Diagramas de flujo del proyecto

Capa de datos

Clase Conexion

```
[Inicio]
 |
 V
[Obtener conexión]
 |
 V
[Realizar consulta]
 |
 V
[Cerrar consulta]
 |
 V
[Cerrar conexión]
 |
 V
[Fin]
```

En este diagrama, los rectángulos representan acciones o pasos específicos en la clase Conexion. A continuación, se explica cada paso en detalle:

1. Obtener conexión: Este paso representa el método getConnection(). Se encarga de establecer una conexión con la base de datos utilizando los parámetros de URL, usuario y contraseña proporcionados.
2. Realizar consulta: Este paso representa la ejecución de consultas o transacciones en la base de datos. Aquí se realizarían operaciones como insertar, actualizar o seleccionar datos.
3. Cerrar consulta: Este paso representa el cierre de la consulta, liberando los recursos utilizados por ella.
4. Cerrar conexión: Este paso representa el cierre de la conexión con la base de datos. Es importante liberar los recursos y cerrar la conexión una vez que ya no se necesite.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Clase RedSocialDAO

```
[Inicio]
 |
 V
[Seleccionar]
 |
 V
[Buscar]
```

```
 |
 V
[Insertar]
 |
 V
[Actualizar]
 |
 V
[Eliminar]
 |
 V
[Fin]
```

1. Seleccionar: Este paso representa el método seleccionar(). Se encarga de realizar una consulta para obtener una lista de objetos RedSocial.
2. Buscar: Este paso representa el método buscar(). Recibe un objeto RedSocial como parámetro y realiza una consulta para buscar y asignar los valores correspondientes al objeto.
3. Insertar: Este paso representa el método insertar(). Recibe un objeto RedSocial como parámetro y realiza una inserción en la base de datos utilizando los valores del objeto.
4. Actualizar: Este paso representa el método actualizar(). Recibe un objeto RedSocial como parámetro y realiza una actualización en la base de datos utilizando los valores del objeto.
5. Eliminar: Este paso representa el método eliminar(). Recibe un objeto RedSocial como parámetro y realiza una eliminación en la base de datos utilizando el ID del objeto.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Clase UsuarioDAO

```
[Inicio]
 |
 V
[Seleccionar]
 |
 V
[Buscar]
 |
 V
[Insertar]
 |
 V
[Actualizar]
 |
 V
[Eliminar]
 |
 V
[Fin]
```

1. Seleccionar: Este paso representa el método seleccionar(). Se encarga de realizar una consulta para obtener una lista de objetos Usuario.
2. Buscar: Este paso representa el método buscar(). Recibe un parámetro id y realiza una consulta para buscar y asignar los valores correspondientes al objeto Usuario.
3. Insertar: Este paso representa el método insertar(). Recibe un objeto Usuario como parámetro y realiza una inserción en la base de datos utilizando los valores del objeto.
4. Actualizar: Este paso representa el método actualizar(). Recibe un objeto Usuario como parámetro y realiza una actualización en la base de datos utilizando los valores del objeto.
5. Eliminar: Este paso representa el método eliminar(). Recibe un objeto Usuario como parámetro y realiza una eliminación en la base de datos utilizando el ID del objeto.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Capa de dominio del problema

Clase RedSocial

```
[Inicio]
 |
 V
[Constructor con parámetro]
 |
 V
[Constructor sin parámetro]
 |
 V
[getIdRedSocial]
 |
 V
[setIdRedSocial]
 |
 V
[getNombre]
 |
 V
[setNombre]
 |
 V
[getEnfoque]
 |
 V
[setEnfoque]
 |
 V
[getContenido]
 |
 V
```

```
[setContenido]
 |
 V
[getTematica]
 |
 V
[setTematica]
 |
 V
[getAudiencia]
 |
 V
[setAudiencia]
 |
 V
[getFuncionalidades]
 |
 V
[setFuncionalidades]
 |
 V
[getRuta]
 |
 V
[setRuta]
 |
 V
[getDescripcion]
 |
 V
[setDescripcion]
 |
 V
[toString]
 |
 V
[Fin]
```

En este diagrama, los rectángulos representan los métodos y las elipses representan los atributos de la clase RedSocial. A continuación, se explica cada paso en detalle:

1. Constructor con parámetro: Este paso representa el constructor de la clase RedSocial que recibe un parámetro id para inicializar el atributo idRedSocial.
2. Constructor sin parámetro: Este paso representa el constructor sin parámetros de la clase RedSocial.
3. getIdRedSocial y setIdRedSocial: Estos pasos representan los métodos getter y setter del atributo idRedSocial.
4. getNombre y setNombre: Estos pasos representan los métodos getter y setter del atributo nombre.
5. getEnfoque y setEnfoque: Estos pasos representan los métodos getter y setter del atributo enfoque.

6. getContenido y setContenido: Estos pasos representan los métodos getter y setter del atributo contenido.
7. getTematica y setTematica: Estos pasos representan los métodos getter y setter del atributo tematica.
8. getAudiencia y setAudiencia: Estos pasos representan los métodos getter y setter del atributo audiencia.
9. getFuncionalidades y setFuncionalidades: Estos pasos representan los métodos getter y setter del atributo funcionalidades.
10. getRuta y setRuta: Estos pasos representan los métodos getter y setter del atributo ruta.
11. getDescripcion y setDescripcion: Estos pasos representan los métodos getter y setter del atributo descripcion.
12. toString: Este paso representa el método toString() que devuelve una representación en forma de cadena de la instancia de la clase RedSocial.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Clase Usuario

```
[Inicio]
 |
 V
[Constructor con parámetro]
 |
 V
[Constructor sin parámetro]
 |
 V
[getRecomendacion]
 |
 V
[setRecomendacion]
 |
 V
[getIdUsuario]
 |
 V
[setIdUsuario]
 |
 V
[toString]
 |
 V
[Fin]
```

1. Constructor con parámetro: Este paso representa el constructor de la clase Usuario que recibe un parámetro recomendacion para inicializar el atributo recomendacion.

2. Constructor sin parámetro: Este paso representa el constructor sin parámetros de la clase Usuario.
3. getRecomendacion y setRecomendacion: Estos pasos representan los métodos getter y setter del atributo recomendacion.
4. getIdUsuario y setIdUsuario: Estos pasos representan los métodos getter y setter del atributo idUsuario.
5. toString: Este paso representa el método toString() que devuelve una representación en forma de cadena de la instancia de la clase Usuario, mostrando los valores de los atributos recomendacion e idUsuario.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Capa de servicio

Clase RedSocialService

```
[Inicio]
 |
 V
[seleccionar]
 |
 V
[buscar]
 |
 V
[actualizar]
 |
 V
[eliminar]
 |
 V
[Fin]
```

En este diagrama, los rectángulos representan los métodos de la clase RedSocialService, mientras que los óvalos representan los métodos invocados en la instancia de la clase RedSocialDAO. A continuación, se explica cada paso en detalle:

1. seleccionar: Este paso representa el método seleccionar() de la clase RedSocialService. Este método invoca el método seleccionar() de la instancia repositorio de la clase RedSocialDAO para obtener una lista de objetos RedSocial.
2. buscar: Este paso representa el método buscar(RedSocial red) de la clase RedSocialService. Este método invoca el método buscar(RedSocial red) de la instancia repositorio de la clase RedSocialDAO para buscar una instancia específica de RedSocial en función del objeto red proporcionado.
3. actualizar: Este paso representa el método actualizar(RedSocial red) de la clase RedSocialService. Este método invoca el método actualizar(RedSocial red) de la instancia repositorio de la clase RedSocialDAO para actualizar una instancia específica de RedSocial en la base de datos.

4. eliminar: Este paso representa el método eliminar(RedSocial red) de la clase RedSocialService. Este método invoca el método eliminar(RedSocial red) de la instancia repositorio de la clase RedSocialDAO para eliminar una instancia específica de RedSocial de la base de datos.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Clase UsuarioService

```
[Inicio]
 |
 V
[seleccionar]
 |
 V
[buscar]
 |
 V
[insertar]
 |
 V
[actualizar]
 |
 V
[eliminar]
 |
 V
[Fin]
```

1. seleccionar: Este paso representa el método seleccionar() de la clase UsuarioService. Este método invoca el método seleccionar() de la instancia repositorio de la clase UsuarioDAO para obtener una lista de objetos Usuario.

2. buscar: Este paso representa el método buscar(int id) de la clase UsuarioService. Este método invoca el método buscar(int id) de la instancia repositorio de la clase UsuarioDAO para buscar un usuario específico en función del id proporcionado.

3. insertar: Este paso representa el método insertar(Usuario usuario) de la clase UsuarioService. Este método invoca el método insertar(Usuario usuario) de la instancia repositorio de la clase UsuarioDAO para insertar un nuevo usuario en la base de datos.

4. actualizar: Este paso representa el método actualizar(Usuario usuario) de la clase UsuarioService. Este método invoca el método actualizar(Usuario usuario) de la instancia repositorio de la clase UsuarioDAO para actualizar un usuario existente en la base de datos.

5. eliminar: Este paso representa el método eliminar(Usuario usuario) de la clase UsuarioService. Este método invoca el método eliminar(Usuario usuario) de la instancia repositorio de la clase UsuarioDAO para eliminar un usuario existente de la base de datos.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Capa de Servlets

Pregunta1Servlet

```
[Inicio]
 |
 V
[Obtener redes]
 |
 V
[Iterar redes]
 |
 V
[Obtener audiencias]
 |
 V
[Enviar a JSP]
 |
 V
[Fin]
```

En este diagrama, los rectángulos representan las acciones y los óvalos representan los datos obtenidos o enviados. A continuación, se explica cada paso en detalle:

1. Obtener redes: En este paso, se accede a la base de datos y se obtiene una lista de redes sociales utilizando el método seleccionar() de la clase RedSocialService.

2. Iterar redes: En este paso, se itera sobre la lista de redes sociales obtenida en el paso anterior.

3. Obtener audiencias: En este paso, se procesa cada red social iterada y se obtienen las audiencias asociadas a cada una. Se utilizan métodos de la clase RedSocial para obtener las audiencias y se almacenan en un conjunto (Set) para evitar repeticiones.

4. Enviar a JSP: En este paso, se envían las audiencias obtenidas al JSP (pregunta1.jsp) utilizando el objeto request.setAttribute(). Luego, se utiliza RequestDispatcher para reenviar la solicitud y la respuesta al JSP correspondiente.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Pregunta2Servlet

```
[Inicio]
 |
 V
[Obtener respuesta]
 |
 V
[Obtener redes]
 |
 V
```

```
[Filtrar redes]
 |
 V
[Verificar cantidad de resultados]
 |
 V
[Insertar usuario y mostrar resultado final]
 |
 V
[Obtener temáticas]
 |
 V
[Guardar resultados parciales y enviar a JSP]
 |
 V
[Fin]
```

1. Obtener respuesta: En este paso, se obtiene la respuesta del usuario desde los parámetros de la solicitud. La respuesta se almacena en una variable.

2. Obtener redes: En este paso, se accede a la base de datos y se obtiene una lista de redes sociales utilizando el método seleccionar() de la clase RedSocialService.

3. Filtrar redes: En este paso, se filtran las redes sociales obtenidas en el paso anterior según la respuesta del usuario. Se utilizan métodos de la clase RedSocial y la función filter() de Java Streams para filtrar las redes que contienen la respuesta en su audiencia.

4. Verificar cantidad de resultados: En este paso, se verifica la cantidad de resultados obtenidos después de filtrar las redes sociales. Si solo hay un resultado, se procede a insertar un nuevo usuario en la base de datos con la recomendación y se muestra el resultado final en el JSP correspondiente. En caso contrario, se continúa con el siguiente paso.

5. Obtener temáticas: En este paso, se obtienen las temáticas de las redes sociales que quedaron después del filtrado. Se utiliza un conjunto (Set) para evitar repeticiones.

6. Guardar resultados parciales y enviar a JSP: En este paso, se guardan los resultados parciales (redes sociales) en el contexto de la aplicación (ServletContext) utilizando el método setAttribute(). Luego, se envían las temáticas y la respuesta anterior al JSP (pregunta2.jsp) utilizando el objeto request.setAttribute(). Por último, se utiliza RequestDispatcher para reenviar la solicitud y la respuesta al JSP correspondiente.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Pregunta3Servlet

```
[Inicio]
 |
 V
[Obtener respuesta]
 |
```

```
 V
[Obtener redes]
 |
 V
[Filtrar redes]
 |
 V
[Verificar cantidad de resultados]
 |
 V
[Insertar usuario y mostrar resultado final]
 |
 V
[Obtener enfoques]
 |
 V
[Guardar resultados parciales y enviar a JSP]
 |
 V
[Fin]
```

1. Obtener respuesta: En este paso, se obtiene la respuesta del usuario desde los parámetros de la solicitud. La respuesta se almacena en una variable.

2. Obtener redes: En este paso, se obtiene el conjunto de redes sociales que se guardaron en el contexto de la aplicación en la clase Pregunta2Servlet.

3. Filtrar redes: En este paso, se filtran las redes sociales obtenidas en el paso anterior según la respuesta del usuario. Se utilizan métodos de la clase RedSocial y la función filter() de Java Streams para filtrar las redes que contienen la respuesta en su temática.

4. Verificar cantidad de resultados: En este paso, se verifica la cantidad de resultados obtenidos después de filtrar las redes sociales. Si solo hay un resultado, se procede a insertar un nuevo usuario en la base de datos con la recomendación y se muestra el resultado final en el JSP correspondiente. En caso contrario, se continúa con el siguiente paso.

5. Obtener enfoques: En este paso, se obtienen los enfoques de las redes sociales que quedaron después del filtrado. Se utiliza un conjunto (Set) para evitar repeticiones.

6. Guardar resultados parciales y enviar a JSP: En este paso, se guardan los resultados parciales (redes sociales) en el contexto de la aplicación (ServletContext) utilizando el método setAttribute(). Luego, se envían los enfoques y la respuesta anterior al JSP (pregunta3.jsp) utilizando el objeto request.setAttribute(). Por último, se utiliza RequestDispatcher para reenviar la solicitud y la respuesta al JSP correspondiente.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Pregunta4Servlet

```
[Inicio]
 |
 V
[Obtener respuesta]
 |
 V
[Obtener redes]
 |
 V
[Filtrar redes]
 |
 V
[Verificar cantidad de resultados]
 |
 V
[Insertar usuario y mostrar resultado final]
 |
 V
[Obtener funcionalidades]
 |
 V
[Guardar resultados parciales y enviar a JSP]
 |
 V
[Fin]
```

1. Obtener respuesta: En este paso, se obtiene la respuesta del usuario desde los parámetros de la solicitud. La respuesta se almacena en una variable.

2. Obtener redes: En este paso, se obtiene el conjunto de redes sociales que se guardaron en el contexto de la aplicación en la clase Pregunta3Servlet.

3. Filtrar redes: En este paso, se filtran las redes sociales obtenidas en el paso anterior según la respuesta del usuario. Se utilizan métodos de la clase RedSocial y la función filter() de Java Streams para filtrar las redes que contienen la respuesta en su enfoque.

4. Verificar cantidad de resultados: En este paso, se verifica la cantidad de resultados obtenidos después de filtrar las redes sociales. Si solo hay un resultado, se procede a insertar un nuevo usuario en la base de datos con la recomendación y se muestra el resultado final en el JSP correspondiente. En caso contrario, se continúa con el siguiente paso.

5. Obtener funcionalidades: En este paso, se obtienen las funcionalidades de las redes sociales que quedaron después del filtrado. Se utiliza un conjunto (Set) para evitar repeticiones.

6. Guardar resultados parciales y enviar a JSP: En este paso, se guardan los resultados parciales (redes sociales) en el contexto de la aplicación (ServletContext) utilizando el método setAttribute(). Luego, se envían las funcionalidades y la respuesta anterior al JSP (pregunta4.jsp) utilizando el objeto request.setAttribute(). Por último, se utiliza RequestDispatcher para reenviar la solicitud y la respuesta al JSP correspondiente.

Finalmente, el flujo del proceso llega al rectángulo "Fin", lo que indica que el proceso ha finalizado.

Capa de Test

Clase Test

```
[Inicio]
 |
 V
[Obtener conexión]
 |
 V
[Obtener redes]
 |
 V
[Obtener audiencias]
 |
 V
[Mostrar audiencias]
 |
 V
[Obtener respuesta 1]
 |
 V
[Filtrar redes por respuesta 1]
 |
 V
[Obtener temáticas]
 |
 V
[Mostrar temáticas]
 |
 V
[Obtener respuesta 2]
 |
 V
[Filtrar redes por respuesta 2]
 |
 V
[Verificar cantidad de resultados 2]
 |
 V
[Mostrar enfoques]
 |
 V
[Obtener respuesta 3]
 |
 V
[Filtrar redes por respuesta 3]
 |
 V
[Verificar cantidad de resultados 3]
```

```
 |
 V
[Mostrar funcionalidades]
 |
 V
[Obtener respuesta 4]
 |
 V
[Fin]
```

En este diagrama, los rectángulos representan acciones y los óvalos representan datos obtenidos o enviados. A continuación, se explica cada paso en detalle:

1. Obtener conexión: En este paso, se obtiene una conexión a la base de datos utilizando la clase Conexion.
2. Obtener redes: En este paso, se obtienen las redes sociales de la base de datos utilizando la clase RedSocialDAO.
3. Obtener audiencias: En este paso, se extraen las audiencias de las redes sociales obtenidas en el paso anterior. Se utiliza un conjunto (Set) para evitar repeticiones.
4. Mostrar audiencias: En este paso, se muestran por consola las audiencias disponibles.
5. Obtener respuesta 1: En este paso, se obtiene la respuesta del usuario para la primera pregunta.
6. Filtrar redes por respuesta 1: En este paso, se filtran las redes sociales según la respuesta del usuario. Se utilizan métodos de la clase RedSocial y la función filter() de Java Streams para filtrar las redes que contienen la respuesta en su audiencia.
7. Obtener temáticas: En este paso, se extraen las temáticas de las redes sociales obtenidas en el paso anterior. Se utiliza un conjunto (Set) para evitar repeticiones.
8. Mostrar temáticas: En este paso, se muestran por consola las temáticas filtradas.
9. Obtener respuesta 2: En este paso, se obtiene la respuesta del usuario para la segunda pregunta.
10. Filtrar redes por respuesta 2: En este paso, se filtran las redes sociales según la respuesta del usuario. Se utilizan métodos de la clase RedSocial y la función filter() de Java Streams para filtrar las redes que contienen la respuesta en su temática.
11. Verificar cantidad de resultados 2: En este paso, se verifica la cantidad de resultados obtenidos después de filtrar las redes sociales. Si solo hay un resultado, se muestra por consola la red social predilecta. En caso contrario, se continúa con el siguiente paso.
12. Mostrar enfoques: En este paso, se muestran por consola los enfoques disponibles.
13. Obtener respuesta 3: En este paso, se obtiene la respuesta del usuario para la tercera pregunta.
14. Filtrar redes por respuesta 3: En este paso, se filtran las redes sociales según la respuesta del usuario. Se utilizan métodos de la clase RedSocial y la función filter() de Java Streams para filtrar las redes que contienen la respuesta en su enfoque.

15. Verificar cantidad de resultados 3: En este paso, se verifica la cantidad de resultados obtenidos después de filtrar las redes sociales. Si solo hay un resultado, se muestra por consola la red social predilecta. En caso contrario, se continúa con el siguiente paso.
16. Mostrar funcionalidades: En este paso, se muestran por consola las funcionalidades disponibles.
17. Obtener respuesta 4: En este paso, se obtiene la respuesta del usuario para la cuarta pregunta.
18. Fin: El flujo del programa llega al rectángulo "Fin", lo que indica que el programa ha terminado.