**Project 5**

**Task 1**

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

It involves 3 configuration files.

I need to set up a database in the context.xml and specify the total number of connections and the number of idle connections. Then we need to set the factory for connection instances to talk to the particular database, TestDB. Then in the servlet, we use resource TestDB to establish connections to enable connection pooling.

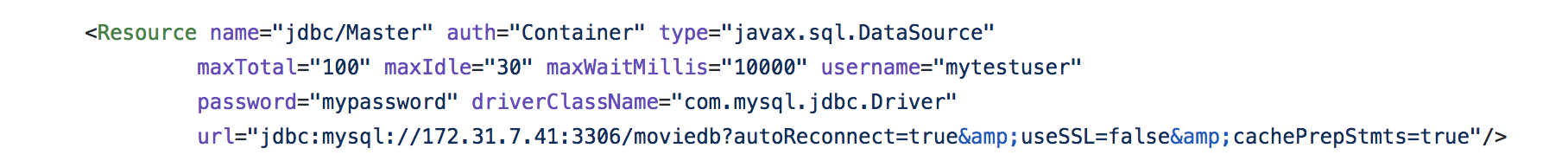
* **File name, line numbers as in Github**

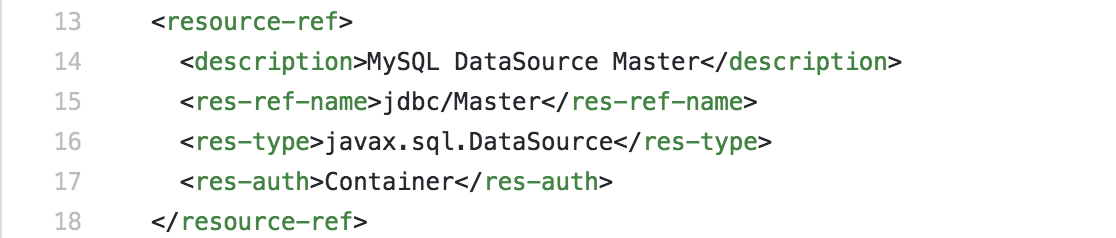
[cs122b-spring18-team-104](https://github.com/UCI-Chenli-teaching/cs122b-spring18-team-104)/[project2](https://github.com/UCI-Chenli-teaching/cs122b-spring18-team-104/tree/master/project2)/[WebContent](https://github.com/UCI-Chenli-teaching/cs122b-spring18-team-104/tree/master/project2/WebContent)/[WEB-INF](https://github.com/UCI-Chenli-teaching/cs122b-spring18-team-104/tree/master/project2/WebContent/WEB-INF)/web.xml // line 13-18

[cs122b-spring18-team-104](https://github.com/UCI-Chenli-teaching/cs122b-spring18-team-104)/[project2](https://github.com/UCI-Chenli-teaching/cs122b-spring18-team-104/tree/master/project2)/[WebContent](https://github.com/UCI-Chenli-teaching/cs122b-spring18-team-104/tree/master/project2/WebContent)/[META-INF](https://github.com/UCI-Chenli-teaching/cs122b-spring18-team-104/tree/master/project2/WebContent/META-INF)/context.xml // line 22-25

And in every servlet, I establish connection by connecting to TestDB (in my case, I call it jdbc/Master as reference to the snapshots below).

* **Snapshots showing use in your code**







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

For every Servlet, I use prepared Statements to prepare each query. In this case, only then the statement is compiled just once and can be used repeatedly with different parameters which results in better performance.

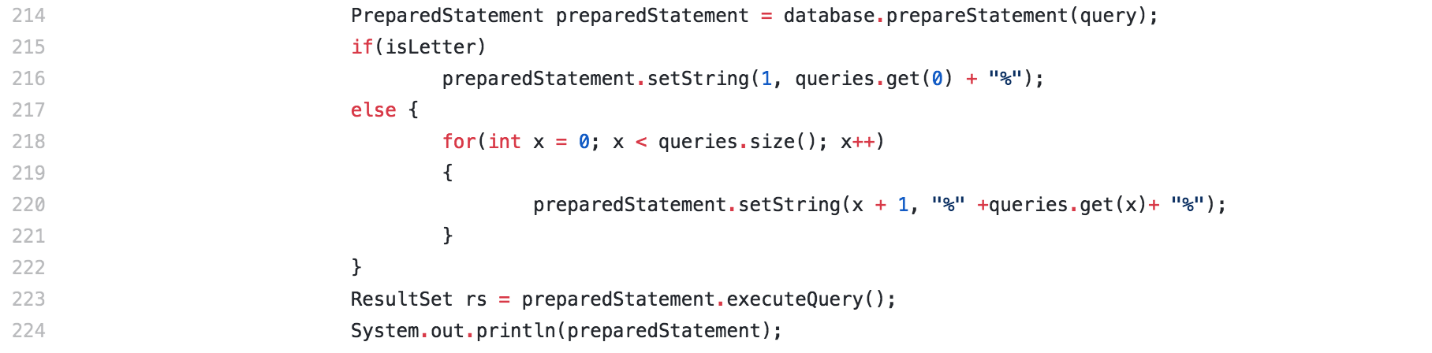
After we applying the connection pooling, we need to set the url in the context.xml to url="jdbc:mysql://172.31.7.41:3306/moviedb?autoReconnect=true&amp;useSSL=false&amp;cachePrepStmts=true"/>

Then we can make sure that prepared statements work well with the connection pooing.

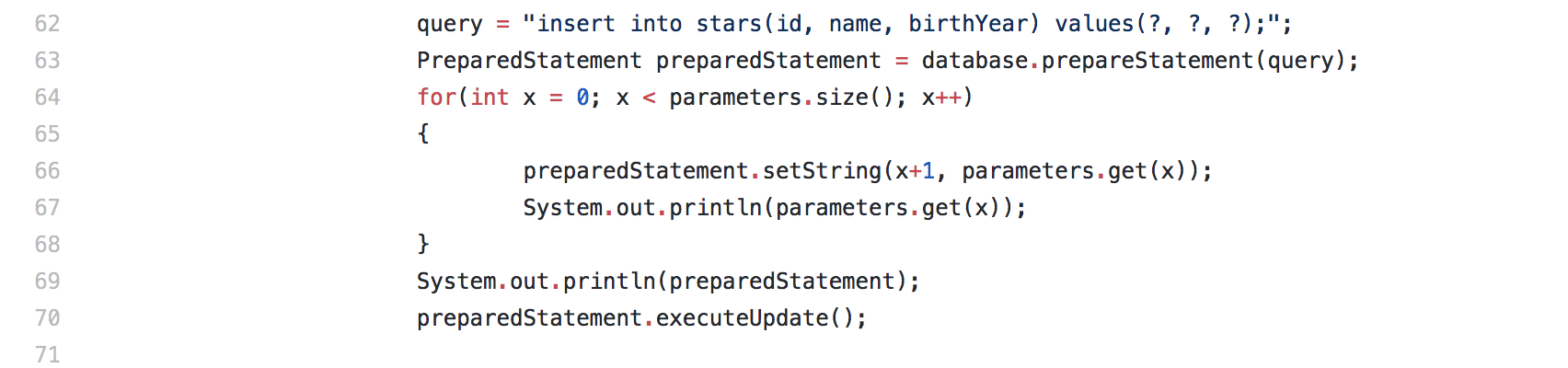
* **File name, line numbers as in Github**

In all the servlets which involve query, we used prepared statements.

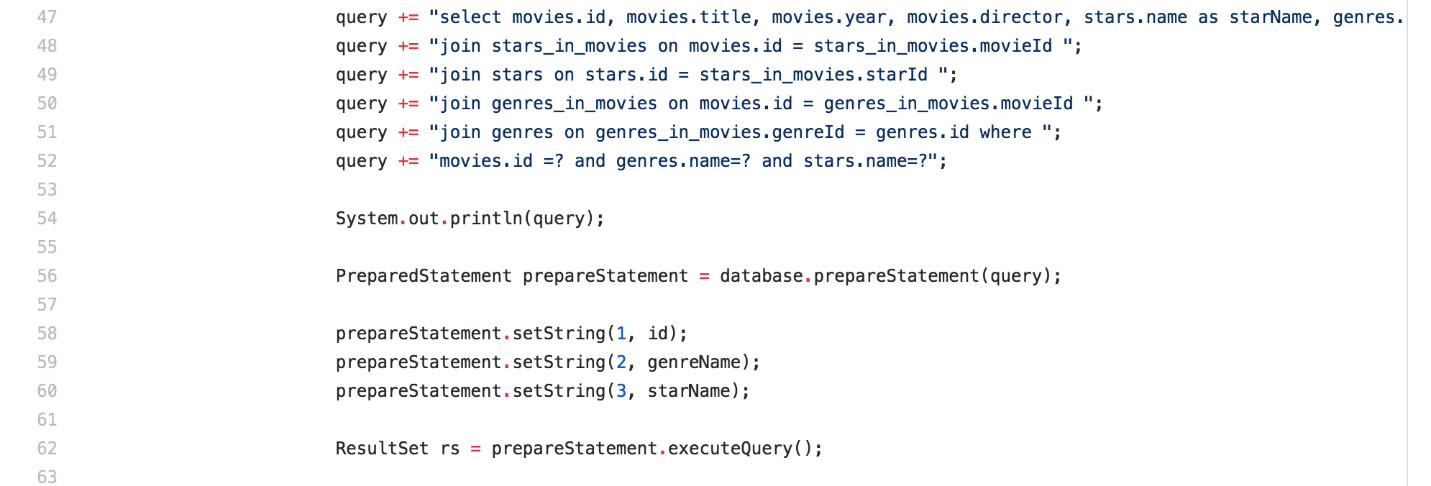
cs122b-spring18-team-104/project2/src/MovieListServlet.java



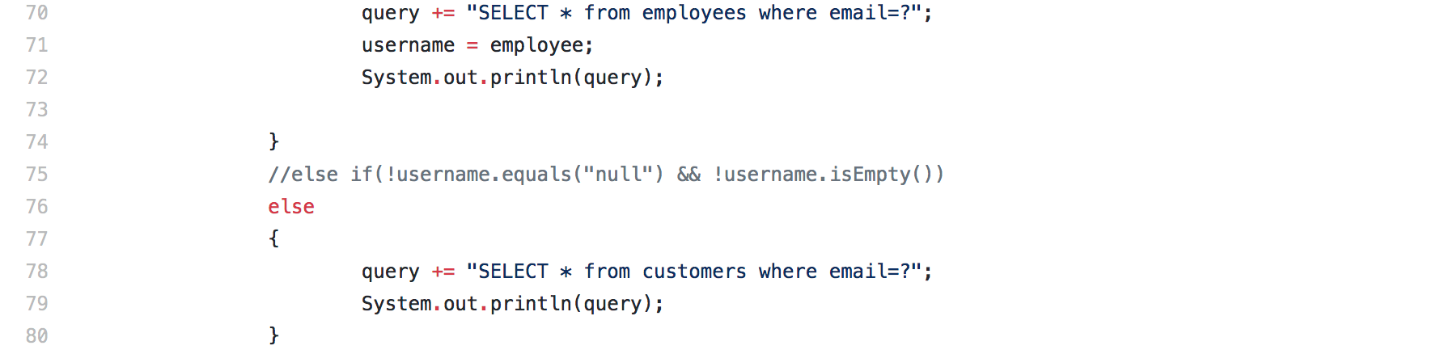
cs122b-spring18-team-104/project2/src/InsertStarServlet.java



cs122b-spring18-team-104/project2/src/EmployeeSearchServlet.java



cs122b-spring18-team-104/project2/src/LoginServlet.java



**Task 2**

* **Address of AWS and Google instances**

AWS:

Instance 1: 54.153.105.163

Master: 54.183.219.216

Slave: 54.67.73.197

Google Cloud: 35.196.57.81

* **How does load balancing work?**

I did MySQL master/slave replication. Configure the original instance properly to enable load balancing, connection pooling, sticky sessions.

For master/slave replication, I already checked with pet example provided by professor.

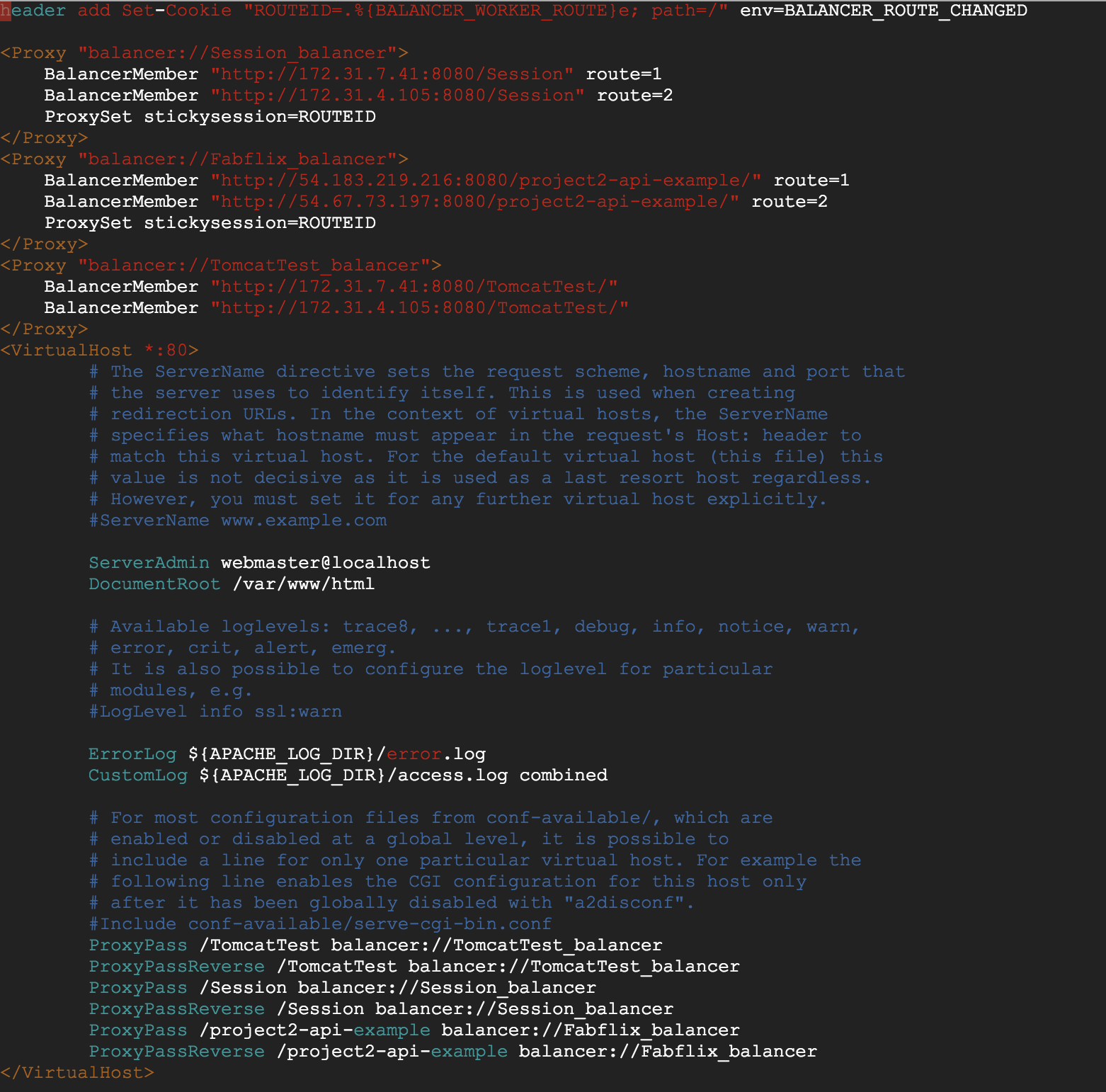
To enable load balancing, Configure the Apache2 web server to use its balancer to the url of fabflix. For the sticky session, we should add a line to the 000-default.conf:

Header add Set-Cookie "ROUTEID=.%{BALANCER\_WORKER\_ROUTE}e; path=/" env=BALANCER\_ROUTE\_CHANGED

The configuration file on AWS instance 1 is as below.



The configuration file on Google Cloud is as below, very much like the 000-default.conf on AWS. But I used public IP address of master/slave for Google Cloud instance.



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

When accessing a 'read' page. The read page servlet then connects with 'jdbc/moviedb', which sets url 'localhost'.

As localhost here is original instance public IP, the proxy will then route requests to either of the new instances, master or slave.

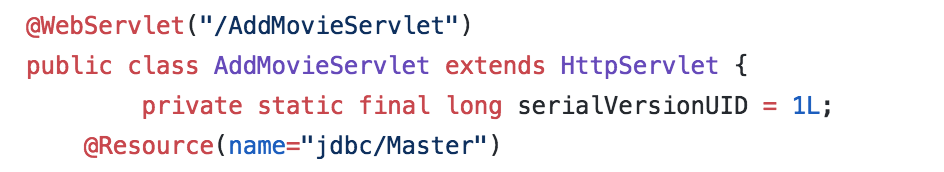
When accessing a 'write' page. The write page servlet then connects with 'jdbc/Master', which sets url to my master server public IP. Then the writing operation will be sent to the master replication.

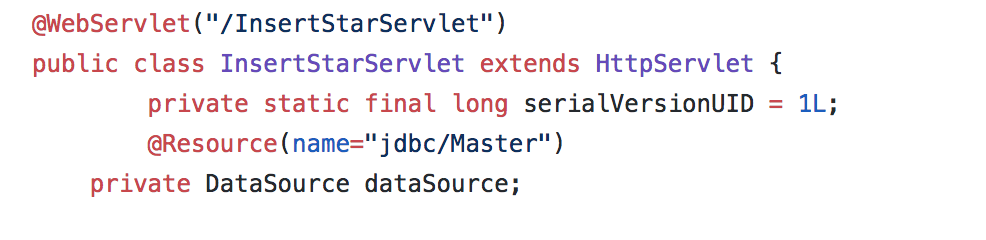
In the servlet which will do writes to the database, I will set the resource to 'jdbc/Master’. Like

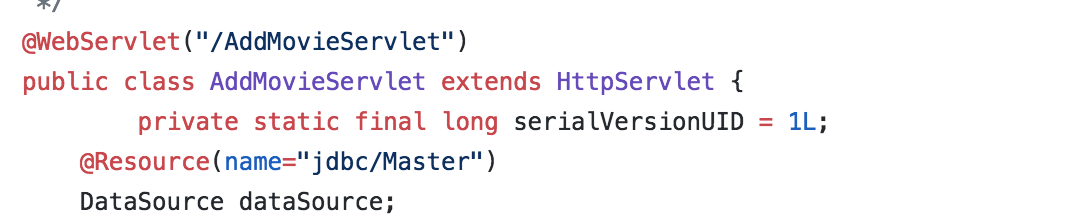
cs122b-spring18-team-104/project2/src/AddMovieServlet.java

cs122b-spring18-team-104/project2/src/AddServlet.java

cs122b-spring18-team-104/project2/src/InsertStarServlet.java

I will set the resource to jdbc/Master





**Task 3**

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

Yes, in the jmeterReport folder.

* Have you uploaded the HTML file to Github? Where is it located?
* Yes, in the jmeterReport folder.
* Have you uploaded the script  to Github? Where is it located?

Yes, in the jmeterReport folder.

* Have you uploaded the WAR file and README  to Github? Where is it located?
* Yes, in the jmeterReport folder.

README: cs122b-spring18-team-104/README.md