Here's another example for students to implement, focusing on **Session Tracking Using URL Rewriting** in NetBeans.

Implementing Session Tracking Using URL Rewriting in NetBeans

Step-by-Step Example

1. **Create a New Servlet Project:**
    - Open NetBeans.
    - Go to **File > New Project**, select **Java Web > Web Application**, and click **Next**.
    - Name the project URLRewritingExample and select **Apache Tomcat** as the server.
2. **Create the First Servlet:** This servlet will create a session and store user information using URL rewriting.
    - Right-click on **Source Packages** and create a new servlet named StartSessionServlet.

```
import java.io.IOException;
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 javax.servlet.http.HttpSession;

@WebServlet("/startsession")
public class StartSessionServlet extends HttpServlet {

   protected void doGet(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {
      // Create a new session or retrieve an existing one
      HttpSession session = request.getSession();

      // Set an attribute "username" with a sample value
      session.setAttribute("username", "JaneDoe");

      // Send a response with a link to the next servlet, using URL rewriting
      String url = response.encodeURL("retrieveSession");
      response.setContentType("text/html");
      response.getWriter().println("Session started. <a href='" + url + "'>Go to Retrieve
Session</a>");
   }
}
```

3. **Create the Second Servlet:** This servlet will retrieve and display the session attribute.

- Right-click and create another servlet named RetrieveSessionServlet.

```
import java.io.IOException;
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 javax.servlet.http.HttpSession;

@WebServlet("/retrieveSession")
public class RetrieveSessionServlet extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        // Retrieve the session
        HttpSession session = request.getSession(false);
        String username = null;

        // Check if session exists and get the "username" attribute
        if (session != null) {
            username = (String) session.getAttribute("username");
        }

        // Set response content type and display the username
        response.setContentType("text/html");
        if (username != null) {
            response.getWriter().println("Welcome back, " + username + "!");
        } else {
            response.getWriter().println("No active session found.");
        }
    }
}
```

4. **Run the Project:**
    - Access the first servlet via http://localhost:8080/URLRewritingExample/startsession to create the session and store the user attribute.
    - Click the generated link to visit http://localhost:8080/URLRewritingExample/retrieveSession, where the second servlet retrieves and displays the session data ("Welcome back, JaneDoe!").

A breakdown of each component and its purpose within this example on **Session Tracking Using URL Rewriting**.

Step 1: Create a New Servlet Project

- **Project Creation**: In this step, you create a new web application project in NetBeans, which serves as a container for our servlets.
- **Project Naming**: Naming the project URLRewritingExample helps identify it easily.
- **Server Selection**: Apache Tomcat is chosen as the server to run our Java-based web application.

Step 2: Create the First Servlet (StartSessionServlet)

- **Servlet Purpose**: This servlet initiates a user session and stores a piece of information (username) in the session. It then generates a link to the second servlet, using URL rewriting to maintain session tracking without cookies.

Code Components in StartSessionServlet

```
import java.io.IOException;
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 javax.servlet.http.HttpSession;
```

- **Imports**: These statements import necessary libraries for servlets, handling HTTP requests/responses, and managing sessions.

```
@WebServlet("/startsession")
```

- **Annotation**: @WebServlet("/startsession") maps this servlet to the URL /startsession. When a user visits this URL, the code in this servlet will execute.

```
public class StartSessionServlet extends HttpServlet {
```

- **Servlet Class**: Defines the StartSessionServlet class, which extends HttpServlet to inherit servlet functionality.

```
HttpSession session = request.getSession();
```

- **Session Creation**: request.getSession() either creates a new session if it doesn't exist or retrieves the existing one for this user.

```
session.setAttribute("username", "JaneDoe");
```

- **Setting Session Attribute**: session.setAttribute("username", "JaneDoe") stores the username in the session with the key "username".

```
String url = response.encodeURL("retrieveSession");
```

- **URL Rewriting**: response.encodeURL("retrieveSession") generates a URL that includes the session ID. This way, even if cookies are disabled, the session remains trackable.

    response.setContentType("text/html");

    response.getWriter().println("Session started. Go to Retrieve Session");

- **Response**: The servlet sets the response type to HTML and sends a clickable link with the rewritten URL to the user, allowing them to access the RetrieveSessionServlet.

Step 3: Create the Second Servlet (RetrieveSessionServlet)

- **Servlet Purpose**: This servlet retrieves the session and reads the username attribute, then displays a welcome message using the stored data.

Code Components in RetrieveSessionServlet

```
import java.io.IOException;
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 javax.servlet.http.HttpSession;
```

- **Imports**: These imports allow the servlet to work with HTTP requests, responses, and sessions.

```
@WebServlet("/retrieveSession")
```

- **Annotation**: @WebServlet("/retrieveSession") maps this servlet to the /retrieveSession URL, so it responds when the user clicks the link generated in StartSessionServlet.

```
public class RetrieveSessionServlet extends HttpServlet {
```

- **Servlet Class**: Defines RetrieveSessionServlet, extending HttpServlet to use its methods for handling HTTP requests.

```
HttpSession session = request.getSession(false);
```

- **Retrieving Session**: request.getSession(false) fetches the existing session without creating a new one. If no session exists, it returns null.

```
String username = null;
if (session != null) {
    username = (String) session.getAttribute("username");
}
```

- **Session Check and Attribute Retrieval**: Here, the code checks if a session exists. If so, it retrieves the username attribute.

```
response.setContentType("text/html");
if (username != null) {
    response.getWriter().println("Welcome back, " + username + "!");
} else {
    response.getWriter().println("No active session found.");
}
```

- **Response**: Sets the response type as HTML and displays a welcome message if the session and username attribute exist. If not, it informs the user that no active session was found.

Step 4: Run the Project

- **Accessing the Servlets**:
    - Visiting http://localhost:8080/URLRewritingExample/startsession runs StartSessionServlet, starts a session, and sets the username attribute. It also generates a link to RetrieveSessionServlet.
    - Clicking this link (or visiting http://localhost:8080/URLRewritingExample/retrieveSession) runs RetrieveSessionServlet, which retrieves and displays the session data.

Summary of Key Concepts

1. **Session Tracking**: The process of maintaining user-specific data across multiple requests using session objects.
2. **URL Rewriting**: Embedding session IDs in URLs to track sessions when cookies aren't used.
3. **Session Attribute**: Storing key-value pairs (e.g., "username": "JaneDoe") in a session for later retrieval.
4. **Servlet Chaining**: Linking servlets in sequence to handle different stages of a process.