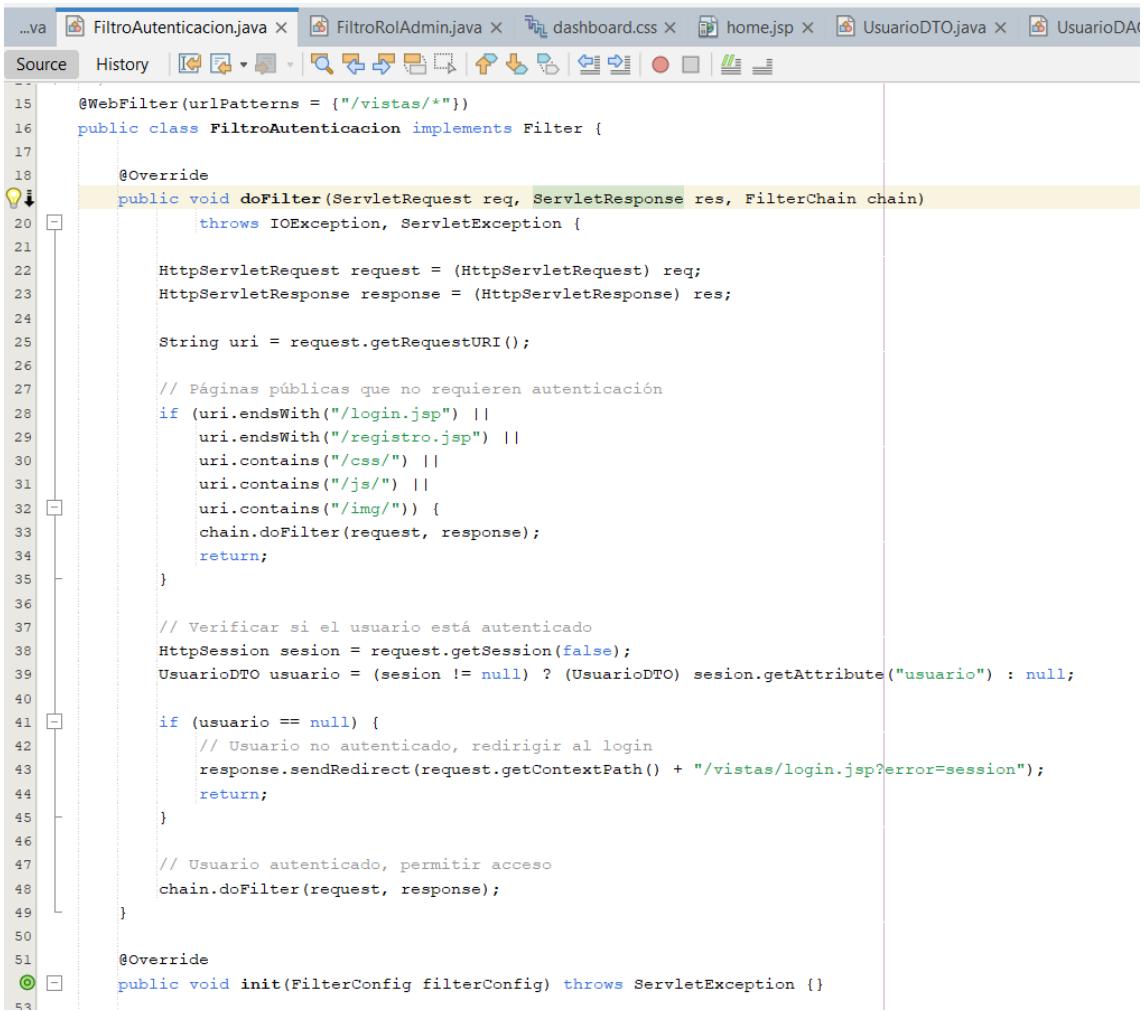
Buscar carro por ID

- Cliente Control para API de Carros (escritura)



Manejo de Sesiones

- Filtro de Autenticación



The screenshot shows a Java code editor with the file `FiltroAutenticacion.java` open. The code implements a `Filter` interface using the `@WebFilter` annotation. It checks if the user is authenticated by looking for a session attribute named "usuario". If the user is not authenticated, it redirects them to the login page. Otherwise, it allows access. The code also handles specific URL patterns that do not require authentication.

```
15  @WebFilter(urlPatterns = {"/*"})
16  public class FiltroAutenticacion implements Filter {
17
18      @Override
19      public void doFilter(ServletRequest req, ServletResponse res, FilterChain chain)
20          throws IOException, ServletException {
21
22          HttpServletRequest request = (HttpServletRequest) req;
23          HttpServletResponse response = (HttpServletResponse) res;
24
25          String uri = request.getRequestURI();
26
27          // Páginas públicas que no requieren autenticación
28          if (uri.endsWith("/login.jsp") ||
29              uri.endsWith("/registro.jsp") ||
30              uri.contains("/css/") ||
31              uri.contains("/js/") ||
32              uri.contains("/img/")) {
33              chain.doFilter(request, response);
34              return;
35          }
36
37          // Verificar si el usuario está autenticado
38          HttpSession sesion = request.getSession(false);
39          UsuarioDTO usuario = (sesion != null) ? (UsuarioDTO) sesion.getAttribute("usuario") : null;
40
41          if (usuario == null) {
42              // Usuario no autenticado, redirigir al login
43              response.sendRedirect(request.getContextPath() + "/vistas/login.jsp?error=session");
44              return;
45          }
46
47          // Usuario autenticado, permitir acceso
48          chain.doFilter(request, response);
49      }
50
51      @Override
52      public void init(FilterConfig filterConfig) throws ServletException {}
```

- Filtro de Roles

```

public class FiltroRolAdmin implements Filter {

    @Override
    public void doFilter(ServletRequest req, ServletResponse res, FilterChain chain)
            throws IOException, ServletException {

        HttpServletRequest request = (HttpServletRequest) req;
        HttpServletResponse response = (HttpServletResponse) res;

        String uri = request.getRequestURI();
        String metodo = request.getMethod(); // Detectar si es POST

        HttpSession sesion = request.getSession(false);
        UsuarioDTO usuario = (sesion != null) ? (UsuarioDTO) sesion.getAttribute("usuario") : null;

        // Permitir acceso si es POST a /sugerencia, /libro o /contacto (envío de formularios)
        if ("POST".equalsIgnoreCase(metodo)) &&
                (uri.endsWith("/sugerencia") || uri.endsWith("/libro") || uri.endsWith("/contacto"))) {
            chain.doFilter(request, response);
            return;
        }

        // Verificar autenticación
        if (usuario == null) {
            response.sendRedirect(request.getContextPath() + "/vistas/login.jsp?error=session");
            return;
        }

        // Verificar rol
        if (!usuario.esAdmin()) {
            response.sendRedirect(request.getContextPath() + "/vistas/usuario/home.jsp?error=permission");
            return;
        }

        // Usuario es admin, permitir acceso
        chain.doFilter(request, response);
    }

    @Override

```

Glosario de términos

- MVC: Modelo-Vista-Controlador, patrón de arquitectura para aplicaciones web.
- TEA: Tasa Efectiva Anual.
- Simulación financiera: cálculo de cuotas en base a monto, plazo e interés.
- Pre-cotización: documento digital con el detalle del plan de financiamiento.

Bibliografía

- Documentación oficial de Spring Boot: <https://spring.io/projects/spring-boot>
- Documentación de MySQL: <https://dev.mysql.com/doc>

- Toyota Perú – Información de modelos: <https://www.toyotaperu.com.pe>