Connection Pool Statistics

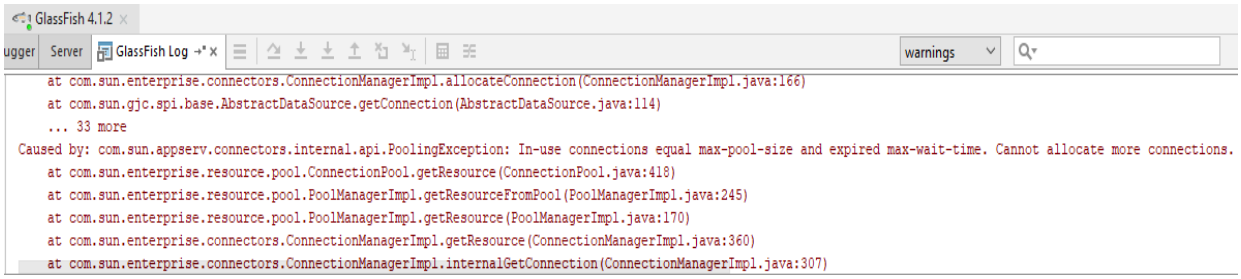
Java Technologies (Hw. 6)

For this experiment we have integrated in our glassfish sever configuration a connection pool for PostgreSQL.

On server side we created a servlet in order to fulfill requests.

On client side we have a Java application which creates a predefined number of threads. In order to have the best results we added a *CountDownLatch,* a tool that helps us send request at "exactly" same time.

1. 32 connections – 100 requests at same time – without closing connections
    a. Server side failure:



    b. Client side failure

```
    at com.company.MySecondThread.run(MySecondThread.java:30)
java.net.SocketException: Unexpected end of file from server
    at sun.net.www.http.HttpClient.parseHTTPHeader(HttpClient.java:851)
    at sun.net.www.http.HttpClient.parseHTTP(HttpClient.java:678)
    at sun.net.www.http.HttpClient.parseHTTPHeader(HttpClient.java:848)
    at sun.net.www.http.HttpClient.parseHTTP(HttpClient.java:678)
    at sun.net.www.protocol.http.HttpURLConnection.getInputStream0(HttpURLConnection.java:1587)
    at sun.net.www.protocol.http.HttpURLConnection.getInputStream(HttpURLConnection.java:1492)
    at com.company.MySecondThread.run(MySecondThread.java:30)
java.net.SocketException: Unexpected end of file from server
```

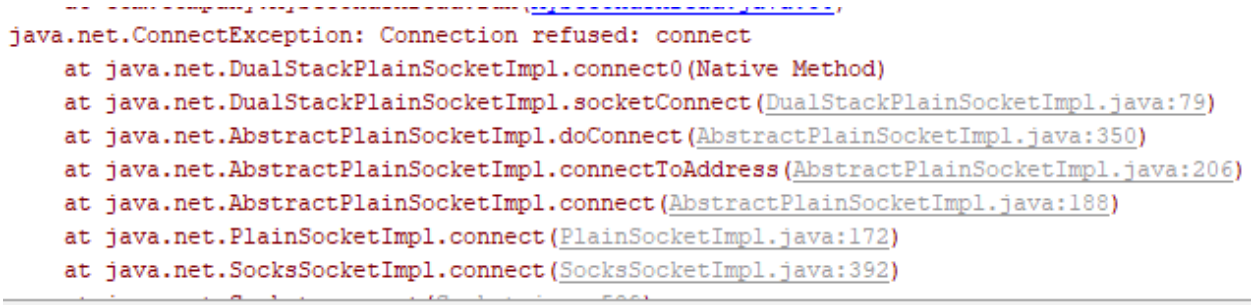
No error has been found if we increase the number of connections to exceed number of requests.

2. 32 connections – 10000 requests at same time – closing connections

No errors have been found if we close each connection after each request.

3. Singleton connection

   Using just one connection will result in connection refused exception because a servlet thread is trying to acquire the database connection which can be occupied at a given moment.



   In case of requests <1000 no failures have been detected so far but for large number of requests e.g. 10000 we have detected around 50-500 failures.