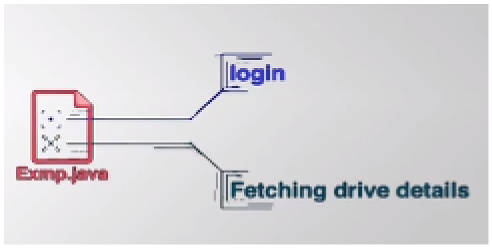
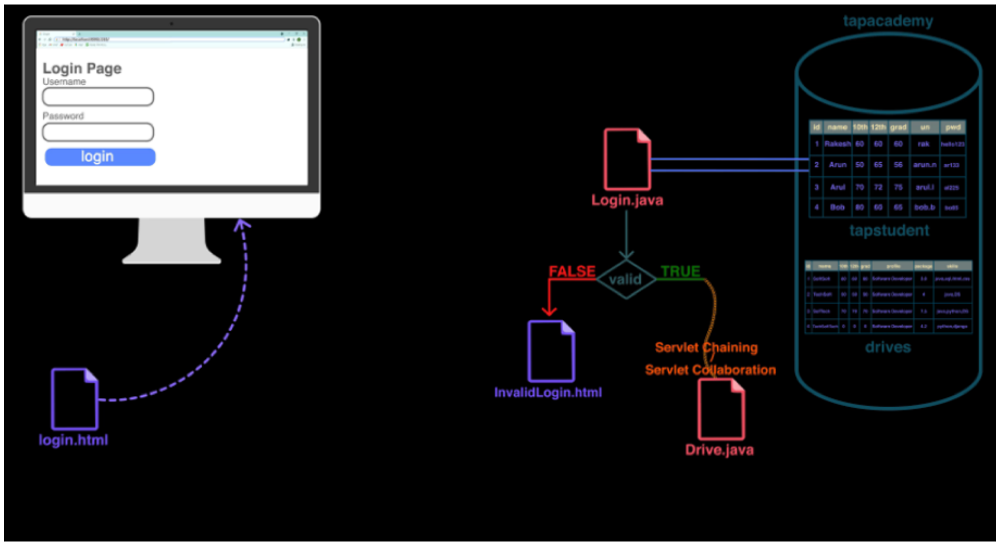
Day - 8
Servlet Chaining

In the previous session, we had seen that our servlet **Exmp.java** was performing two activities i.e, **login activity and Fetching drive details.**



But in real time scenarios, a servlet should perform only one activity at a time It should always follow the ***Single Responsibility Principle*** where a single class should have a **single responsibility**. So we will split the **Exmp.java** servlet into two different classes **Login.java** and **Drive.java**, where Login.java servlet's responsibility would be to handle login details and Drives.java servlet's responsibility would be to fetch drive details.
When the client requests for login page, then **login.html** is sent as response and after the client enters username & password and clicks on submit button. If it is a invalid login details then the request should be forwarded to **invalidlogin.html** but if its a valid login details then Login.java servlet now chains the control to Drives.java servlet. The process of chaining the control from one servlet to another is called **Servlet Chaining** or **Servlet Collaboration**.



For that we require the **RequestDispatcher** object. It is the duty of the RequestDispatcher object to forward the request from one file to another.
Let us now try to implement this in our code, i.e., separate the login details and fetching drive details into two different files.

1. Create two servlets (**Validation.java** and **Drive.java**) in the ***com.tap.student*** package
   Note: We have named the Login.java file as Validation.java as a file Login was already present in our project

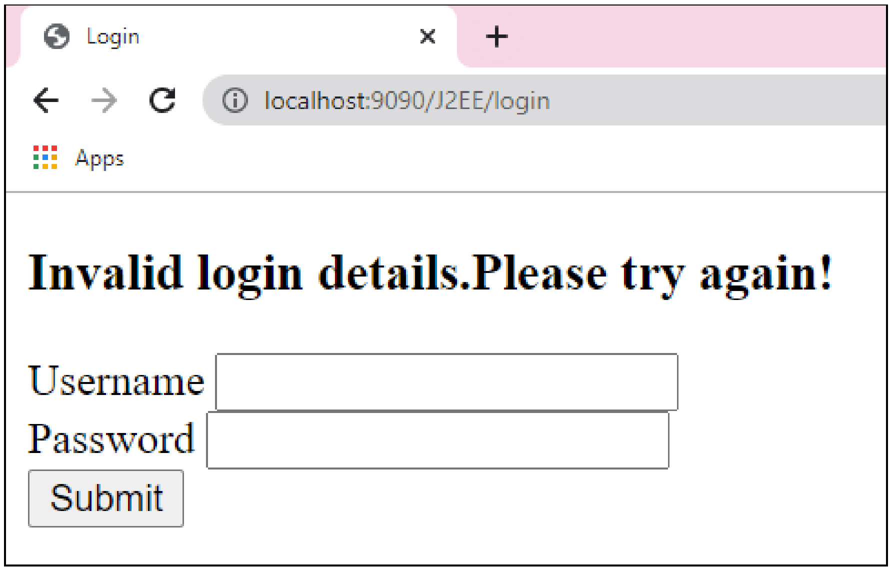
```java
package com.tap.student;

public class Validation extends HttpServlet {
	Connection con = null;
	PreparedStatement pstmt = null;
	ResultSet res = null;
	String url = "jdbc:mysql://localhost:3306/tapacademy";
	String un = "root";
	String pwd = "root";

	@Override
	public void init() throws ServletException {
		try {
			Class.forName("com.mysql.cj.jdbc.Driver");
			con = DriverManager.getConnection(url, un, pwd);

		} catch (Exception e) {
			e.printStackTrace();
		}

	}

	@Override
	protected void doPost(HttpServletRequest req, HttpServletResponse resp)
			throws ServletException, IOException {

		resp.setContentType("text/html");
		PrintWriter writer =  resp.getWriter();

		String username = req.getParameter("username");
		String password = req.getParameter("password");

		try {
			String query = "select * from tapstudent where un = ? and pwd = ?";
			pstmt = con.prepareStatement(query);
			pstmt.setString(1, username);
			pstmt.setString(2, password);
			res = pstmt.executeQuery();

			if (res.next()==true) {

			}
			else {
				RequestDispatcher rd = 
req.getRequestDispatcher("/invalidlogin.html");
				rd.forward(req, resp);
```

```
                    }

                } catch (Exception e) {
                    e.printStackTrace();
                }
        }

}
```

Now change the URL-mapping in web.xml as shown below

```
<servlet>
      <servlet-name>Login</servlet-name>
      <servlet-class>com.tap.student.Validation</servlet-class>
</servlet>

<servlet-mapping>
      <servlet-name>Login</servlet-name>
      <url-pattern>/login</url-pattern>
</servlet-mapping>
```

So when the user enters invalid login details then **Validation** Servlet will forward the request to **Invalidlogin.html**



Invalid login details.Please try again!

Username
Password
Submit

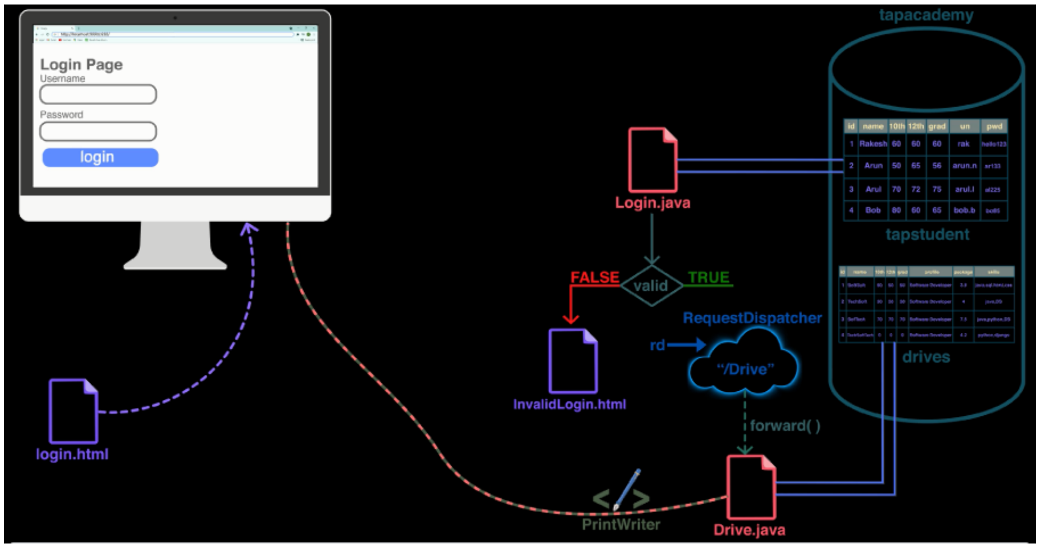
Now if the user enters valid login details, then the Validation servlet has to chain the control to the Drive servlet.
Let us write the logic for that.
We have to first register the Drive servlet in the deployment descriptor so that when we mention the path in the RequestDispatcher object, it can map it to Drive.java.

```
<servlet>
      <servlet-name>Drive</servlet-name>
      <servlet-class>com.tap.student.Drive</servlet-class>
</servlet>

<servlet-mapping>
      <servlet-name>Drive</servlet-name>
      <url-pattern>/drive</url-pattern>
</servlet-mapping>
```

Now that our servlet is registered in the deployment descriptor, let us now write the code-

```
if (res.next()==true)
{
   req.getRequestDispatcher("/drive").forward(req, resp);;
}
```

The RequestDispatcher object will now forward the request to the **Drive.java** servlet, and the Drive servlet would be connected to the drives table and now with the help of **PrintWriter**, we can display all the data present in the drives table.



id	name	10th	12th	grad	un	pwd
1	Rakesh	60	60	60	rak	hello123
2	Arun	50	65	56	arun.n	ar133
3	Arul	70	72	75	arul.l	al225
4	Bob	80	60	65	bob.b	bo85

id	name	10th	12th	grad	profile	package	skills
1	[illegible]	60	60	60	Software Developer	3.8	java,sql,html,css
2	TechSoft	90	90	90	Software Developer	4	java,DS
3	SofTech	70	70	70	Software Developer	7.5	java,python,DS
4	DarkSoftTech	0	0	0	Software Developer	4.2	python,django

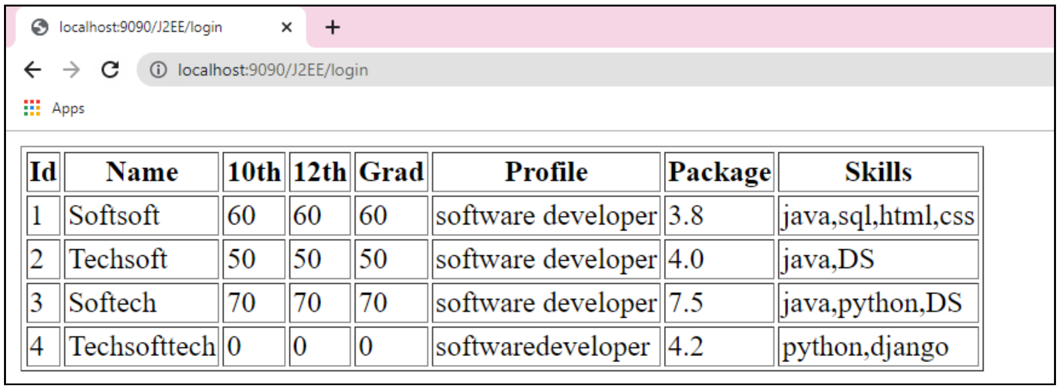
Type in the following code in Drive.java

```
package com.tap.student;

public class Drive extends HttpServlet {
      Connection con = null;
      PreparedStatement pstmt = null;
      ResultSet res = null;
      String url = "jdbc:mysql://localhost:3306/tapacademy";
      String un = "root";
      String pwd = "root";
```

```java
@Override
public void init() throws ServletException {
      try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            con = DriverManager.getConnection(url, un, pwd);

      } catch (Exception e) {
            e.printStackTrace();
      }

}

@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp)
      throws ServletException, IOException {

      PrintWriter writer =  resp.getWriter();
      try {
            String query2 = "select * from drives";
            Statement stmt = con.createStatement();
            ResultSet res2 = stmt.executeQuery(query2);

            writer.println("<table border=\"1\">\r\n"
                        + "\r\n"
                        + "   <tr>\r\n"
                        + "         <th>Id</th>\r\n"
                        + "         <th>Name</th>\r\n"
                        + "         <th>10th</th>\r\n"
                        + "         <th>12th</th>\r\n"
                        + "         <th>Grad</th>\r\n"
                        + "         <th>Profile</th>\r\n"
                        + "         <th>Package</th>\r\n"
                        + "         <th>Skills</th>   \r\n"
                        + "   </tr>");

            while(res2.next()==true){
                  int id = res2.getInt(1);
                  String name = res2.getString(2);
                  int ten = res2.getInt(3);
                  int twe = res2.getInt(4);
                  int grad = res2.getInt(5);
                  String profile = res2.getString(6);
                  float pac = res2.getFloat(7);
                  String skills = res2.getString(8);
```

```
                        writer.println("<tr>\r\n"
                                    + "         <td>" + id + "</td>\r\n"
                                    + "         <td>" + name + "</td>\r\n"
                                    + "         <td>" + ten + "</td>\r\n"
                                    + "         <td>" + twe + "</td>\r\n"
                                    + "         <td>" + grad + "</td>\r\n"
                                    + "         <td>" + profile + "</td>\r\n"
                                    + "         <td>" + pac + "</td>\r\n"
                                    + "         <td>" + skills + "</td>\r\n"
                                    + "   </tr>");
                  }
                  writer.println("</table>");

            } catch (Exception e) {
                  e.printStackTrace();
            }

      }
}
```

And when we deploy the project in the browser and enter a valid username & password, we will get the following-



Id	Name	10th	12th	Grad	Profile	Package	Skills
1	Softsoft	60	60	60	software developer	3.8	java,sql,html,css
2	Techsoft	50	50	50	software developer	4.0	java,DS
3	Softech	70	70	70	software developer	7.5	java,python,DS
4	Techsofttech	0	0	0	softwaredeveloper	4.2	python,django

Since the chaining happens only once, it is called **One Level Chaining.**