Project 5: Report

TeamNumber: 127 Name: Linyue Chu #80563910

**Task 1**

* **How did you use connection pooling?**

1. Firstly, modify the **web.xml** in WEB-INF: add resource reference name, and resource type.
2. Then, modify the **context.xml** in META-INF: check if there exists a resource element (its name, type, username, password), in the former projects I have done these before, it is already there.
3. Thirdly, obtain environment naming service, use context.loopup() to retrieve the name of the object, in this case, it is “java:comp/env” (Since for each servlet, the operation is the same so I only snapped the according part in SearchAutoComplete.java).
4. Finally, look up from the data source, get the connection from data source.

(The Figure 1—3 shows the code snippet.)

* **File name, line numbers as in Github**

|  |  |
| --- | --- |
| File Name | Line Numbers |
| project2/WeContent/WEB-INF/web.xml | 12-17 |
| project2/WeContent/META-INF/context.xml | 3-13 |
| project2/src/(default package)/SearchAutoComplete.java | 70-83 |
| project2/src/(default package)/SearchFullText.java | 54-75 |
| project2/src/(default package)/SearchingServlet.java | 64-83 |
| project2/src/(default package)/ShowMetaDataServlet.java | 66-79 |
| project2/src/(default package)/SingleMovieServlet.java | 45-58 |
| project2/src/(default package)/SingleStarServlet.java | 45-58 |
| project2/src/(default package)/MovieServlet.java | 44-57 |

* **Snapshots showing use in your code**

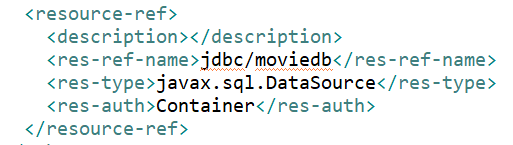


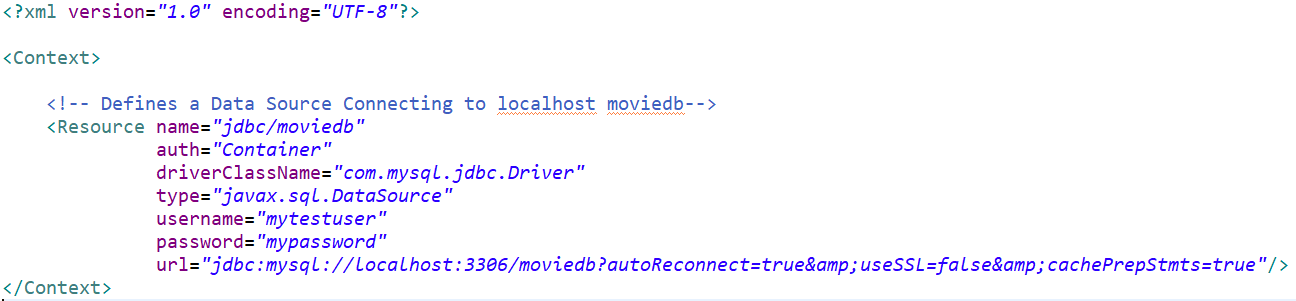
Figure 1. the configuration in web.xml  


Figure 2. the configuration in context.xml

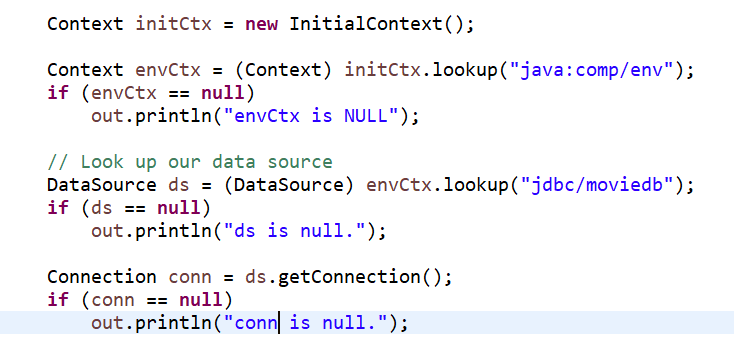


Figure 3. the code snippet in SearchAutoComplete.java

* **How did you use Prepared Statements?**

1. Firstly, modify context.xml in META-INF: set the cachePrepStms to true.
2. Then, create a PreparedStatement Object.
3. Thirdly, supply values to PrepardStatement parameters
4. Finally, executing the PreparedStatement Objects.

* **File name, line numbers as in Github**

|  |  |
| --- | --- |
| File Name | Line Numbers |
| project2/WeContent/META-INF/context.xml | 12 |
| project2/src/(default package)/SearchFullText.java | 82-112 |

* **Snapshots showing use in your code**

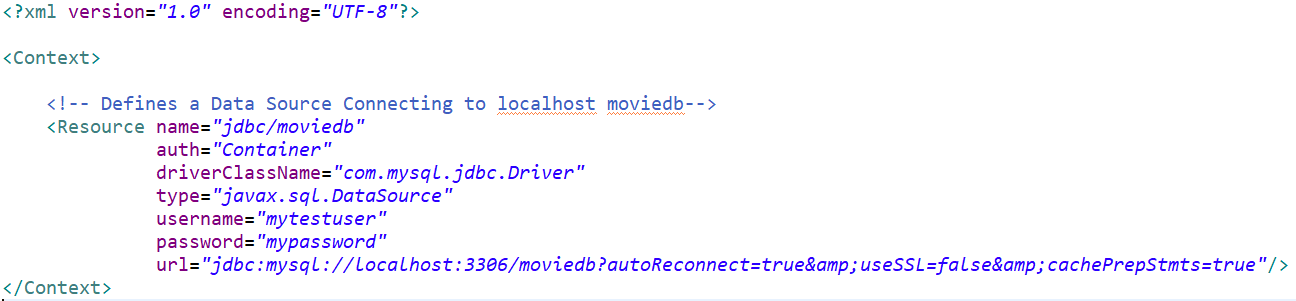


Figure 4.the configuration in context.xml

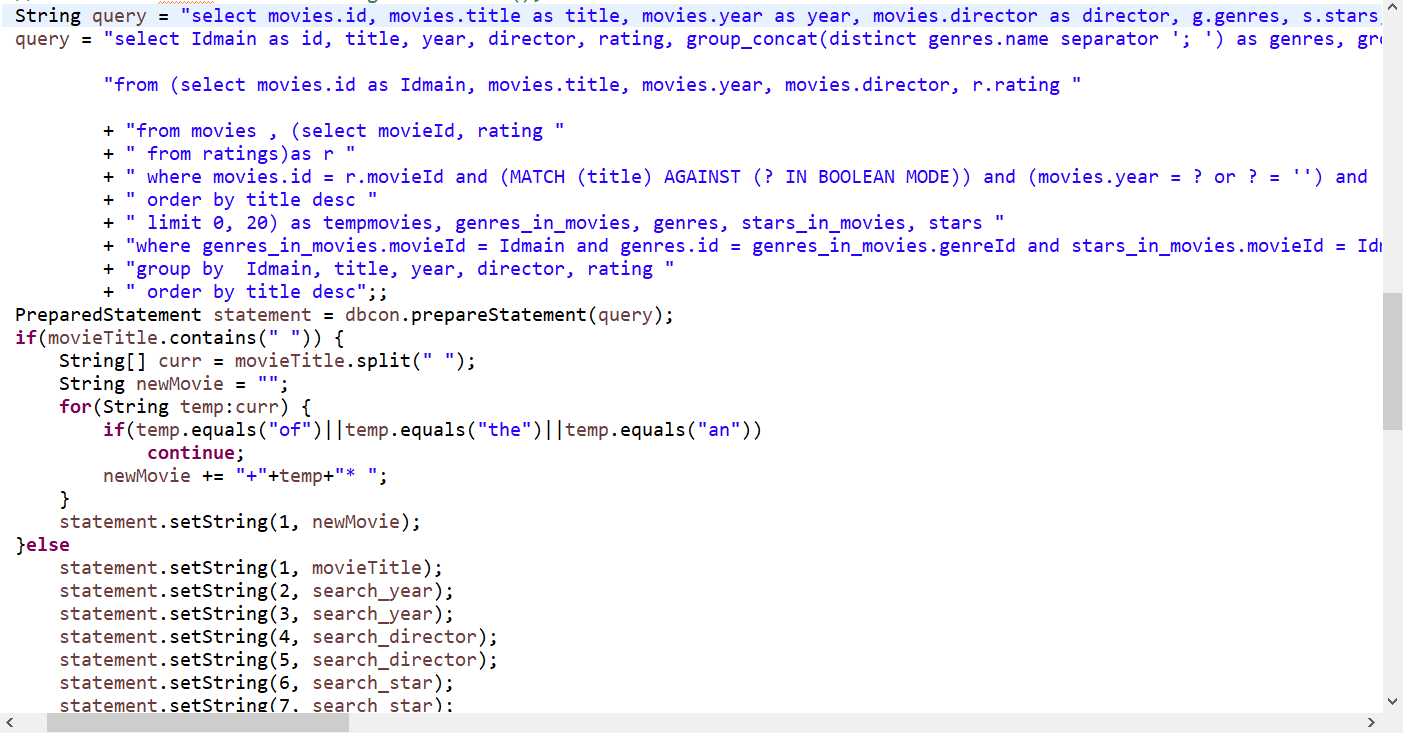


Figure 5. the code snippets of Prepared Statement in SearchFullText.java

**Task 2**

* **Address of AWS and Google instances**

Google instance IP: 35.243.132.79

AWS: instance1(original) IP: 18.191.153.29

master (2): 18.217.44.186

slave (3): 18.224.199.204

* **Have you verified that they are accessible? Does Fablix site get opened both on Google’s 80 port and AWS’ 8080 port?**
  1. Have verified that they are accessible.
  2. The Fablix site get opened both on Google’s 80 port and AWS’ 8080 port. You need to input the following url to get the service.

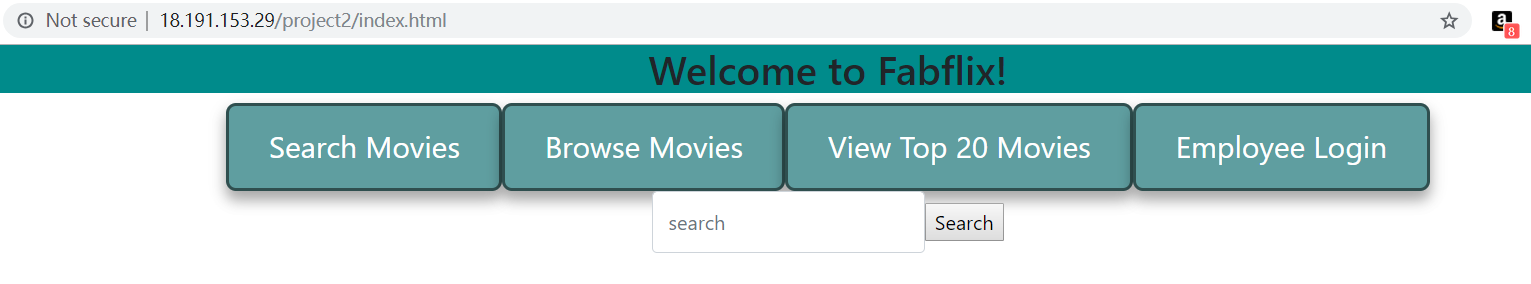


Figure 6. AWS instance

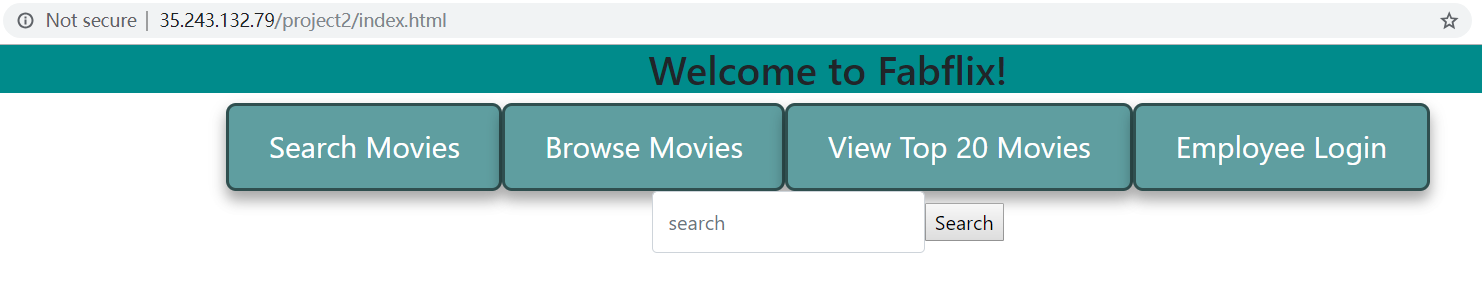


Figure 7. Google Cloud

* **Explain how connection pooling works with two backend SQL (in your code)?**

1. Modify the 000-default.conf file, in particular, the setting for load balancer.
2. In ProxySet, enable the sticky session.
3. In Proxy, name it project2\_balancer, and use the IP of instance2(master) and instance3(slave) to route. And set the sticky session id.
4. Also set the ProxyPass and ProxyPassReverse accordingly.

* **File name, line numbers as in Github**

|  |  |
| --- | --- |
| File Name | Line Numbers |
| project2/WeContent/WEB-INF/web.xml | 12-17 |
| project2/WeContent/META-INF/context.xml | 3-13 |
| project2/src/(default package)/SearchAutoComplete.java | 70-83 |
| project2/src/(default package)/SearchFullText.java | 54-75 |
| project2/src/(default package)/SearchingServlet.java | 64-83 |
| project2/src/(default package)/ShowMetaDataServlet.java | 66-79 |
| project2/src/(default package)/SingleMovieServlet.java | 45-58 |
| project2/src/(default package)/SingleStarServlet.java | 45-58 |
| project2/src/(default package)/MovieServlet.java | 44-57 |

* **Snapshots**



Figure 8. the 000-default.conf in AWS

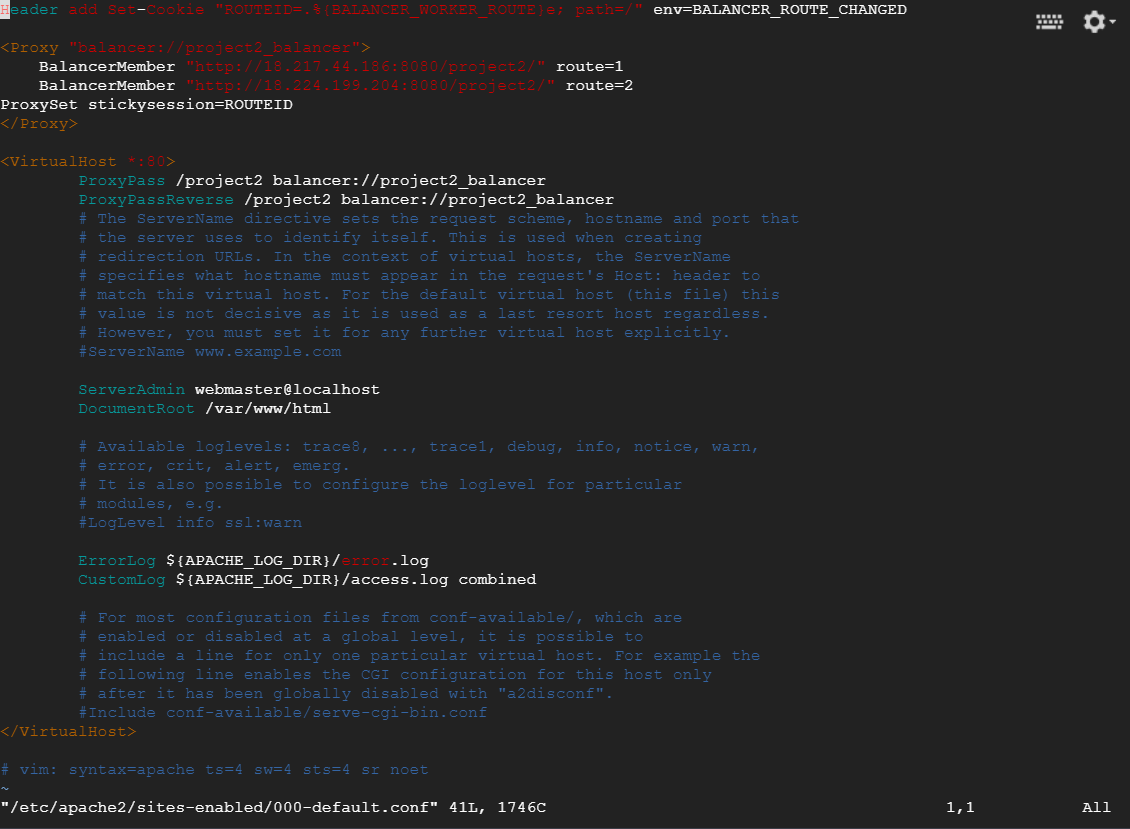


Figure 9. the 000-default.conf in Google Cloud

Since we cannot upload the 000-default.conf to github, the snapshot is given above. The connection pooling is given in the task 1.

* **How read/write requests were routed?**

The operation of scaling Fabflix(project2) requires a load balancer to distribute the request from user to two-backend instances. The two mysql databases in the two instances has master-slave relationship between them, in which the write operation in the Master mysql will propagate and repeat in the Slave mysql through a log file. The Slave will not propagate the writing operation to the Master. Write in Slave will break the synchronization between Master and Slave.

The extra connection pooling resource provide a direct connection for both instances to connect to the Master mysql database. When there is a writing operation in Slave’s servlet, the servlet will directly connect with Master’s mysql and perform the writing.

Both instance2(Master) and instance3(Slave) has the same version of scaled Fabflix deployed, each with three connection pooling resources. The servlet will choose which connection pooling to use based on type of request.

* + **File name, line numbers as in Github**

|  |  |
| --- | --- |
| File Name | Line Numbers |
| project2/WeContent/WEB-INF/web.xml | 18-23 |
| project2/WeContent/META-INF/context.xml | 14-20 |
| project2/src/(default package)/EmployeeServlet.java | 72-85 |

* **Snapshots**

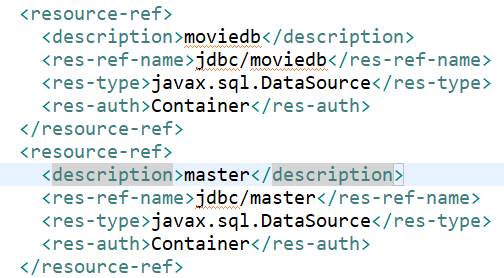


Figure 10. the snippets of web.xml  


Figure 11. the snippets of context.xml

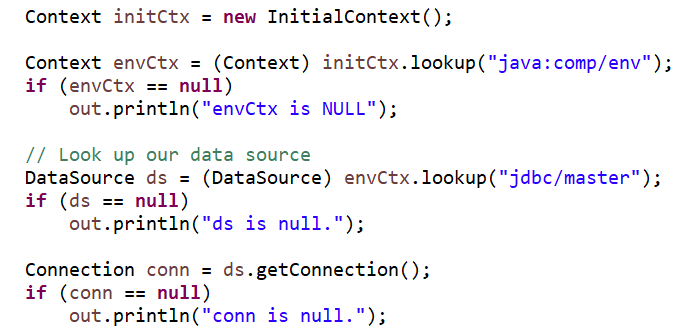


Figure 12. the snippets in EmployeeServlet.java

**Task 3**

* **Have you uploaded the log files to Github? Where is it located?**

They are saved in TestResults, which includes txt files and jmeter snaps. The txt files are the logfiles of TS and TJ. I wrote a servlet named TestPerformance.java to test the server.

* **Have you uploaded the HTML file (with all sections including analysis, written up) to Github? Where is it located?**

jmeter\_report.html, it’s in TestResults.

* **Have you uploaded the script to Github? Where is it located?**

cal.py which is also in TestResults.

* **Have you uploaded the WAR file and README to Github? Where is it located?**

project2.war

READMD.md