Codigo del proyecto

Capa de datos

Clase conexion

```
package mx.com.recomendador.datos;

import java.sql.*;

/**
 *
 * @author Angel Franco
 */

public class Conexion {
    private static final String JDBC_URL =
"jdbc:mysql://localhost:3306/uaem_introduccion_ia?
useSSL=false&useTimezone=true&serverTimezone=UTC&allowPublicKeyRetrieval=true";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "L4c1b0rgv4c4#";

    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD);
    }

    public static void close(ResultSet resultado) throws SQLException {
        resultado.close();
    }

    public static void close(PreparedStatement instruccion) throws SQLException {
        instruccion.close();
    }

    public static void close(Statement instruccion) throws SQLException {
        instruccion.close();
    }

    public static void close(Connection conexion) throws SQLException {
        conexion.close();
    }
}
```

Clase RedSocialDAO

```
package mx.com.recomendador.datos;

import java.sql.Connection;
```

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import mx.com.recomendador.domain.RedSocial;

/**
 *
 * @author Angel Franco
 */
public class RedSocialDAO {

    private static final String SQL_SELECT_BY_ID = "SELECT nombre, enfoque,
contenido, tematica, audiencia, funcionalidades, ruta, descripcion " + "FROM
red_social WHERE idredsocial = ?";
    private static final String SQL_SELECT = "SELECT idredsocial, nombre, enfoque,
contenido, tematica, audiencia, funcionalidades, ruta, descripcion" + " FROM
red_social";
    private static final String SQL_INSERT = "INSERT INTO red_social(nombre,
enfoque, contenido, tematica, audiencia, funcionalidades, ruta, descripcion) " + "
VALUES(?, ?, ?, ?, ?, ?, ?, ?)";
    private static final String SQL_UPDATE = "UPDATE red_social " + "SET nombre=?,
enfoque=?, contenido=?, tematica=?, audiencia=?, funcionalidades=?, ruta=?,
descripcion=? WHERE idredsocial=?";
    private static final String SQL_DELETE = "DELETE FROM red_social WHERE
idredsocial=?";
    private Connection conexionTransacciones;

    public RedSocialDAO(Connection conexionTransacciones) {
        this.conexionTransacciones = conexionTransacciones;
    }

    public RedSocialDAO() {
    }

    public List<RedSocial> seleccionar() {
        Connection conexion = null;
        PreparedStatement instruccion = null;
        ResultSet resultado = null;
        List<RedSocial> redes = new ArrayList<>();

        try {
            conexion = this.conexionTransacciones != null ?
this.conexionTransacciones : Conexion.getConnection();
            instruccion = conexion.prepareStatement(SQL_SELECT);
            resultado = instruccion.executeQuery();

            while (resultado.next()) {
                RedSocial red = new RedSocial();
                red.setIdRedSocial(resultado.getInt("idredsocial"));
                red.setNombre(resultado.getString("nombre"));
                red.setEnfoque(resultado.getString("enfoque"));
                red.setContenido(resultado.getString("contenido"));
```

```
                red.setTematica(resultado.getString("tematica"));
                red.setAudiencia(resultado.getString("audiencia"));
                red.setFuncionalidades(resultado.getString("funcionalidades"));
                red.setRuta(resultado.getString("ruta"));
                red.setDescripcion(resultado.getString("descripcion"));

                redes.add(red);
            }
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                Conexion.close(resultado);
                Conexion.close(instruccion);
                if (this.conexionTransacciones == null) {
                    Conexion.close(conexion);
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        return redes;
    }

    public RedSocial buscar(RedSocial red) {
        Connection conexion = null;
        PreparedStatement instruccion = null;
        ResultSet resultado = null;

        try {
            conexion = this.conexionTransacciones != null ?
this.conexionTransacciones : Conexion.getConnection();
            instruccion = conexion.prepareStatement(SQL_SELECT_BY_ID);
            instruccion.setInt(1, red.getIdRedSocial());
            resultado = instruccion.executeQuery();

            if(resultado.next()) {
                String nombre = resultado.getString("nombre");
                String enfoque = resultado.getString("enfoque");
                String contenido = resultado.getString("contenido");
                String tematica = resultado.getString("tematica");
                String audiencia = resultado.getString("audiencia");
                String funcionalidades = resultado.getString("funcionalidades");
                String ruta = resultado.getString("ruta");
                String descripcion = resultado.getString("descripcion");

                red.setNombre(nombre);
                red.setEnfoque(enfoque);
                red.setContenido(contenido);
                red.setTematica(tematica);
                red.setAudiencia(audiencia);
                red.setFuncionalidades(funcionalidades);
                red.setRuta(ruta);
                red.setDescripcion(descripcion);
```

```
            }
        } catch(SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                Conexion.close(resultado);
                Conexion.close(instruccion);
                if (this.conexionTransacciones == null) Conexion.close(conexion);
            } catch(SQLException e) {
                e.printStackTrace();
            }
            return red;
        }
    }

    public void insertar(RedSocial red) throws SQLException {
        Connection conexion = null;
        PreparedStatement instruccion = null;

        try {
            conexion = this.conexionTransacciones != null ?
this.conexionTransacciones : Conexion.getConnection();
            instruccion = conexion.prepareStatement(SQL_INSERT);

            instruccion.setString(1, red.getNombre());
            instruccion.setString(2, red.getEnfoque());
            instruccion.setString(3, red.getContenido());
            instruccion.setString(4, red.getTematica());
            instruccion.setString(5, red.getAudiencia());
            instruccion.setString(6, red.getFuncionalidades());
            instruccion.setString(7, red.getRuta());
            instruccion.setString(8, red.getDescripcion());

            instruccion.executeUpdate();
        } finally {
            Conexion.close(instruccion);
            if (this.conexionTransacciones == null) {
                Conexion.close(conexion);
            }
        }
    }

    public void actualizar(RedSocial red) {
        Connection conexion = null;
        PreparedStatement instruccion = null;

        try {
            conexion = this.conexionTransacciones != null ?
this.conexionTransacciones : Conexion.getConnection();
            instruccion = conexion.prepareStatement(SQL_UPDATE);

            instruccion.setString(1, red.getNombre());
            instruccion.setString(2, red.getEnfoque());
            instruccion.setString(3, red.getContenido());
```

```
            instruccion.setString(4, red.getTematica());
            instruccion.setString(5, red.getAudiencia());
            instruccion.setString(6, red.getFuncionalidades());
            instruccion.setString(7, red.getRuta());
            instruccion.setString(8, red.getDescripcion());
            instruccion.setInt(9, red.getIdRedSocial());

            instruccion.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                Conexion.close(instruccion);
                if (this.conexionTransacciones == null) {
                    Conexion.close(conexion);
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }

        }
    }

    public void eliminar(RedSocial red) throws SQLException {
        Connection conexion = null;
        PreparedStatement instruccion = null;

        try {
            conexion = this.conexionTransacciones != null ?
this.conexionTransacciones : Conexion.getConnection();
            instruccion = conexion.prepareStatement(SQL_DELETE);

            instruccion.setInt(1, red.getIdRedSocial());
            instruccion.executeUpdate();

        } finally {
            Conexion.close(instruccion);
            if (this.conexionTransacciones == null) {
                Conexion.close(conexion);
            }
        }
    }
}
```

Clase UsuarioDAO

```
package mx.com.recomendador.datos;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
```

```
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import mx.com.recomendador.domain.Usuario;

/**
 *
 * @author Angel Franco
 */

public class UsuarioDAO {
     private static final String SQL_SELECT_BY_ID = "SELECT idusuario,
recomendacion " + "FROM usuario WHERE idusuario = ?";
     private static final String SQL_SELECT = "SELECT idusuario, recomendacion" +
" FROM usuario";
     private static final String SQL_INSERT = "INSERT INTO usuario(recomendacion)
" + " VALUES(?)";
     private static final String SQL_UPDATE = "UPDATE usuario " + "SET
recomendacion=? WHERE idusuario=?";
     private static final String SQL_DELETE = "DELETE FROM usuario WHERE
idusuario=?";
     private Connection conexionTransacciones;

    public UsuarioDAO(Connection conexionTransacciones) {
        this.conexionTransacciones = conexionTransacciones;
    }

    public UsuarioDAO() {
    }

    public List<Usuario> seleccionar() throws SQLException {
        Connection conexion = null;
        PreparedStatement instruccion = null;
        ResultSet resultado = null;
        List<Usuario> usuarios = new ArrayList<>();

        try {
            conexion = this.conexionTransacciones != null ?
this.conexionTransacciones : Conexion.getConnection();
            instruccion = conexion.prepareStatement(SQL_SELECT);
            resultado = instruccion.executeQuery();

            while (resultado.next()) {
                Usuario usuario = new Usuario();
                usuario.setIdUsuario(resultado.getInt("idusuario"));
                usuario.setRecomendacion(resultado.getInt("recomendacion"));
                usuarios.add(usuario);
            }
        } finally {
            Conexion.close(resultado);
            Conexion.close(instruccion);
            if (this.conexionTransacciones == null) {
                Conexion.close(conexion);
            }
```

```
        }
        return usuarios;
    }

    public Usuario buscar(int id) throws SQLException {
        Connection conexion = null;
        PreparedStatement instruccion = null;
        ResultSet resultado = null;
        Usuario usuario = null;

        try {
            conexion = this.conexionTransacciones != null ?
this.conexionTransacciones : Conexion.getConnection();
            instruccion = conexion.prepareStatement(SQL_SELECT_BY_ID);
            instruccion.setInt(1, id);
            resultado = instruccion.executeQuery();

            resultado.absolute(1);
            int idusuario = resultado.getInt("idusuario");
            int recomendacion = resultado.getInt("recomendacion");

            usuario.setIdUsuario(idusuario);
            usuario.setRecomendacion(recomendacion);

        } finally {
            Conexion.close(resultado);
            Conexion.close(instruccion);
            if (this.conexionTransacciones == null) {
                Conexion.close(conexion);
            }
        }
        return usuario;
    }

    public void insertar(Usuario usuario) throws SQLException  {
        Connection conexion = null;
        PreparedStatement instruccion = null;

        try {
            conexion = this.conexionTransacciones != null ?
this.conexionTransacciones : Conexion.getConnection();
            instruccion = conexion.prepareStatement(SQL_INSERT);
            instruccion.setInt(1, usuario.getRecomendacion());
            instruccion.executeUpdate();

        } finally {
            Conexion.close(instruccion);
            if (this.conexionTransacciones == null) {
                Conexion.close(conexion);
            }
        }
    }

    public void actualizar(Usuario usuario) throws SQLException {
```

```
        Connection conexion = null;
        PreparedStatement instruccion = null;

        try {
            conexion = this.conexionTransacciones != null ?
this.conexionTransacciones : Conexion.getConnection();
            instruccion = conexion.prepareStatement(SQL_UPDATE);

            instruccion.setInt(1, usuario.getRecomendacion());
            instruccion.setInt(2, usuario.getIdUsuario());
            instruccion.executeUpdate();

        } finally {
            Conexion.close(instruccion);
            if (this.conexionTransacciones == null) {
                Conexion.close(conexion);
            }
        }
    }

    public void eliminar(Usuario usuario) throws SQLException {
        Connection conexion = null;
        PreparedStatement instruccion = null;

        try {
            conexion = this.conexionTransacciones != null ?
this.conexionTransacciones : Conexion.getConnection();
            instruccion = conexion.prepareStatement(SQL_DELETE);

            instruccion.setInt(1, usuario.getIdUsuario());
            instruccion.executeUpdate();

        } finally {
            Conexion.close(instruccion);
            if (this.conexionTransacciones == null) {
                Conexion.close(conexion);
            }
        }
    }
}
```

Capa del dominio del problema

Clase RedSocial

```
package mx.com.recomendador.domain;

/**
 *
 * @author Angel Franco
 */
```

```
public class RedSocial {
    private int idRedSocial;
    private String nombre;
    private String enfoque;
    private String contenido;
    private String tematica;
    private String audiencia;
    private String funcionalidades;
    private String ruta;
    private String descripcion;

    public RedSocial(int id) {
        this.idRedSocial = id;
    }

    public RedSocial() {
    }

    public int getIdRedSocial() {
        return this.idRedSocial;
    }

    public void setIdRedSocial(int idRedSocial) {
        this.idRedSocial = idRedSocial;
    }

    public String getNombre() {
        return this.nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getEnfoque() {
        return this.enfoque;
    }

    public void setEnfoque(String enfoque) {
        this.enfoque = enfoque;
    }

    public String getContenido() {
        return this.contenido;
    }

    public void setContenido(String contenido) {
        this.contenido = contenido;
    }

    public String getTematica() {
        return this.tematica;
    }
```

```
    public void setTematica(String tematica) {
        this.tematica = tematica;
    }

    public String getAudiencia() {
        return this.audiencia;
    }

    public void setAudiencia(String audiencia) {
        this.audiencia = audiencia;
    }

    public String getFuncionalidades() {
        return this.funcionalidades;
    }

    public void setFuncionalidades(String funcionalidades) {
        this.funcionalidades = funcionalidades;
    }

    public String getRuta() {
        return this.ruta;
    }

    public void setRuta(String ruta) {
        this.ruta = ruta;
    }

    public String getDescripcion() {
        return this.descripcion;
    }

    public void setDescripcion(String descripcion) {
        this.descripcion = descripcion;
    }

    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        sb.append("idRedSocial = ").append(idRedSocial);
        sb.append(", nombre = ").append(nombre).append("\n");
        sb.append(", enfoque = ").append(enfoque).append("\n");
        sb.append(", contenido = ").append(contenido).append("\n");
        sb.append(", tematica = ").append(tematica).append("\n");
        sb.append(", audiencia = ").append(audiencia).append("\n");
        sb.append(", funcionalidades = ").append(funcionalidades).append("\n");
        sb.append(", ruta = ").append(ruta);
        sb.append(", descripcion = ").append(descripcion).append("\n");
        return sb.toString();
    }
}
```

Clase Usuario

```
package mx.com.recomendador.domain;

/**
 *
 * @author Angel Franco
 */

public class Usuario {
    private int recomendacion;
    private int idUsuario;

    public int getRecomendacion() {
        return this.recomendacion;
    }

    public void setRecomendacion(int recomendacion) {
        this.recomendacion = recomendacion;
    }

    public int getIdUsuario() {
        return this.idUsuario;
    }

    public void setIdUsuario(int idUsuario) {
        this.idUsuario = idUsuario;
    }

    public Usuario(int recomendacion) {
        this.recomendacion = recomendacion;
    }

    public Usuario() {
    }

    @Override
    public String toString() {
        return "recomendacion=" + this.recomendacion + ", idUsuario=" +
this.idUsuario;
    }
}
```

Capa de Servicio

Clase RedSocialService

```
package mx.com.recomendador.service;

import java.sql.SQLException;
```

```
import java.util.List;
import mx.com.recomendador.datos.RedSocialDAO;
import mx.com.recomendador.domain.RedSocial;

/**
 *
 * @author Angel Franco
 */

public class RedSocialService {
    private static RedSocialDAO repositorio = new RedSocialDAO();

    public static List<RedSocial> seleccionar() {
        return repositorio.seleccionar();
    }

    public static RedSocial buscar(RedSocial red) {
        return repositorio.buscar(red);
    }

    public static void actualizar(RedSocial red) {
        repositorio.actualizar(red);
    }

    public static void eliminar(RedSocial red) throws SQLException {
        repositorio.eliminar(red);
    }
}
```

Clase UsuarioService

```
package mx.com.recomendador.service;

import java.sql.SQLException;
import java.util.List;
import mx.com.recomendador.datos.UsuarioDAO;
import mx.com.recomendador.domain.Usuario;

/**
 *
 * @author Angel Franco
 */

public class UsuarioService {
    private static UsuarioDAO repositorio = new UsuarioDAO();

    public static List<Usuario> seleccionar() throws SQLException {
        return repositorio.seleccionar();
    }

    public static Usuario buscar(int id) throws SQLException {
```

```
        return repositorio.buscar(id);
    }

    public static void insertar(Usuario usuario) throws SQLException {
        repositorio.insertar(usuario);
    }

    public static void actualizar(Usuario usuario) throws SQLException {
        repositorio.actualizar(usuario);
    }

    public static void eliminar(Usuario usuario) throws SQLException {
        repositorio.eliminar(usuario);
    }
}
```

Capa de Servlets

Pregunta1Servlet

```
package mx.com.recomendador.servlets;

import java.io.IOException;
import java.util.*;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import mx.com.recomendador.domain.RedSocial;
import mx.com.recomendador.service.RedSocialService;

/**
 *
 * @author Angel Franco
 */

@WebServlet("/Pregunta1")
public class Pregunta1Servlet extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        //  accedo a la bd y obtengo las redes
        List<RedSocial> redes = new ArrayList<>();
        redes = RedSocialService.seleccionar();

        //  1era audiencia, 2da tematica, 3era enfoque, 4ta funcionalidades

        //  primera pregunta
        Set<String> audiencias = new HashSet<>();
        redes.forEach(red -> {
            //  para dar formato y no repetir palabras en audiencia
            String[] palabras = red.getAudiencia().toLowerCase().split(",?\\s+
```

```
(,\\s+)?(y\\s+)?");
            for (String palabra : palabras) audiencias.add(palabra);
        });

        //  enviar al jsp
        request.setAttribute("audiencias", audiencias);
        RequestDispatcher dispatcher =
request.getRequestDispatcher("pregunta1.jsp");
        dispatcher.forward(request, response);
    }
}
package mx.com.recomendador.servlets;

import java.io.IOException;
import java.util.*;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import mx.com.recomendador.domain.RedSocial;
import mx.com.recomendador.service.RedSocialService;

/**
 *
 * @author Angel Franco
 */

@WebServlet("/Pregunta1")
public class Pregunta1Servlet extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        //  accedo a la bd y obtengo las redes
        List<RedSocial> redes = new ArrayList<>();
        redes = RedSocialService.seleccionar();

        //  1era audiencia, 2da tematica, 3era enfoque, 4ta funcionalidades

        //  primera pregunta
        Set<String> audiencias = new HashSet<>();
        redes.forEach(red -> {
            //  para dar formato y no repetir palabras en audiencia
            String[] palabras = red.getAudiencia().toLowerCase().split(",?\\s+
(,\\s+)?(y\\s+)?");
            for (String palabra : palabras) audiencias.add(palabra);
        });

        //  enviar al jsp
        request.setAttribute("audiencias", audiencias);
        RequestDispatcher dispatcher =
request.getRequestDispatcher("pregunta1.jsp");
        dispatcher.forward(request, response);
    }
}
```

Pregunta2Servlet

```
package mx.com.recomendador.servlets;

import java.io.IOException;
import java.sql.SQLException;
import java.util.*;
import java.util.stream.Collectors;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import mx.com.recomendador.domain.RedSocial;
import mx.com.recomendador.domain.Usuario;
import mx.com.recomendador.service.RedSocialService;
import mx.com.recomendador.service.UsuarioService;

/**
 *
 * @author Angel Franco
 */

@WebServlet("/Pregunta2")
public class Pregunta2Servlet extends HttpServlet {

    protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        //  obtener respuesta
        String respuesta = request.getParameter("audiencia");

        //  pregunta 2, obtener las opciones anteriores
        List<RedSocial> redes = new ArrayList<>();
        redes = RedSocialService.seleccionar();

        //  filtrarlas por la opcion elegida
        Set<RedSocial> resultados = redes.stream().filter(red ->
red.getAudiencia().toLowerCase().contains(respuesta)).collect(Collectors.toSet());
        if(resultados.size() == 1) {
            try {
                Usuario usuario = new Usuario();

usuario.setRecomendacion(resultados.iterator().next().getIdRedSocial());
                UsuarioService.insertar(usuario);
                request.setAttribute("resultadoFinal", resultados);
                RequestDispatcher dispatcher =
request.getRequestDispatcher("resultado.jsp");
                dispatcher.forward(request, response);
            } catch (SQLException ex) {
                ex.printStackTrace();
            }
        } else {
            Set<String> tematicas = new HashSet<>();
```

```
            resultados.forEach(red -> tematicas.add(red.getTematica()));

            //  aqui agregare las respuestas
            ServletContext application = getServletContext();
            application.setAttribute("resultados1", resultados);

            request.setAttribute("tematicas", tematicas);

            RequestDispatcher dispatcher =
request.getRequestDispatcher("pregunta2.jsp");
            dispatcher.forward(request, response);
        }
    }
}
```

Pregunta3Servlet

```
package mx.com.recomendador.servlets;

import java.io.IOException;
import java.sql.SQLException;
import java.util.*;
import java.util.stream.Collectors;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import mx.com.recomendador.domain.RedSocial;
import mx.com.recomendador.domain.Usuario;
import mx.com.recomendador.service.UsuarioService;

/**
 *
 * @author Angel Franco
 */

@WebServlet("/Pregunta3")
public class Pregunta3Servlet extends HttpServlet {

    protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        String respuesta = request.getParameter("tematica");

        //  pregunta 3
        ServletContext application = getServletContext();
        Set<RedSocial> redes = (Set<RedSocial>)
application.getAttribute("resultados1");

        Set<RedSocial> resultados = redes.stream().filter(red ->
red.getTematica().contains(respuesta)).collect(Collectors.toSet());
        if(resultados.size() == 1) {
            try {
```

```
                Usuario usuario = new Usuario();

usuario.setRecomendacion(resultados.iterator().next().getIdRedSocial());
                UsuarioService.insertar(usuario);
                request.setAttribute("resultadoFinal", resultados);
                RequestDispatcher dispatcher =
request.getRequestDispatcher("resultado.jsp");
                dispatcher.forward(request, response);
            } catch (SQLException ex) {
                ex.printStackTrace();
            }
        } else {
            Set<String> enfoques = new HashSet<>();
            resultados.forEach(red -> enfoques.add(red.getEnfoque()));
            application.setAttribute("resultados2", resultados);
            request.setAttribute("enfoques", enfoques);

            RequestDispatcher dispatcher =
request.getRequestDispatcher("pregunta3.jsp");
            dispatcher.forward(request, response);
        }
    }
}
```

Pregunta4Servlet

```
package mx.com.recomendador.servlets;

import java.io.IOException;
import java.sql.SQLException;
import java.util.HashSet;
import java.util.Set;
import java.util.stream.Collectors;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import mx.com.recomendador.domain.RedSocial;
import mx.com.recomendador.domain.Usuario;
import mx.com.recomendador.service.UsuarioService;

/**
 *
 * @author Angel Franco
 */

@WebServlet("/Pregunta4")
public class Pregunta4Servlet extends HttpServlet{
```

```
    protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        //  obtenemos la respuesta
        String respuesta = request.getParameter("enfoque");

        //  obtenemos las opciones a filtrar
        ServletContext application = getServletContext();
        Set<RedSocial> redes = (Set<RedSocial>)
application.getAttribute("resultados2");

        Set<RedSocial> resultados = redes.stream().filter(red ->
red.getEnfoque().contains(respuesta)).collect(Collectors.toSet());
        if(resultados.size() == 1) {
            try {
                Usuario usuario = new Usuario();

usuario.setRecomendacion(resultados.iterator().next().getIdRedSocial());
                UsuarioService.insertar(usuario);
                request.setAttribute("resultadoFinal", resultados);
                RequestDispatcher dispatcher =
request.getRequestDispatcher("resultado.jsp");
                dispatcher.forward(request, response);
            } catch (SQLException ex) {
                ex.printStackTrace();
            }
        } else {
            Set<String> funcionalidades = new HashSet<>();
            resultados.forEach(instancia ->
funcionalidades.add(instancia.getFuncionalidades()));

            application.setAttribute("resultadosFunciones", resultados);
            request.setAttribute("funcionalidades", funcionalidades);
            RequestDispatcher dispatcher =
request.getRequestDispatcher("pregunta4.jsp");
            dispatcher.forward(request, response);
        }
    }
}
```

Capa de Test

Clase Test

```
package mx.com.recomendador.test;

import java.sql.Connection;
import java.sql.SQLException;
import java.util.*;
import java.util.stream.Collectors;
import mx.com.recomendador.datos.Conexion;
import mx.com.recomendador.datos.RedSocialDAO;
```

```
import mx.com.recomendador.domain.RedSocial;

/**
 *
 * @author Angel Franco
 */

public class Test {
    public static void main(String[] args) {
        Connection conexion = null;
        try {
            Scanner sc = new Scanner(System.in);
            conexion = Conexion.getConnection();
            if (conexion.getAutoCommit()) conexion.setAutoCommit(false);
            List<RedSocial> redes = new ArrayList<>();
            RedSocialDAO consulta = new RedSocialDAO();
            redes = consulta.seleccionar();

            //  1era audiencia, 2da tematica, 3era enfoque, 4ta funcionalidades

            //  primera pregunta
            Set<String> audiencias = new HashSet<>();
            redes.forEach(red -> {
                String[] palabras = red.getAudiencia().toLowerCase().split(",?\\s+
(,\\s+)?(y\\s+)?");
                for(String palabra : palabras) audiencias.add(palabra);
            });
            // mostrar las audiencias
            audiencias.forEach(System.out::println);
            System.out.println("");
            System.out.println("copia y pega la opcion deseada: ");
            String respuesta = sc.next();
            System.out.println("");

            //  primer filtro
            //  segunda pregunta
            Set<RedSocial> filtro1 = redes.stream().filter(red ->
red.getAudiencia().toLowerCase().contains(respuesta)).collect(Collectors.toSet());
            Set<String> tematicas = new HashSet<>();
            filtro1.forEach(instancia -> tematicas.add(instancia.getTematica()));

            //  mostrar tematicas filtradas
            tematicas.forEach(System.out::println);
            System.out.println("");
            System.out.println("copia y pega la opcion deseada: ");
            String respuesta1 = sc.next();
            System.out.println("");

            //  tercer pregunta
            Set<RedSocial> filtro2 = filtro1.stream().filter(red ->
red.getTematica().contains(respuesta1)).collect(Collectors.toSet());
            if(filtro2.size() == 1) System.out.println("Tu red social predilecta
es: " + filtro2);
            else {
```

```
                Set<String> enfoques = new HashSet<>();
                filtro2.forEach(instancia ->
enfoques.add(instancia.getEnfoque()));
                enfoques.forEach(System.out::println);
                System.out.println("");
                System.out.println("copia y pega la opcion deseada: ");
                String respuesta2 = sc.next();
                System.out.println("");

                //  cuarta pregunta
                Set<RedSocial> filtro3 = redes.stream().filter(red ->
red.getEnfoque().contains(respuesta2)).collect(Collectors.toSet());
                if(filtro3.size() == 1) System.out.println("Tu red social
predilecta es: " + filtro3);
                else {
                    Set<String> funcionalidades = new HashSet<>();
                    filtro2.forEach(instancia ->
enfoques.add(instancia.getFuncionalidades()));
                    enfoques.forEach(System.out::println);
                    System.out.println("");
                    System.out.println("copia y pega la opcion deseada: ");
                    String respuesta4 = sc.next();
                    System.out.println("");
                }
            }

        } catch (SQLException ex) {
            ex.printStackTrace(System.out);
            System.out.println("Entramos al rollback");
            try {
                conexion.rollback();
            } catch (SQLException ex1) {
                ex1.printStackTrace(System.out);
            }
        }
    }
}
```

Capa web o de presentacion

Index o inicio

```
<%--
    Document   : index
    Created on : 7 jun 2023, 10:20:39
    Author     : Angel Franco
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
    <head>
        <title>Recomendador de redes</title>
        <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
        <link rel="stylesheet" href="resources/css/estilos.css" />
    </head>
    <body>
        <h1>Recomendador de redes sociales</h1>
        <h2>Que es una red social?</h2>
        <p>
            Una red social es una plataforma en línea que permite a las personas
conectarse, comunicarse y compartir información con otras personas que comparten
intereses, actividades, amistades o relaciones en común.<br/>
            Estas plataformas brindan a los usuarios la capacidad de crear
perfiles personales, interactuar con otros usuarios a través de mensajes,
comentarios, chats o publicaciones, y compartir contenido multimedia como fotos,
videos o enlaces.<br/>
            Las redes sociales facilitan la creación y el mantenimiento de
relaciones sociales en línea, ya sea con amigos, familiares, compañeros de trabajo
o personas con intereses similares. También ofrecen características que permiten a
los<br/>
            usuarios seguir a otros usuarios, recibir actualizaciones de sus
actividades, unirse a grupos o comunidades temáticas y participar en debates o
discusiones. Además de las interacciones personales, las redes sociales<br/>
            también se utilizan para promover marcas, empresas, productos y
eventos. Muchas empresas y organizaciones utilizan las redes sociales como una
herramienta de marketing y publicidad para alcanzar a su público objetivo<br/>
            y generar interés en sus productos o servicios.
        </p>
        <h2>Caracteristicas de una red social</h2>
        <p>
            1. Perfiles de usuarios: Las redes sociales permiten a los usuarios
crear perfiles personales donde pueden proporcionar información sobre sí mismos,
como nombre, foto, ubicación, intereses, etc. Estos perfiles ayudan a
identificar<br/>
            y distinguir a los usuarios entre sí.<br/>
            2. Conexiones y amistades: Las redes sociales facilitan la conexión y
el establecimiento de relaciones entre los usuarios. Los usuarios pueden enviar
solicitudes de amistad, seguir a otros usuarios o unirse a grupos y
comunidades<br/>
            de interés compartido.<br/>
            3. Interacción y comunicación: Las redes sociales brindan diversas
formas de interactuar y comunicarse con otros usuarios. Esto puede incluir la
capacidad de enviar mensajes privados, comentarios en publicaciones, chats en
tiempo<br/>
            real o participar en discusiones en grupos.<br/>
            4. Compartir contenido: Una de las principales características de las
redes sociales es la posibilidad de compartir contenido. Los usuarios pueden
compartir fotos, videos, enlaces, actualizaciones de estado, artículos, eventos
y<br/>
            otros tipos de contenido multimedia con su red de contactos.<br/>
            5. Notificaciones y actualizaciones: Las redes sociales proporcionan
notificaciones y actualizaciones en tiempo real para mantener a los usuarios
informados sobre las actividades de sus contactos, como nuevos mensajes,<br/>
            comentarios, menciones o eventos próximos.<br/>
```

```
            6. Privacidad y configuraciones de seguridad: Las redes sociales
suelen ofrecer configuraciones de privacidad y seguridad que permiten a los
usuarios controlar quién puede ver su perfil y su contenido. Estas configuraciones
pueden<br/>
            incluir opciones para establecer la visibilidad de publicaciones,
limitar el acceso a cierta información o bloquear usuarios no deseados.<br/>
            7. Descubrimiento de contenido y recomendaciones: Muchas redes
sociales utilizan algoritmos para ofrecer a los usuarios contenido relevante y
recomendaciones basadas en sus intereses y actividades anteriores. Estos
algoritmos pueden<br/>
            mostrar publicaciones populares, sugerir amigos o recomendar grupos y
páginas relacionadas.<br/>
            8. Herramientas de seguimiento y análisis: Algunas redes sociales
proporcionan herramientas de seguimiento y análisis que permiten a los usuarios
monitorear y medir su actividad en la plataforma, como el número de seguidores,
las<br/>
            interacciones con las publicaciones y las estadísticas de
visualización.<br/>
            Estas son algunas de las características comunes que se encuentran en
las redes sociales, aunque cada plataforma puede tener características adicionales
o especializadas según su enfoque y público objetivo.
        </p>
        <h2>Realizar test</h2>
        <form action="Pregunta1" method="GET">
            <button type="submit">Empezar test</button>
        </form>
    </body>
</html>
```

pregunta1

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:c="http://java.sun.com/jsp/jstl/core">
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Pregunta</title>
        <link rel="stylesheet" href="resources/css/estilos.css" />
    </head>
    <body>
        <h1>En qué rango de edad te encuentras?</h1>
        <form action="Pregunta2" method="POST">
            <c:forEach items="${requestScope.audiencias}" var="opcion">
                <input type="radio" name="audiencia" value="${opcion}"
id="${opcion}" required="true"/>
                <label for="${opcion}">${opcion}</label>
                <br/>
                <br/>
            </c:forEach>
```

```
            <input type="submit" value="Enviar" />
        </form>
    </body>
</html>
```

pregunta2

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:c="http://java.sun.com/jsp/jstl/core">
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Pregunta</title>
        <link rel="stylesheet" href="resources/css/estilos.css" />
    </head>
    <body>
        <h1>Con que tematica te identificas mas?</h1>
        <form action="Pregunta3" method="POST">
            <c:forEach items="${requestScope.tematicas}" var="opcion">
                <input type="radio" name="tematica" value="${opcion}"
id="${opcion}" required="true"/>
                <label for="${opcion}">${opcion}</label>
                <br/>
                <br/>
            </c:forEach>
            <input type="submit" value="Enviar" />
        </form>
    </body>
</html>
```

pregunta3

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:c="http://java.sun.com/jsp/jstl/core">
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Pregunta</title>
        <link rel="stylesheet" href="resources/css/estilos.css" />
    </head>
    <body>
        <h1>Con que enfoque te identificas?</h1>
        <form action="Pregunta4" method="POST">
            <c:forEach items="${requestScope.enfoques}" var="opcion">
                <input type="radio" name="enfoque" value="${opcion}"
```

```
id="${opcion}" required="true"/>
                <label for="${opcion}">${opcion}</label>
                <br/>
                <br/>
            </c:forEach>
            <input type="submit" value="Enviar" />
        </form>
    </body>
</html>
```

pregunta4

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:c="http://java.sun.com/jsp/jstl/core">
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Pregunta</title>
        <link rel="stylesheet" href="resources/css/estilos.css" />
    </head>
    <body>
        <h1>Con que funcionalidades te identificas?</h1>
        <form action="Resultado" method="POST">
            <c:forEach items="${requestScope.funcionalidades}" var="opcion">
                <input type="radio" name="funcion" value="${opcion}"
id="${opcion}" required="true"/>
                <label for="${opcion}">${opcion}</label>
                <br/>
                <br/>
            </c:forEach>
            <input type="submit" value="Enviar" />
        </form>
    </body>
</html>
```

resultado

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:c="http://java.sun.com/jsp/jstl/core">
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Resultado</title>
        <link rel="stylesheet" href="resources/css/estilos.css" />
    </head>
```

```
    <body>
        <h1>Resultado</h1>
        <c:forEach items="#{requestScope.resultadoFinal}" var="resultado">
            <h2>${resultado.nombre}</h2>
            <img src=${resultado.ruta} alt=${resultado.ruta}>
            <p>${resultado.descripcion}</p>
            <p>Se ha guardado con exito tu recomendacion!!</p> <br/>
        </c:forEach>
    </body>
</html>
```