LAB 8: DATABASE & JSP

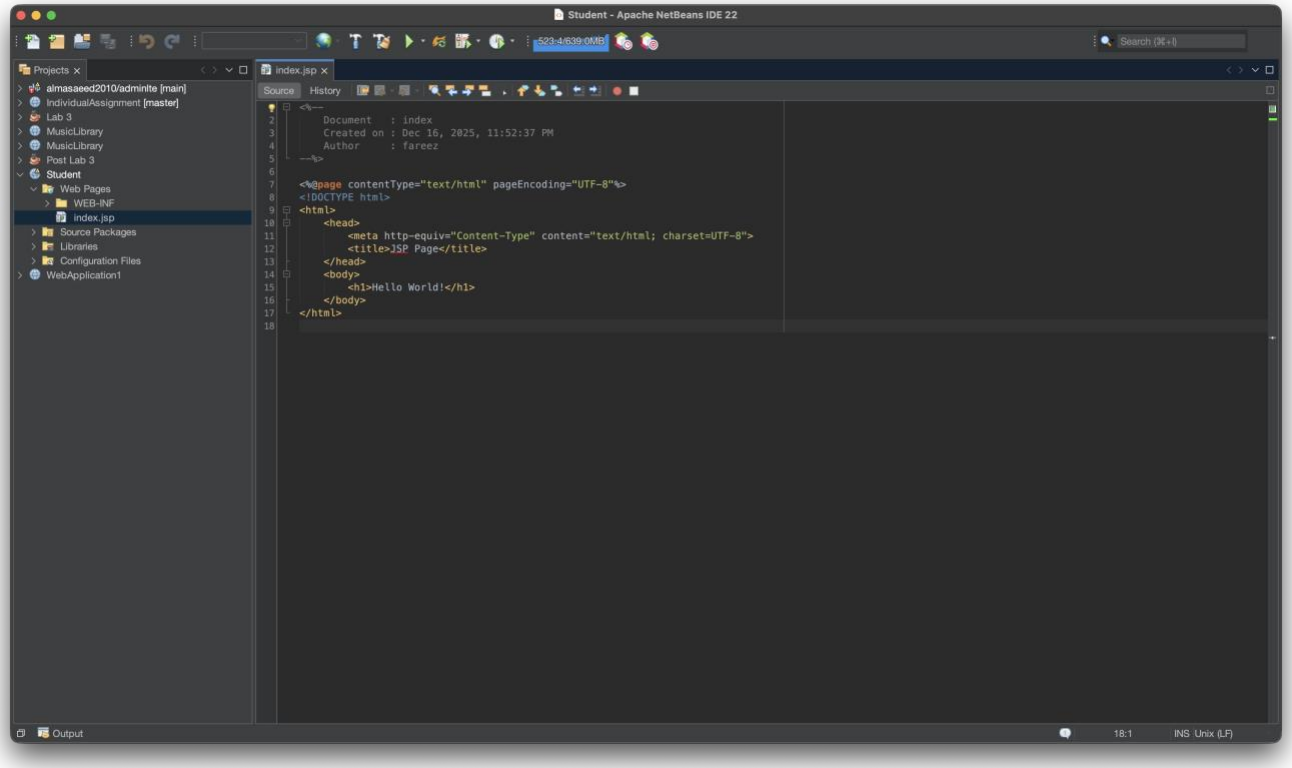
Lab Activities

Exercise 1: Creating a new web application Task

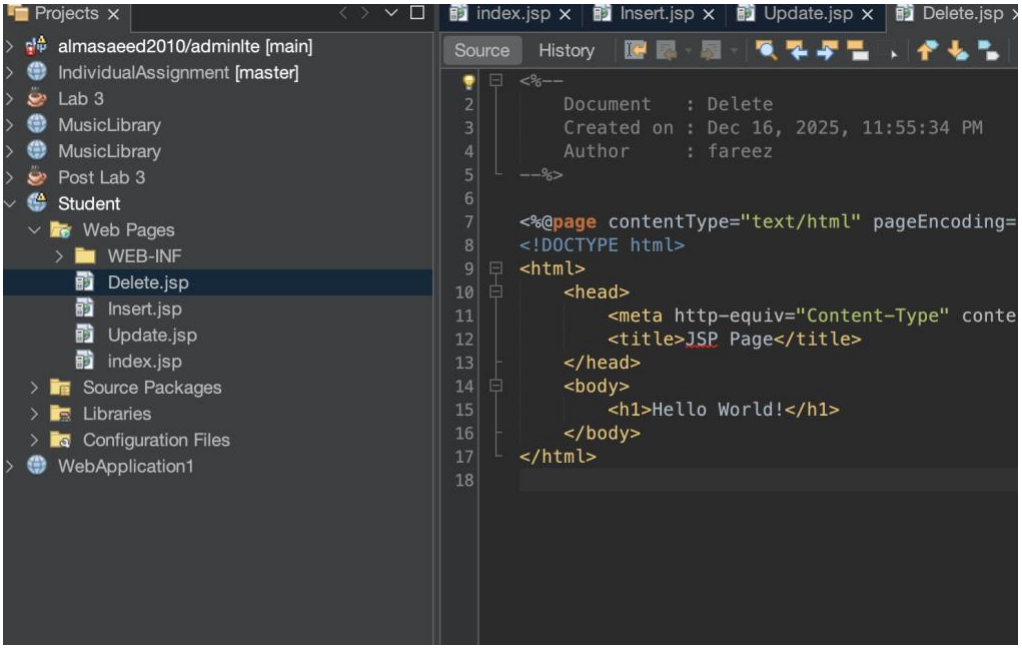
1: Create a new project for web application

a) Create the Student Web application project with the following characteristic: Project: Student
Location: C:\Users\User\Documents\NetBeansProjects
Folder: C:\Users\User\Documents\NetBeansProjects\Student
Server : Personal glassfish
Java EE version: Java EE 5
Contextpath: /Student

Index.jsp page is automatically created by default. Now, you have to create other three JSP pages.



b) Create the Insert.jsp, Update.jsp and Delete.jsp

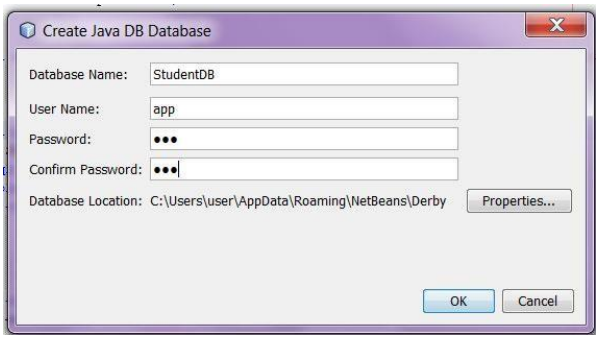


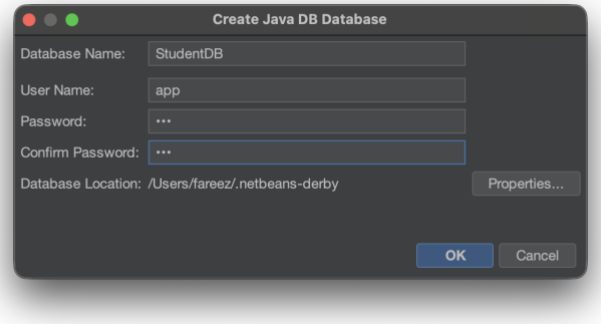
Exercise 2: Creating database and database connection Task

1: Create a new database

a) Go to services tab.

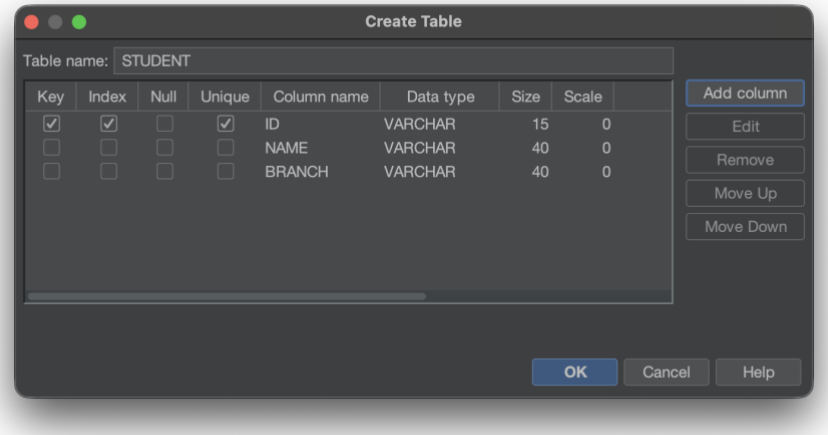
b) Right click on Java DB and chose CreateDatabase Fill in the database details here.





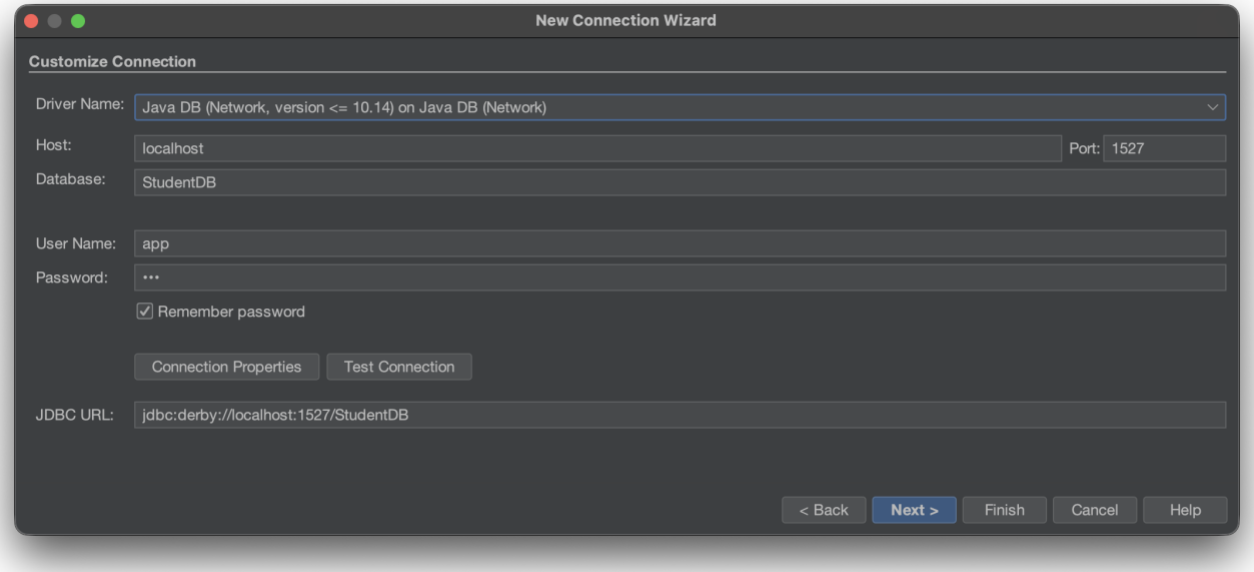
c) Right click the StudentDB database, select connect.
d) Create a table STUDENT in jdbc:derby://localhost:1527/StudentDB [public on PUBLIC] -> APP – Tables -> right click -> Create Table.
e) Create the following columns:

COLUMN NAME	TYPE	DETAIL
ID	VARCHAR(15)	PRIMARY KEY
NAME	VARCHAR(40)	NOT NULL
BRANCH	VARCHAR(40)	NOT NULL



Task 2: Create a new database connection

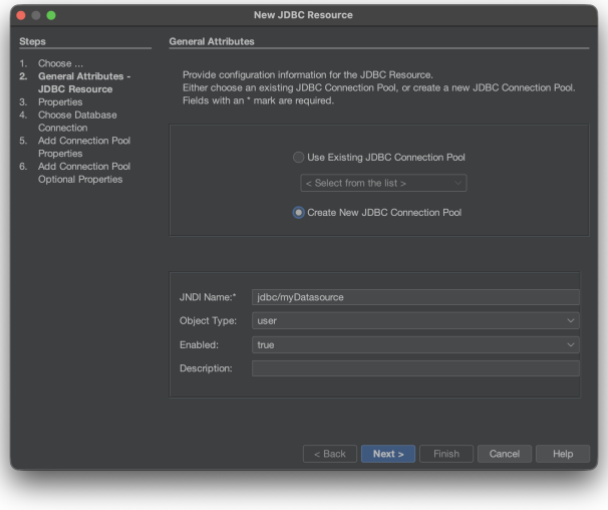
a) Right click on Databases and choose New Connection...
b) Fill the details as below:
   Data Input Mode: Field Entry
   Name: Java DB (Network)
   Host: localhost
   Port: 1527
   Database: StudentDB
   Username: app
   Password: app



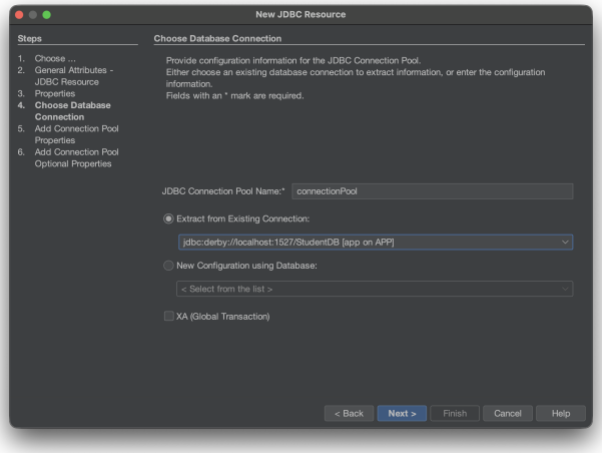
Exercise 3: Creating Database Connection pool in application

a) Go to project window. Right click on server resources -> New -> Others. Select Student project.

b) In Categories chose GlassFish , File Type select JDBC Resource and click next. Select create New JDBC Connection Pool, Give JNDI name jdbc/myDatasource.

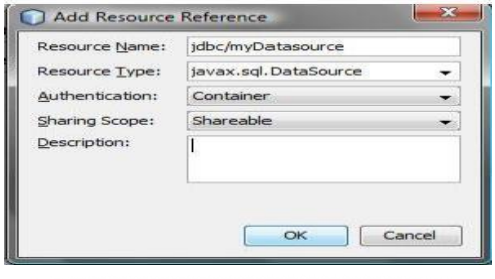


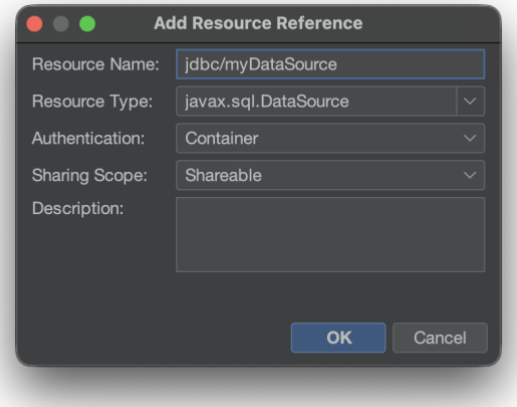
Then give next, so it will ask for properties at that window just click next. You will see window shown below.



c) Give connection pool name. Select a connection that you have created previously in services and click next. You will see connection pool properties. Now click finish button.

d) Open Web pages -> WEB-INF -> web.xml. Click on references tab, in Resource references click on Add.



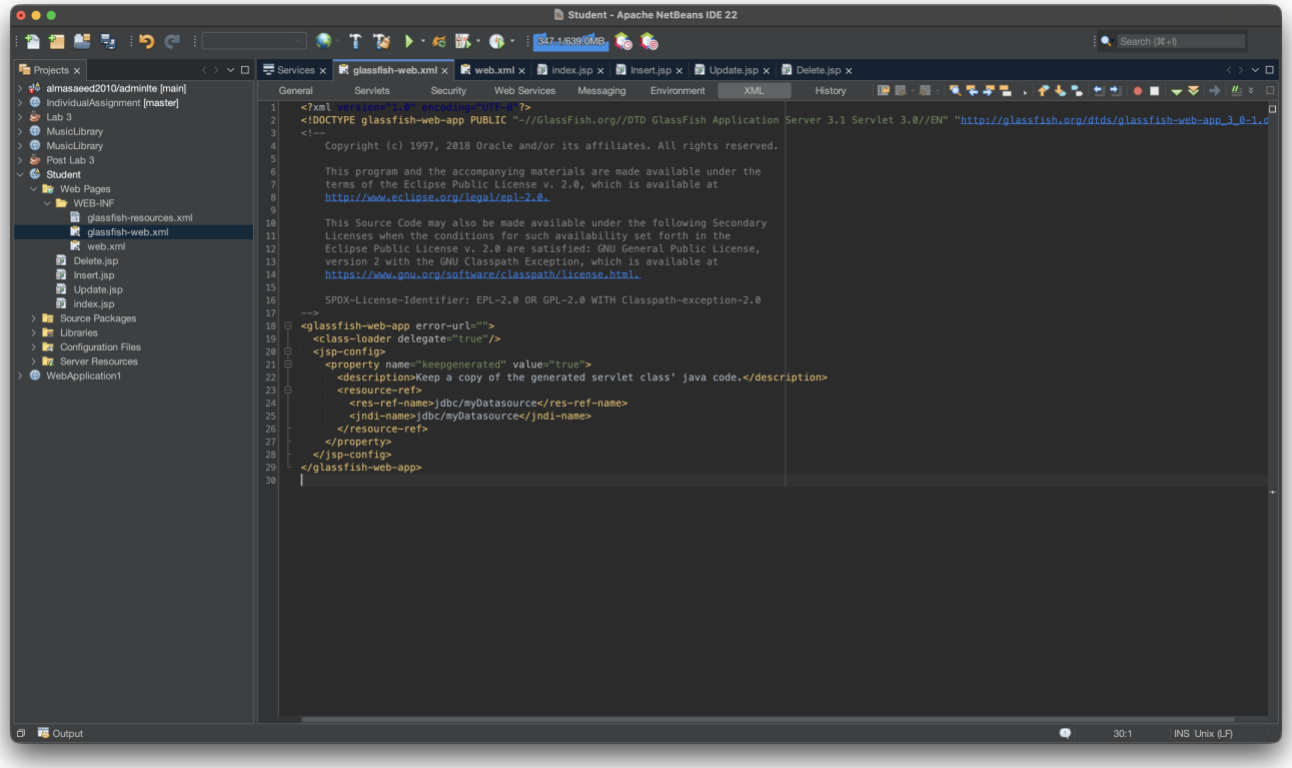


e) Give resource name in the textbox. This is JNDI resource that we already created.

f) Open Web pages -> WEB-INF -> glassfish-web.xml. Between </jsp-config> and </glassfish-webapp> tag enter following lines.

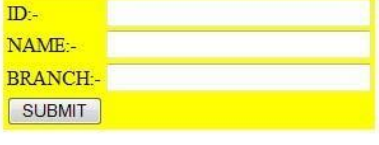
```
<resource-ref>
<res-ref-name>jdbc/myDatasource</res-ref-name>
<jndi-name>jdbc/myDatasource</jndi-name>
</resource-ref>
```

It will create reference name correspond to JNDI name.



Exercise 4: Creating Insert Page

a) Open Insert page, if not already open. Inside body part create one form. Put action of this form as insert.jsp. Inside that form tag create tax boxes and submit button. Put all things in such a way so that final view will look like as follows.



b) Above <form> tag, put three c:set tags to get value of id, name and text.

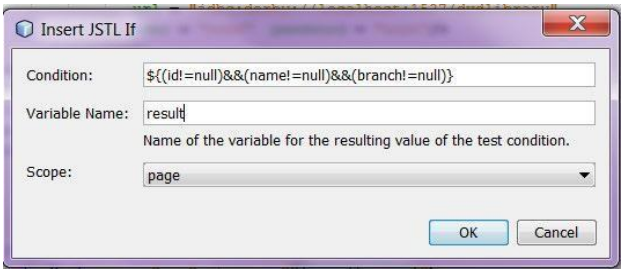
```
<c:set var="id" value="${param.id}"/> <c:set
var="name" value="${param.name}"/> <c:set
var="branch" value="${param.branch}"/>
```

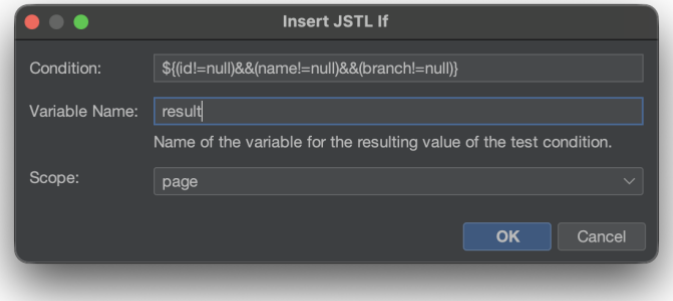
c) For using standard tag library, you have to include following three lines at top in JSP page.

```
<%@taglib prefix="sql" uri="http://java.sun.com/jsp/jstl/sql"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<sql:setDataSource var="myDatasource"
driver="org.apache.derby.jdbc.ClientDataSource"
url="jdbc:derby://localhost:1527/StudentDB" user="app"
password="app"/>
```

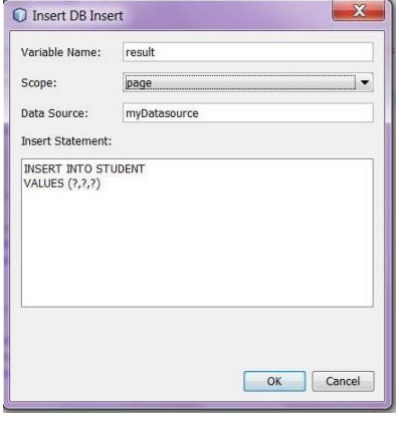
d) After this c:set statements insert one JSTL IF tag. To do this, go to palette window, under that go to JSP tag, drag and drop JSTL If from there to application after last c:set tag. It will show you following window.





e) Write condition as shown in fig. If this condition is true then and only then you want to insert data into database. So all the code inside c:if tage will be executed if and only if condition is true.

f) Inside c:if tag write code for insert query. To do this, go to palette window, under that go to Database tag, drag and drop DB Insert from there to application inside c:if tag. It will show you following window. In which, you have to insert data source name that you have created and insert query as shown in fig.



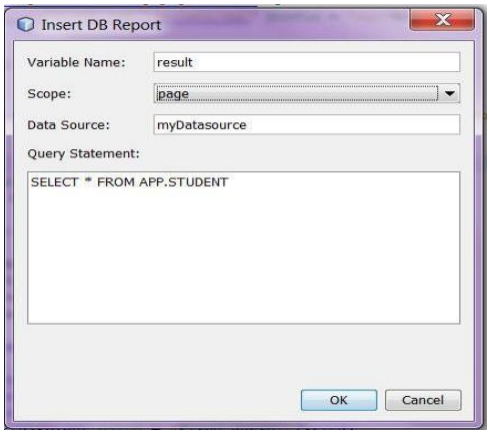
g) You need to insert parameter value that you set using c:set. So after INSERT statement, you place three sql:param tags to set value of question mark. Add following three statements after INSERT statement but inside sql:update tag.

```
<sql:param value="${id}"/> <sql:param
value="${name}"/>          <sql:param
value="${branch}"/>
```

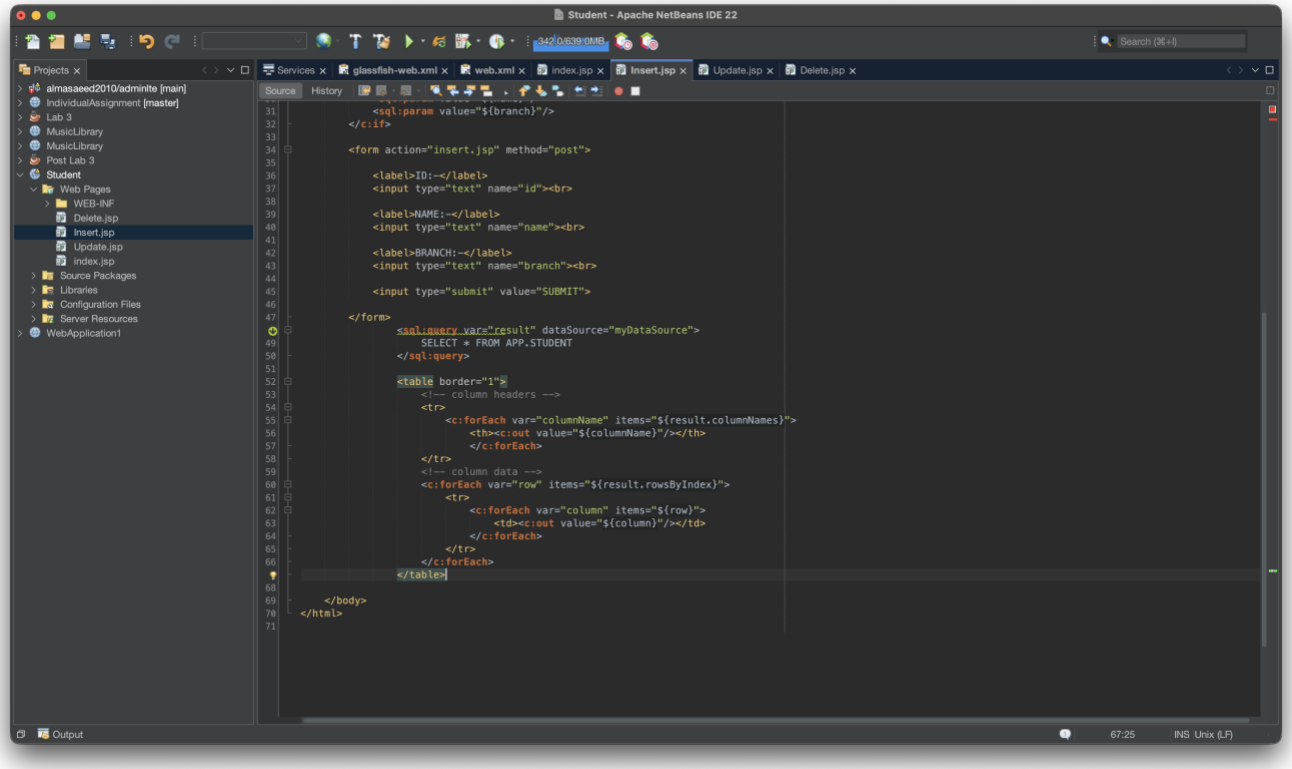
h) These statements set value of id in place of first question mark, value of name in place of second question mark and so on. Now, code to insert data into database table is over.

D

i) You need to print STUDENT table detail. To generate report, go to palette window, under that go to Database tag, drag and drop DB Report from there to application after </form> tag. Here, insert query, data source name and variable as shown in fig and click ok.



When you built and run application for first time it will show empty html table with only column names in report. Because initially, database table is empty. After you insert data into database, report show data into html table.



Exercise 5: Creating Update Page

In the Update page, you cannot update id (primary key). You need to choose any id and enter updated value for name and branch for that. All things are same as Insert.jsp page except followings:

a) After clicking submit button, you run UPDATE query instead of INSERT query.

b) You have to change action of form to update.jsp

c) Instead of using text box for id, you need to use combo box. Code for that is as shown below.

```
<select name="id">
<sql:query var="result" dataSource="${myDatasource}">
SELECT ID FROM STUDENT
</sql:query>
<c:forEach var="row" items="${result.rowsByIndex}">
<c:forEach var="column" items="${row}">
<option> <c:out value="${column}"/></option>
</c:forEach>
</c:forEach>
</select>
```

Instead of insert query you need to put update query as shown below.

```
<sql:update var="res" dataSource="${myDatasource}">  UPDATE
STUDENT SET NAME = ?, BRANCH = ? WHERE ID = ?
<sql:param value="${name}"/>
<sql:param value="${branch}"/> <sql:param
value="${id}"/>
</sql:update>
```

```
<c:set var="id" value="${param.id}"/>
<c:set var="name" value="${param.name}"/>
<c:set var="branch" value="${param.branch}"/>

<c:if test="${(id!=null)&&(name!=null)&&(branch!=null)}" var="result">
    <sql:update var="res" dataSource="${myDatasource}">
       UPDATE STUDENT SET NAME = ?, BRANCH = ? WHERE ID = ?
    <sql:param value="${name}"/>
    <sql:param value="${branch}"/>
    <sql:param value="${id}"/>
    </sql:update>  x
</c:if>

<form action="Update.jsp" method="post">

    <label>ID:-</label>
    <<select name="id">
    <sql:query var="result" dataSource="${myDatasource}">
    SELECT ID FROM STUDENT
    </sql:query>
    <c:forEach var="row" items="${result.rowsByIndex}">
    <c:forEach var="column" items="${row}">
    <option> <c:out value="${column}"/></option>
    </c:forEach>
    </c:forEach>
    </select>

    <label>NAME:-</label>
    <input type="text" name="name"><br>

    <label>BRANCH:-</label>
    <input type="text" name="branch"><br>

    <input type="submit" value="SUBMIT">

</form>
```

Exercise 6: Creating Delete Page

a) All things are same as update page except following.
b) You have to change action of form tag to Delete.jsp
c) Remove text box and label for name and branch. Remove c:set for name and branch.Name and branch are also removed from the code.
d) Add one more option name as select in combo box for id. Now, if id is null or select then you have to do nothing. Otherwise you have to run DELETE query as shown below.

```
<c:set var="id" value="${param.id}"/>
<c:if test="${(id != null) && (id != 'select') }">  <sql:update
var="res" dataSource="${myDatasource}">
DELETE FROM STUDENT WHERE ID = ?
<sql:param value="${id}"/>
</sql:update>
</c:if>
```