Course : Diploma in Multimedia & Infocomm Technology (EGDF15)

Module : Java Enterprise Development (EG3752)

|  |  |
| --- | --- |
| Laboratory : | Lab 4 – Data Retrieval from the Database via JDBC |
| Objectives : | At the end of this session, you should know how to:   * Make a connection to a database using MySQL Type 4 driver * Create a database connection pool and resource in NetBeans * Use the JDBC API to retrieve data from the database * Use PreparedStatement to guard the software against possible SQL injection attacks |
| Software Used : | Java Standard Development Toolkit (JDK™) 8.0  NetBeans IDE 8.0 with GlassFish Server 4.0 bundle  MySQL Community Edition 5.7 or WAMP Server + MySQL Workbench 6.3 |

**Preparatory Task: Working with MySQL Database**

**Installing MySQL Database and MySQL Workbench into Your Notebook**

P.1.1 Go to <http://dev.mysql.com/downloads/mysql/> to download the latest version of MySQL Community Server. The community server comprise of various MySQL tools, including the MySQL Server and MySQL Workbench.

IMPORTANT NOTE: You should download the msi installer unless you are prepared to make you own configurations.

P.1.2 You will be prompted to provide the username and password for the “super administrator” role of the database during installation. Leave the username as **root** and note down the password that you have used for this account.

|  |
| --- |
| **NOTE: If you have a WAMP Server in your setup**  You can also choose to install only MySQL Workbench and connecting the Workbench to the MySQL Server provided in the WAMP Package into Your Notebook. The latest version of the MySQL Workbench can be downloaded from this link:  <http://dev.mysql.com/downloads/workbench/>  You will also need to set the password for the root account of the MySQL Server. |

**Populating Dataset into MySQL**

In this part of the task, we will be setting up a new schema, a new table and populate it with data using MySQL database.

P.2.1 Open **MySQL Workbench 6** andclick on the center of the *MySQL* tile to connect to the database.

P.2.2 Download **EG3752BookDB.sql** script from Blackboard.

P.2.3 Open the script with notepad and replace all instances of **‘jed’** with **‘jed** *{your admin number}***’**. Note that the schema name should not contain any whitespaces.

P.2.4 Save the script and open it using MySQL Workbench

P.2.5 Run the script by clicking on the  icon

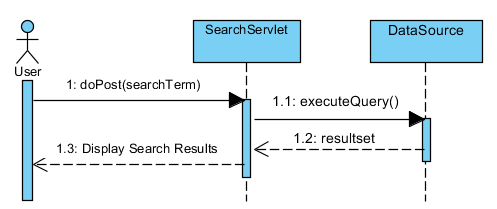
P.2.6 Ensure that a new schema named **‘jed** *{your admin number}***’** is created and a table named **book** is created under the new schema. Ensure that there is data in the **book** table.

**Library Book Management System**

In this lab, we will be creating a library book management system for the school library. The requirement specifications for the project are listed as follows:

|  |  |
| --- | --- |
| **Requirement Reference** | **Requirement Specifications** |
| 1.1 Search | The system shall provide the functionality for users to search books available in the library using part of the title. |
| 1.2 Search Result Display | The system shall display all the details of the book. |

Below is a sequence diagram showing how the different components of the web application will be interacting with each other:



Creating the Project

1.1 Create a web application named **BookManagement**.

1.2 Create a menu to allow books to be searched and added to the Book Management System by replace the codes in index.html with **Code Snippet Lab 4 P 1.2**.

1.3 Add a search page to the project named **search.html** and replace the codes in there with **Code Snippet Lab 4 P 1.3**. Note that a simple “required” data constraint has already been implemented in **search.html**.

1.4 Compile, deploy and test your application to ensure that the search link is working before proceeding.

Connecting to the Database using NetBeans

It is possible to allow NetBeans to see the database connections. In this case, we are creating the connection so that we may reference to this connection when we create the JDBC resource and the connection pool later on.

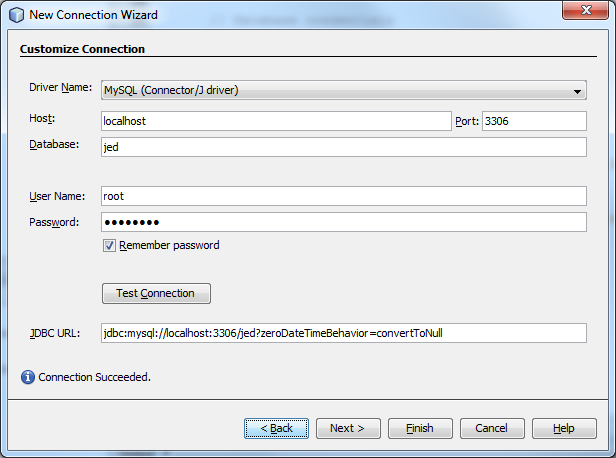
2.1 Go to the **Services** tab and right click on **Database**.

2.2 Select **New Connection**.

2.3 In the **New Connection Wizard** dialog box, choose **MySQL (Connector/J driver)** from the **Driver** dropdown box.

2.4 Click **Next**.

2.5 Under the Database field, input the name of the schema which you have created in the previous preparatory task (ie, **jedxxxxxx**):

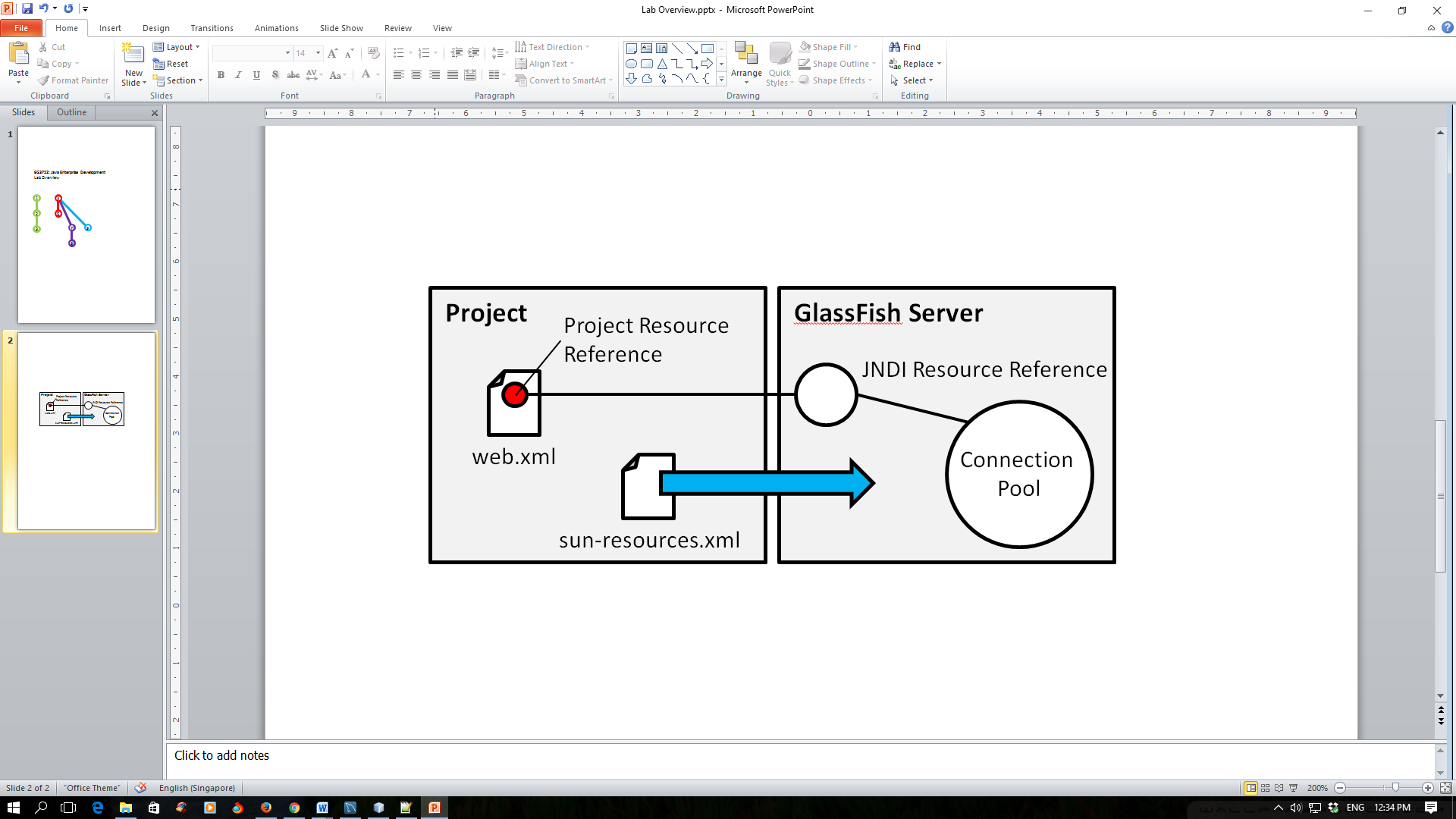


2.6 Test your connection by clicking on the **Test Connection** button. Ensure that the message **Connection Succeeded** is shown before moving on.

2.7 Click **Finish**.

Steps to create a JDBC Resource Name and Connection Pool within GlassFish Server

In order for the web application to use the database connection from a connection pool, create a **connection pool** and **a resource reference** to the connection pool within the GlassFish Server. Finally, a second resource reference is created within the project in **web.xml** to the resource reference configured in GlassFish Server.



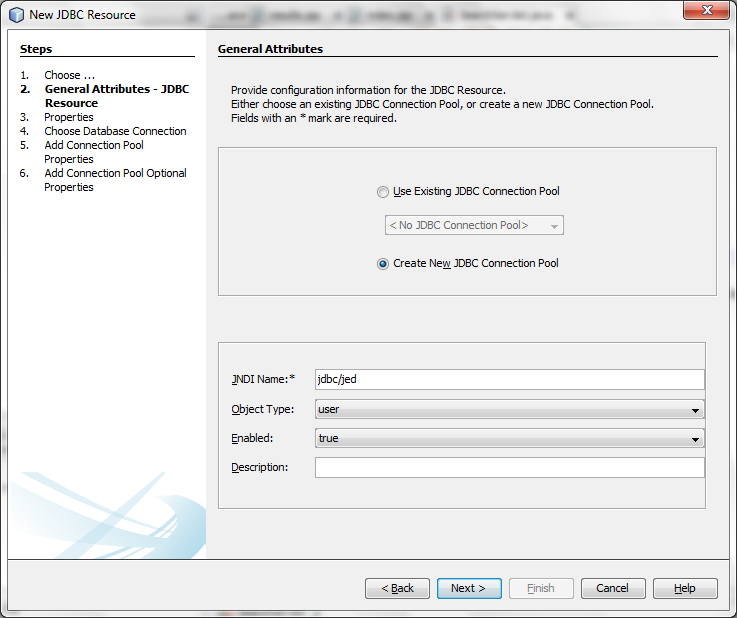
In this part of the task, we will be creating both the resource name and the connection pool at the same time.

3.1 Right click on your project node and choose **New > Other…** .

3.2 Select **JDBC Resource** from the **GlassFish** category.

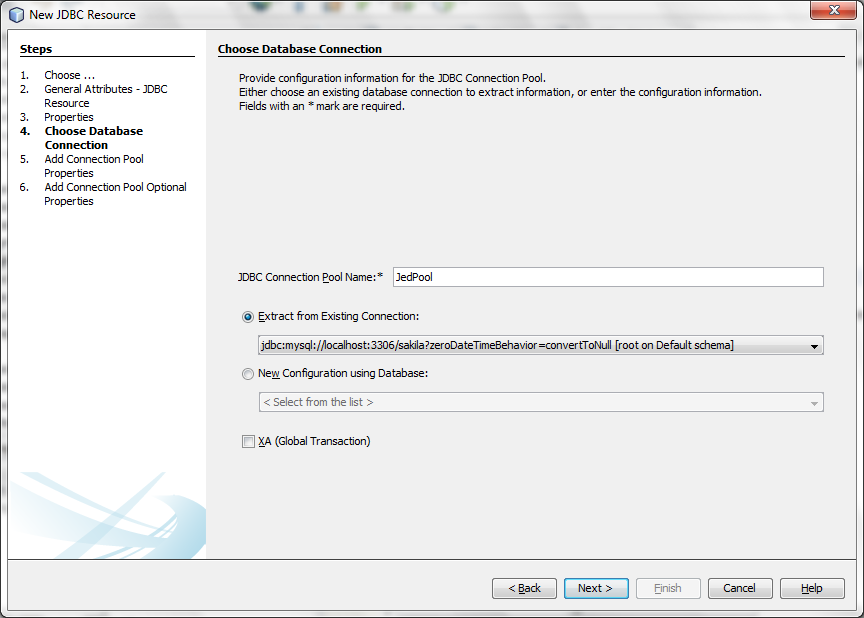
3.3 Click **Next**.

3.4 Select **Create New JDBC Connection Pool** if you have not created a connection pool before. Under the **JNDI Name** field, give your resource a name. By industry practice, JDBC resource names usually starts with “jdbc/”.



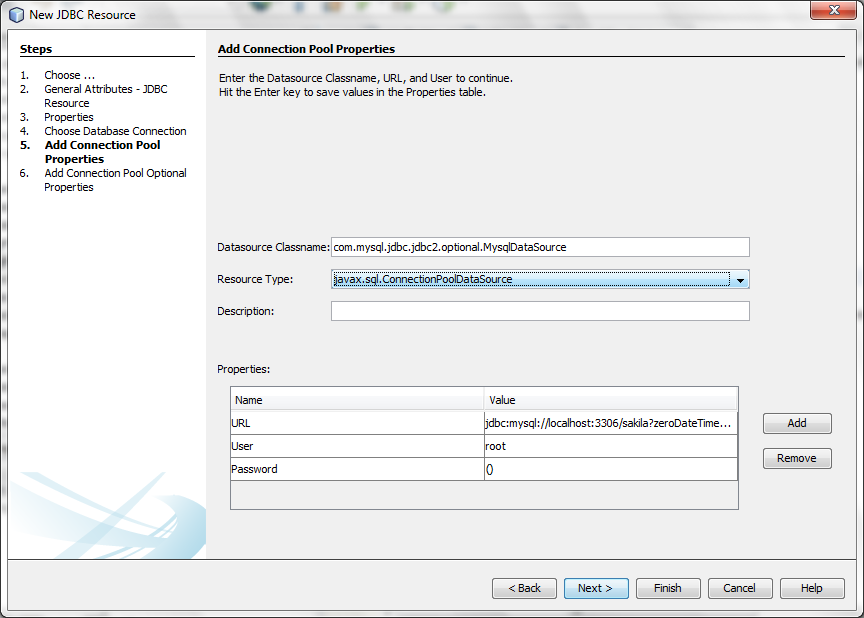
3.5 Click **Next**.

3.6 Name your JDBC Connection Pool. Select **Extract from Existing Connection** and select the connection you have just created in Steps 2.1 to 2.7.



3.7 Click **Next.**

3.8 Select **javax.sql.ConnectionPoolDataSource** as the **Resource Type**.



3.9 Click **Next**.

3.10 Click **Finish**.

Notice that NetBeans has also created a new configuration file, **sun-resources.xml**, within your project folder, found under the **Server Resources** folder. This file is essential to allow your application to use the database connection pool that you have just setup.

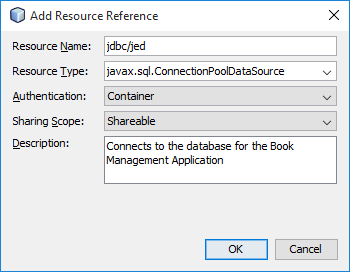
Creating a Resource Reference in the Project to the Resource Reference in GlassFish Server

4.1 Open up the **web.xml** file found under the **Configuration Files** folder in your **Project** explorer. If the file does not exist, create a new one by right clicking on your project node, and choose **New > Other…**. This can be found under the **Web** category.

4.2 Select **Reference** in the tab above.

4.3 Expand the **Resource References** heading, then click **Add**. The **Add Resource Reference** dialog box should open.

4.4 Configure the resource reference as follows:



4.5 Click **OK**.

Retrieving Data from the Database

To complete this part of the lab, you may find it useful to cross reference with your lecture notes on how JDBC is written.

* 1. Create a **SearchServlet** for your project. Look at the codes in **search.html** to decide the appropriate URL Pattern to use and the correct service method (**get** or **post**) to override.
  2. Inject a **DataSource** object as a class wide variable into the servlet. Be sure to import that DataSource object from the **javax.sql** library. The codes required to inject the DataSource are as follows:

@Resource**(**name**=**"jdbc/jed"**)**

private DataSource dsBookCatalogue**;**

5.3 Override the **doGet** or **doPost** method and get a connection from the data source within the method:

Connection connection **=** dsBookCatalogue**.**getConnection**();**

5.4 Create a new **Statement** object from the **Connection** object:

Statement statement **=** connection**.**createStatement**();**

5.5 Retrieve all books with titles containing the search term from the **ResultSet** object:

ResultSet resultset = statement.executeQuery(/\*Your SQL statement goes here\*/);

5.6 Retrieve all the results from the ResultSet and set them into an appropriate data structure:

1. Create a Java class to hold information about a single book. For simplicity, each field in the database table can be mapped to a single member variable in the class to be created. Complete the following class diagram to help you:

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Datatype** |  | **Book** |
| isbn | VARCHAR(15) |  | - isbn:String  - title:  - author:  -  -  - |
| title | VARCHAR(256) |  |
| author | VARCHAR(256) |  |
| year | YEAR |  |
| publisher | VARCHAR(256) |  |
| about | LONGTEXT |  |
| **Book** table in the Database | | 🡪 | **Book** class to be created |

b. To store multiple records, you will find it useful to create a java.util.List of the class such as what you have done in the previous lab.

c. Different records in the results can be assessed by cycling through the **RecordSet** object. This can be achieved using the while loop. An example of what the pseudocode (and some codes) may look like is given below

while(resultset.next())

{

//Create a book object

//Retrieve the data from the recordset and

//store it into a book object.

String isbn = resultset.getString(“isbn”);

book.setIsbn(isbn);

//Store the book object into the list

}

5.7 Close the **ResultSet** object, **Statement** object and followed by the **Connection** object, in that order within the finally clause. Remember to check for null before closing the objects.

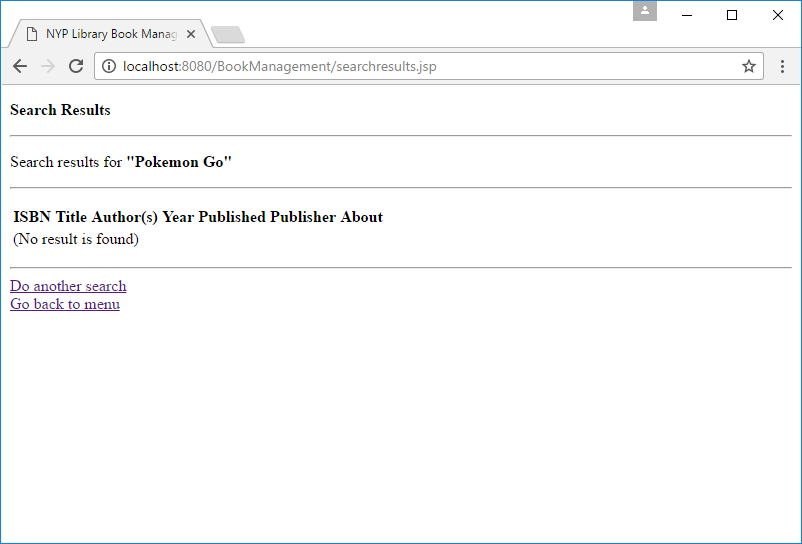
5.8 Save the search term and the search results into the session.

5.9 Make a client-side redirect to **searchresult.jsp**

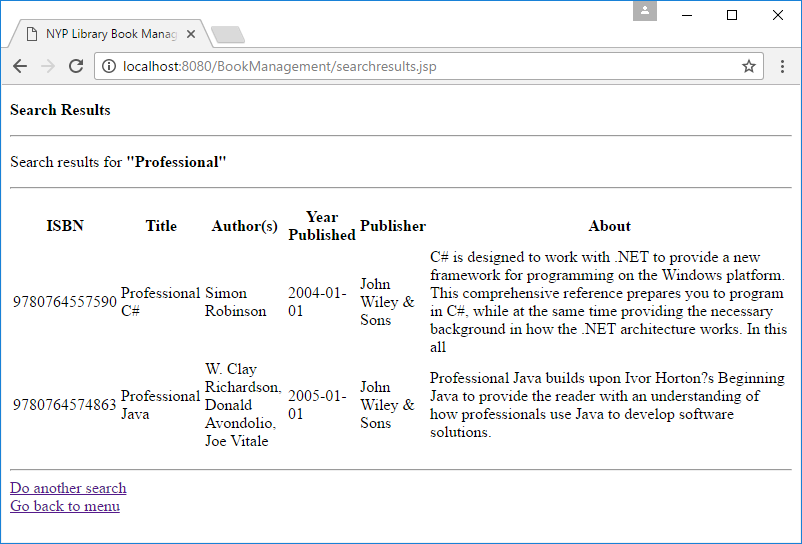
Displaying the Results

6.1 Create a **searchresults.jsp** in the project and replace the codes with **Code Snippet Lab 4 P 6.1**.

6.2 Using the comments in the codes as a guide, implement **searchresults.jsp** such that it will be able to display the result as follows:



Screenshot 1: How searchresults.jsp will look like if no results are returned



Screenshot 2: How searchresults.jsp will look like if some results are retrieved

Hardening the Codes Against SQL Injection

Discussion: If you have done your JDBC codes based on what is recommended, there is still a possibility to hack into your search programme. We can do this via **SQL Injection**. Can you suggest some ways to guard against SQL Injection?

|  |
| --- |
|  |
|  |

7.1 Enhance the security of your programme by using the methods suggested to secure your codes from hacking.

*Hint: Look at the objectives of the lab again.*