Task 1

- How did you use connection pooling?

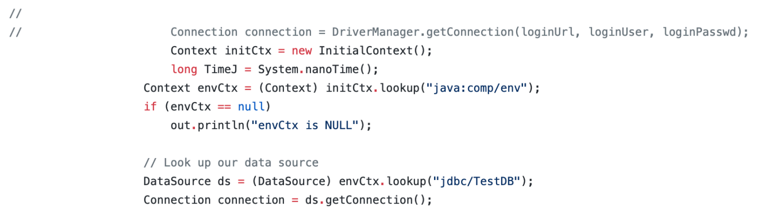
```java
Context initCtx = new InitialContext();

    Context envCtx = (Context) initCtx.lookup("java:comp/env");
    if (envCtx == null)
        out.println("envCtx is NULL");

    // Look up our data source
    DataSource ds = (DataSource) envCtx.lookup("jdbc/TestDB");
    Connection connection = ds.getConnection();
```

I firstly copy the JDBC connector's jar file into the tomcat's library. Then, by adding <Resource> in the WebContent/META-INF/context.xml and using the code above to look up specific connection resource in all servlets that need to connect to database. Specifically, I used connection pooling in my AutoServlet, Browsing Servlet, dashboard servlet, Login Servlet, Movie Servlet, singleMovie Servlet, Single Star servlet with looking up "TestDB" and used connection pooling in my add_movie, completeCheckOut with using look up "Write".

- File name, line numbers as in Github

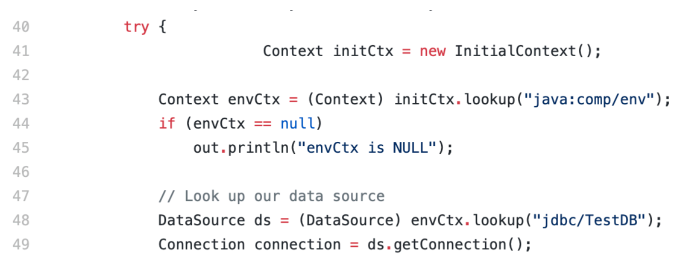
  /project5/src/MovieServlet.java 81-89
  /project5/src/AutoServlet.java 43-49
  /project5/src/BrowsingServlet.java 56-64
  /project5/src/LoginServlet.java 63-69
  /project5/src/add_movie.java 49-56
  /project5/src/completecheckoutServlet.java 68-74
  /project5/src/dashboard.java 45-51
  /project5/src/makechange.java 50-55
  /project5/src/singleMovieServlet.java 43 -50
  /project5/src/singleStarServlet.java 45-53

- Snapshots showing use in your code
  MovieServlet.java 81-89



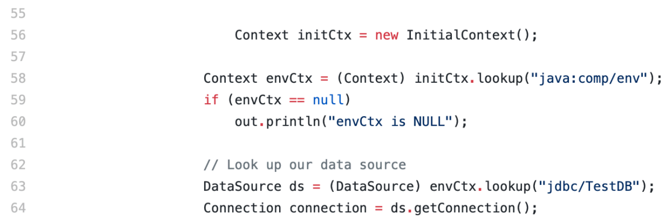
```java
//
//                      Connection connection = DriverManager.getConnection(loginUrl, loginUser, loginPasswd);
                        Context initCtx = new InitialContext();
                        long TimeJ = System.nanoTime();
                    Context envCtx = (Context) initCtx.lookup("java:comp/env");
                    if (envCtx == null)
                        out.println("envCtx is NULL");

                    // Look up our data source
                    DataSource ds = (DataSource) envCtx.lookup("jdbc/TestDB");
                    Connection connection = ds.getConnection();
```

AutoServlet.java 43-49



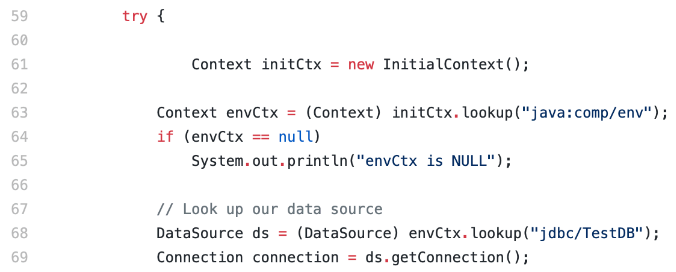
```
40          try {
41                          Context initCtx = new InitialContext();
42
43              Context envCtx = (Context) initCtx.lookup("java:comp/env");
44              if (envCtx == null)
45                  out.println("envCtx is NULL");
46
47              // Look up our data source
48              DataSource ds = (DataSource) envCtx.lookup("jdbc/TestDB");
49              Connection connection = ds.getConnection();
```

BrowsingServlet.java 56-64



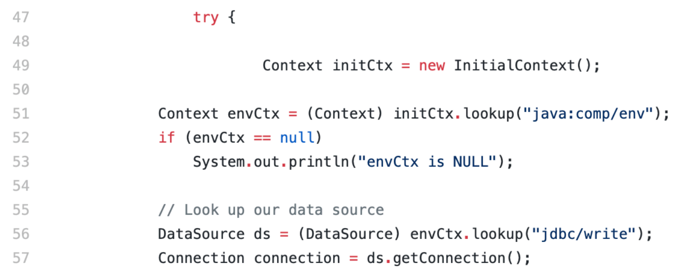
```
55
56                          Context initCtx = new InitialContext();
57
58                      Context envCtx = (Context) initCtx.lookup("java:comp/env");
59                      if (envCtx == null)
60                          out.println("envCtx is NULL");
61
62                      // Look up our data source
63                      DataSource ds = (DataSource) envCtx.lookup("jdbc/TestDB");
64                      Connection connection = ds.getConnection();
```

LoginServlet.java 63-69



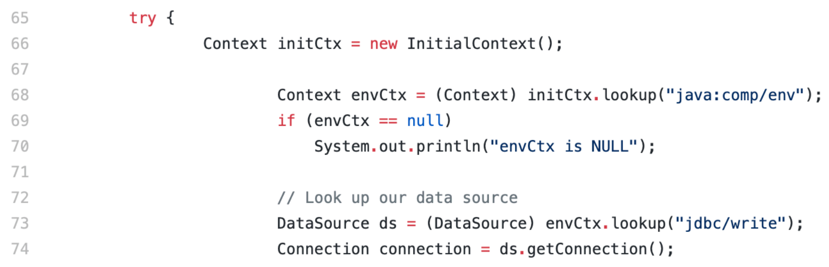
```
59          try {
60
61                  Context initCtx = new InitialContext();
62
63              Context envCtx = (Context) initCtx.lookup("java:comp/env");
64              if (envCtx == null)
65                  System.out.println("envCtx is NULL");
66
67              // Look up our data source
68              DataSource ds = (DataSource) envCtx.lookup("jdbc/TestDB");
69              Connection connection = ds.getConnection();
```

add_movie.java 49-56



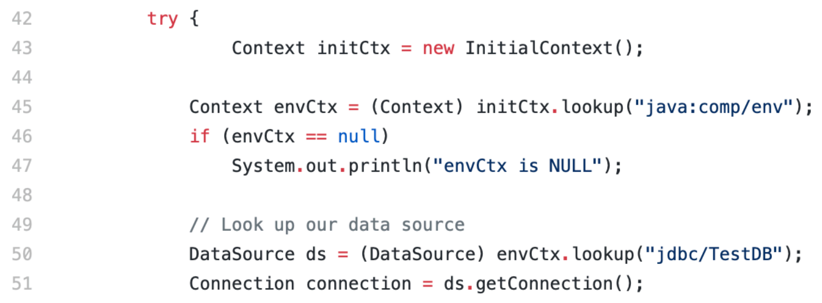
```
47                  try {
48
49                          Context initCtx = new InitialContext();
50
51              Context envCtx = (Context) initCtx.lookup("java:comp/env");
52              if (envCtx == null)
53                  System.out.println("envCtx is NULL");
54
55              // Look up our data source
56              DataSource ds = (DataSource) envCtx.lookup("jdbc/write");
57              Connection connection = ds.getConnection();
```

completecheckoutServlet.java 68-74



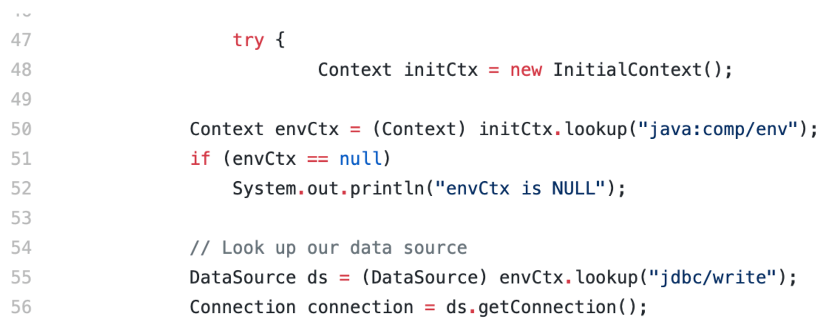
```
65        try {
66              Context initCtx = new InitialContext();
67
68                    Context envCtx = (Context) initCtx.lookup("java:comp/env");
69                    if (envCtx == null)
70                        System.out.println("envCtx is NULL");
71
72                    // Look up our data source
73                    DataSource ds = (DataSource) envCtx.lookup("jdbc/write");
74                    Connection connection = ds.getConnection();
```

dashboard.java 45-51



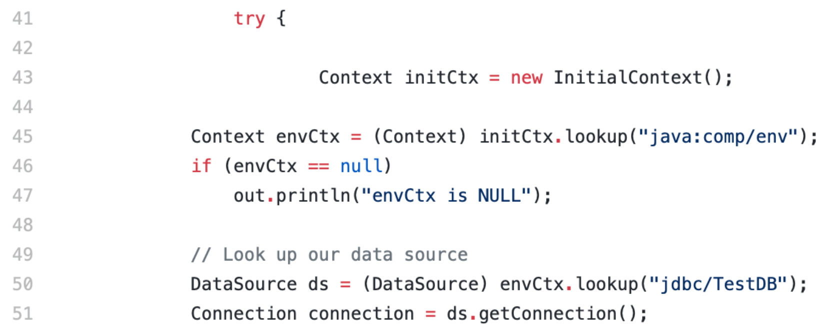
```
42        try {
43              Context initCtx = new InitialContext();
44
45           Context envCtx = (Context) initCtx.lookup("java:comp/env");
46           if (envCtx == null)
47               System.out.println("envCtx is NULL");
48
49           // Look up our data source
50           DataSource ds = (DataSource) envCtx.lookup("jdbc/TestDB");
51           Connection connection = ds.getConnection();
```

makechange.java 50-55



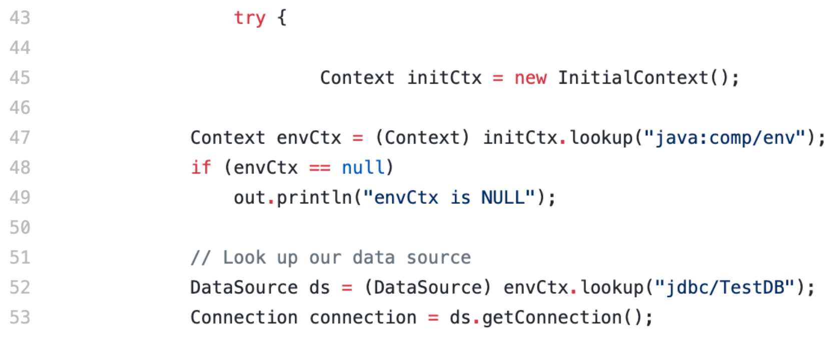
```
47                try {
48                        Context initCtx = new InitialContext();
49
50           Context envCtx = (Context) initCtx.lookup("java:comp/env");
51           if (envCtx == null)
52               System.out.println("envCtx is NULL");
53
54           // Look up our data source
55           DataSource ds = (DataSource) envCtx.lookup("jdbc/write");
56           Connection connection = ds.getConnection();
```

singleMovieServlet.java 43 -50



```
41                try {
42
43                        Context initCtx = new InitialContext();
44
45           Context envCtx = (Context) initCtx.lookup("java:comp/env");
46           if (envCtx == null)
47               out.println("envCtx is NULL");
48
49           // Look up our data source
50           DataSource ds = (DataSource) envCtx.lookup("jdbc/TestDB");
51           Connection connection = ds.getConnection();
```

singleStarServlet.java 45-53



```
43                  try {
44
45                          Context initCtx = new InitialContext();
46
47              Context envCtx = (Context) initCtx.lookup("java:comp/env");
48              if (envCtx == null)
49                  out.println("envCtx is NULL");
50
51              // Look up our data source
52              DataSource ds = (DataSource) envCtx.lookup("jdbc/TestDB");
53              Connection connection = ds.getConnection();
```

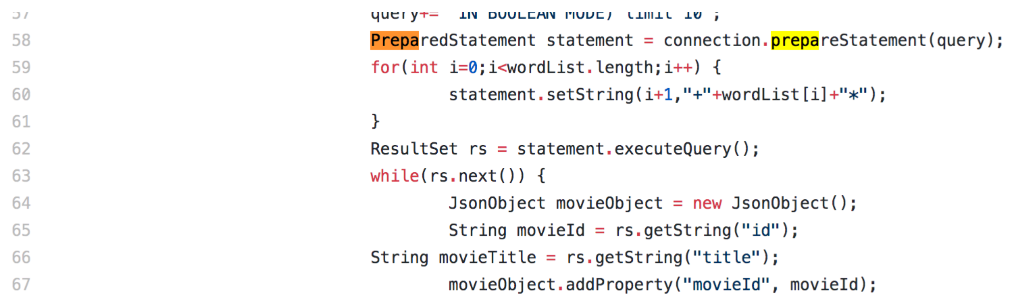
- How did you use Prepared Statements?

  For all plain statement, we changed it to prepared statement and then execute it after we inserted proper variable; So, after execution, It could give us proper data in return.

  - File name, line numbers as in Github
    AutoServlet.java line: 58 ~ 62
    BrowsingServlet.java: 66 ~ 79
    CartServlet.java: 98 ~103
    LoginServlet.java: 71~78
    MovieServlet.java: 118~187
    MovieServlet.java: 260~295
    add_movie.java: 62~113
    completecheckoutServlet.java: 76 ~ 80
    completecheckoutServlet.java: 115~133
    dashboard.java: 54 ~ 60
    makechange.java: 67~75
    singleMovieServlet.java: 55~115
    singleStarServlet.java: 58~109

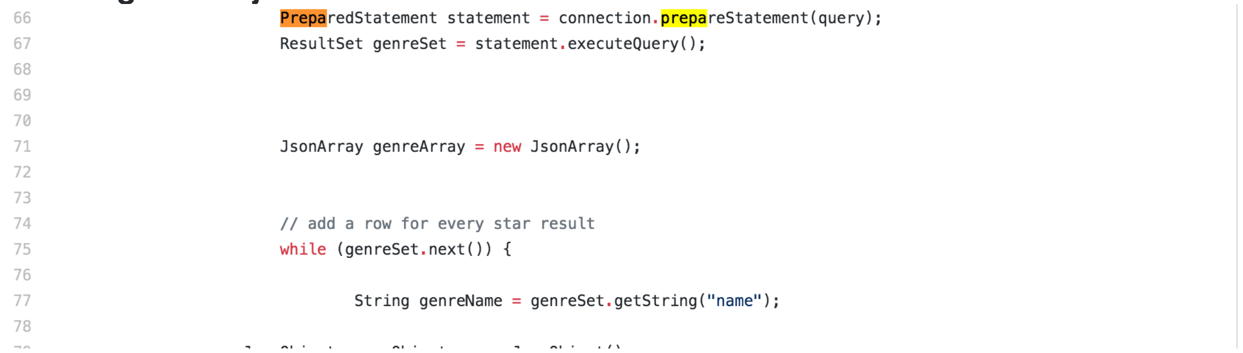
  - Snapshots showing use in your code

AutoServlet.java line: 58 ~ 62



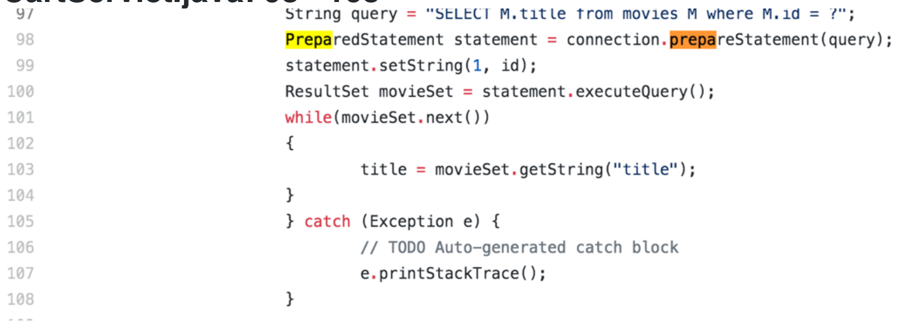
```
57                                  query+=" IN BOOLEAN MODE) limit 10";
58                                  PreparedStatement statement = connection.prepareStatement(query);
59                                  for(int i=0;i<wordList.length;i++) {
60                                          statement.setString(i+1,"+"+wordList[i]+"*");
61                                  }
62                                  ResultSet rs = statement.executeQuery();
63                                  while(rs.next()) {
64                                          JsonObject movieObject = new JsonObject();
65                                          String movieId = rs.getString("id");
66                                  String movieTitle = rs.getString("title");
67                                          movieObject.addProperty("movieId", movieId);
```

BrowsingServlet.java: 66 ~ 79



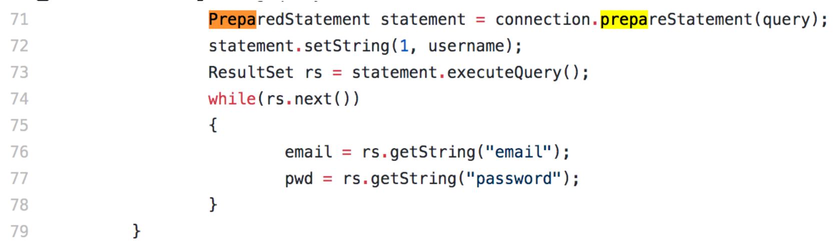
```
66                              PreparedStatement statement = connection.prepareStatement(query);
67                              ResultSet genreSet = statement.executeQuery();
68
69
70
71                              JsonArray genreArray = new JsonArray();
72
73
74                              // add a row for every star result
75                              while (genreSet.next()) {
76
77                                      String genreName = genreSet.getString("name");
78
```

CartServlet.java: 98 ~103



```
97                              String query = "SELECT M.title from movies M where M.id = ?";
98                              PreparedStatement statement = connection.prepareStatement(query);
99                              statement.setString(1, id);
100                             ResultSet movieSet = statement.executeQuery();
101                             while(movieSet.next())
102                             {
103                                     title = movieSet.getString("title");
104                             }
105                             } catch (Exception e) {
106                                     // TODO Auto-generated catch block
107                                     e.printStackTrace();
108                             }
```

LoginServlet.java: 71~78

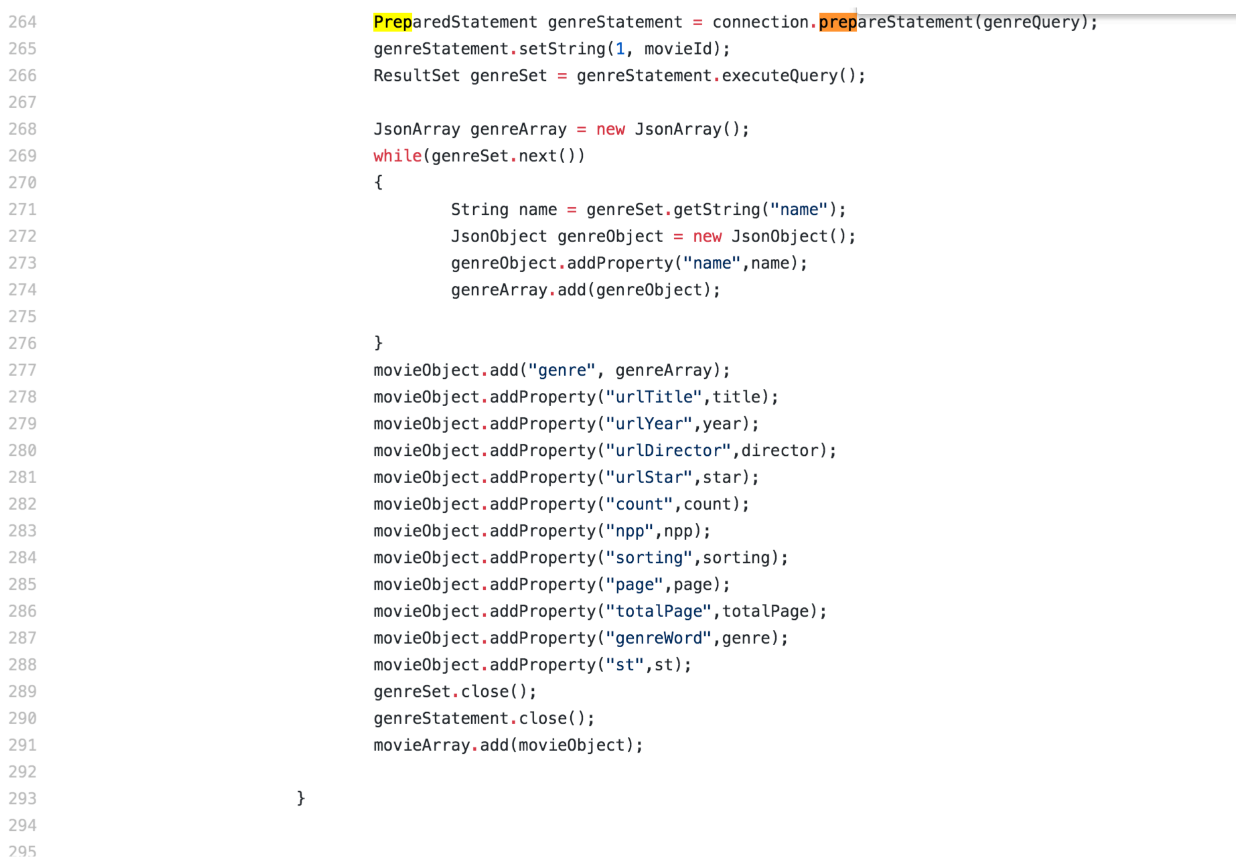


```
71                      PreparedStatement statement = connection.prepareStatement(query);
72                      statement.setString(1, username);
73                      ResultSet rs = statement.executeQuery();
74                      while(rs.next())
75                      {
76                              email = rs.getString("email");
77                              pwd = rs.getString("password");
78                      }
79              }
```

MovieServlet.java: 118~187

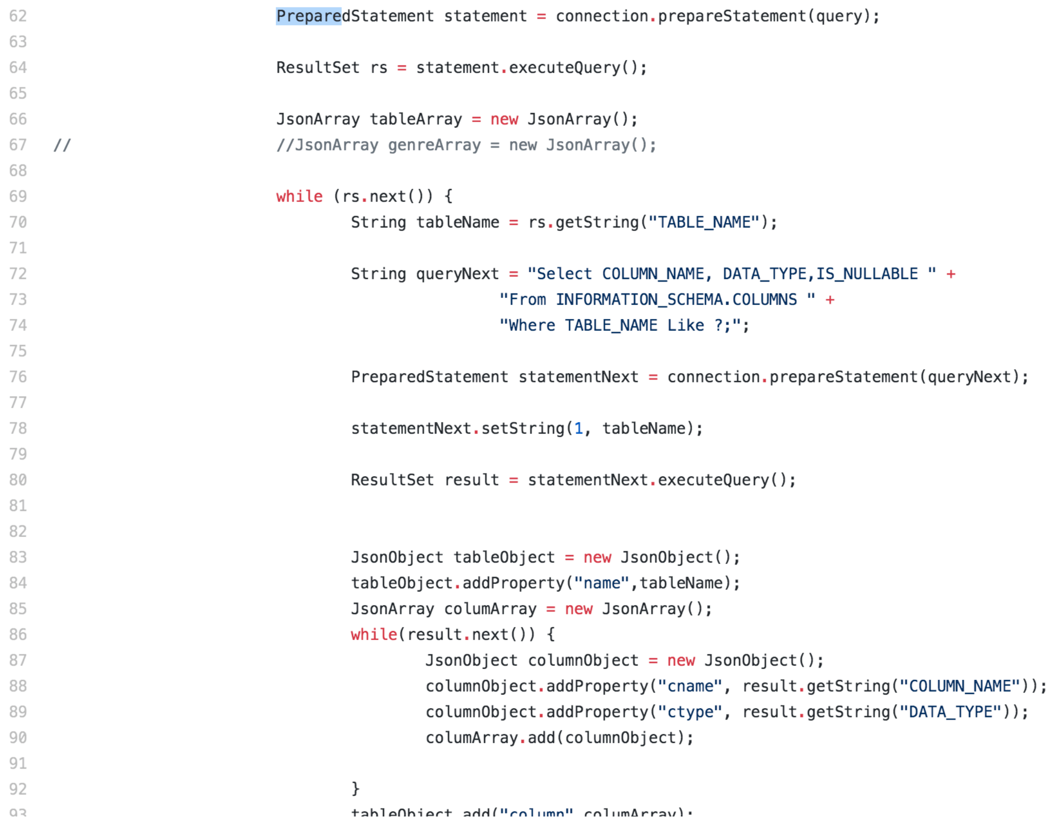
```java
PreparedStatement countStatement = connection.prepareStatement(countQuery);
        PreparedStatement statement = connection.prepareStatement(tableQuery);


if(year.length()>0)
{
        statement.setInt(1,Integer.parseInt(year));
        statement.setInt(2,Integer.parseInt(year));
        countStatement.setInt(1,Integer.parseInt(year));
        countStatement.setInt(2,Integer.parseInt(year));
}
else {
        statement.setInt(1,1000);
        statement.setInt(2,9999);
        countStatement.setInt(1,1000);
        countStatement.setInt(2,9999);
}
if(director.length()>0)
{
        statement.setString(3,"%"+director+"%");
        countStatement.setString(3,"%"+director+"%");
}
else {
        statement.setString(3,"%");
        countStatement.setString(3,"%");
}
if(star.length()>0)
{
        statement.setString(4,"%"+star+"%");
        countStatement.setString(4,"%"+star+"%");
}
else {
```

MovieServlet.java: 260~295



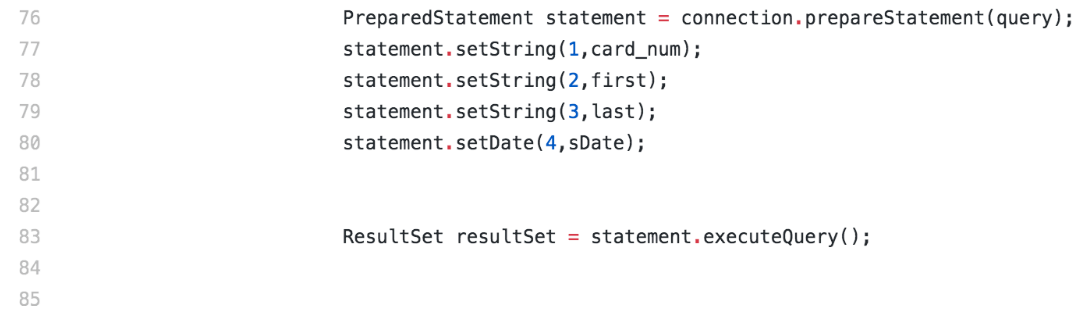
```
                PreparedStatement genreStatement = connection.prepareStatement(genreQuery);
                genreStatement.setString(1, movieId);
                ResultSet genreSet = genreStatement.executeQuery();

                JsonArray genreArray = new JsonArray();
                while(genreSet.next())
                {
                        String name = genreSet.getString("name");
                        JsonObject genreObject = new JsonObject();
                        genreObject.addProperty("name",name);
                        genreArray.add(genreObject);

                }
                movieObject.add("genre", genreArray);
                movieObject.addProperty("urlTitle",title);
                movieObject.addProperty("urlYear",year);
                movieObject.addProperty("urlDirector",director);
                movieObject.addProperty("urlStar",star);
                movieObject.addProperty("count",count);
                movieObject.addProperty("npp",npp);
                movieObject.addProperty("sorting",sorting);
                movieObject.addProperty("page",page);
                movieObject.addProperty("totalPage",totalPage);
                movieObject.addProperty("genreWord",genre);
                movieObject.addProperty("st",st);
                genreSet.close();
                genreStatement.close();
                movieArray.add(movieObject);

        }

```

add_movie.java: 62~113



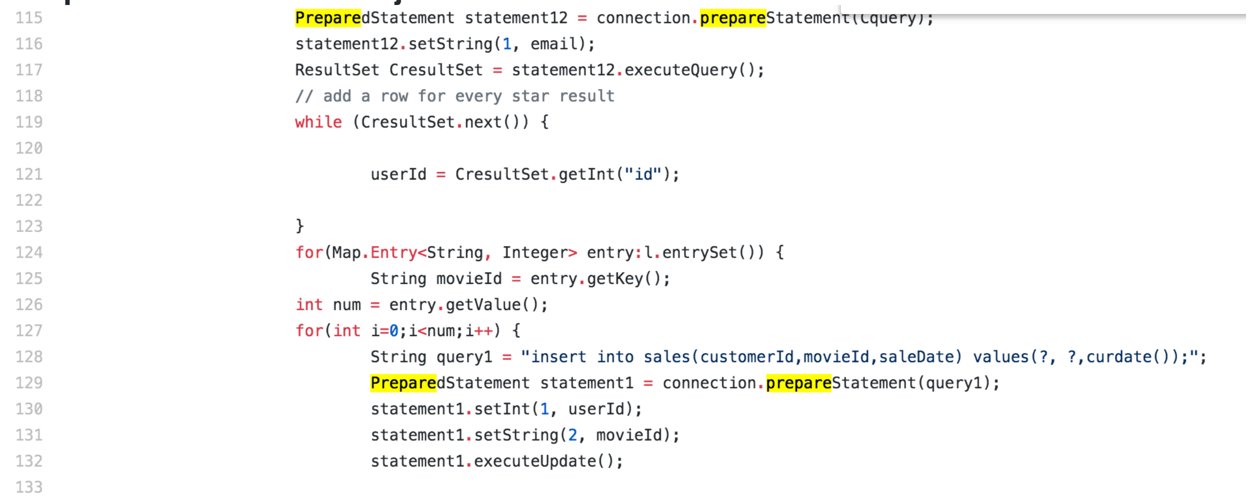
```
                        PreparedStatement statement = connection.prepareStatement(query);

                        ResultSet rs = statement.executeQuery();

                        JsonArray tableArray = new JsonArray();
//                      //JsonArray genreArray = new JsonArray();

                        while (rs.next()) {
                                String tableName = rs.getString("TABLE_NAME");

                                String queryNext = "Select COLUMN_NAME, DATA_TYPE,IS_NULLABLE " +
                                                "From INFORMATION_SCHEMA.COLUMNS " +
                                                "Where TABLE_NAME Like ?;";

                                PreparedStatement statementNext = connection.prepareStatement(queryNext);

                                statementNext.setString(1, tableName);

                                ResultSet result = statementNext.executeQuery();


                                JsonObject tableObject = new JsonObject();
                                tableObject.addProperty("name",tableName);
                                JsonArray columArray = new JsonArray();
                                while(result.next()) {
                                        JsonObject columnObject = new JsonObject();
                                        columnObject.addProperty("cname", result.getString("COLUMN_NAME"));
                                        columnObject.addProperty("ctype", result.getString("DATA_TYPE"));
                                        columArray.add(columnObject);

                                }
                                tableObject.add("column", columArray);
```

completecheckoutServlet.java: 76 ~ 80



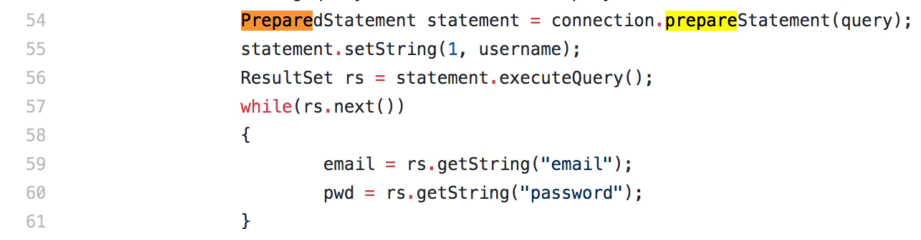
```
76     PreparedStatement statement = connection.prepareStatement(query);
77     statement.setString(1,card_num);
78     statement.setString(2,first);
79     statement.setString(3,last);
80     statement.setDate(4,sDate);
81
82
83     ResultSet resultSet = statement.executeQuery();
84
85
```

completecheckoutServlet.java: 115~133



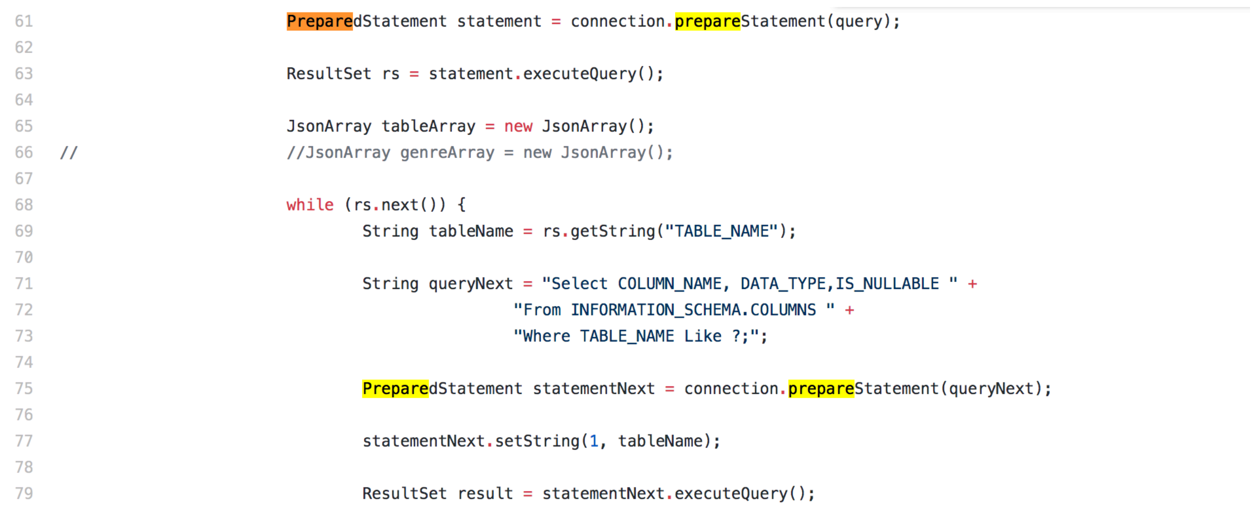
```
115    PreparedStatement statement12 = connection.prepareStatement(cquery);
116    statement12.setString(1, email);
117    ResultSet CresultSet = statement12.executeQuery();
118    // add a row for every star result
119    while (CresultSet.next()) {
120
121            userId = CresultSet.getInt("id");
122
123    }
124    for(Map.Entry<String, Integer> entry:l.entrySet()) {
125            String movieId = entry.getKey();
126    int num = entry.getValue();
127    for(int i=0;i<num;i++) {
128            String query1 = "insert into sales(customerId,movieId,saleDate) values(?, ?,curdate());";
129            PreparedStatement statement1 = connection.prepareStatement(query1);
130            statement1.setInt(1, userId);
131            statement1.setString(2, movieId);
132            statement1.executeUpdate();
133
```

dashboard.java: 54 ~ 60



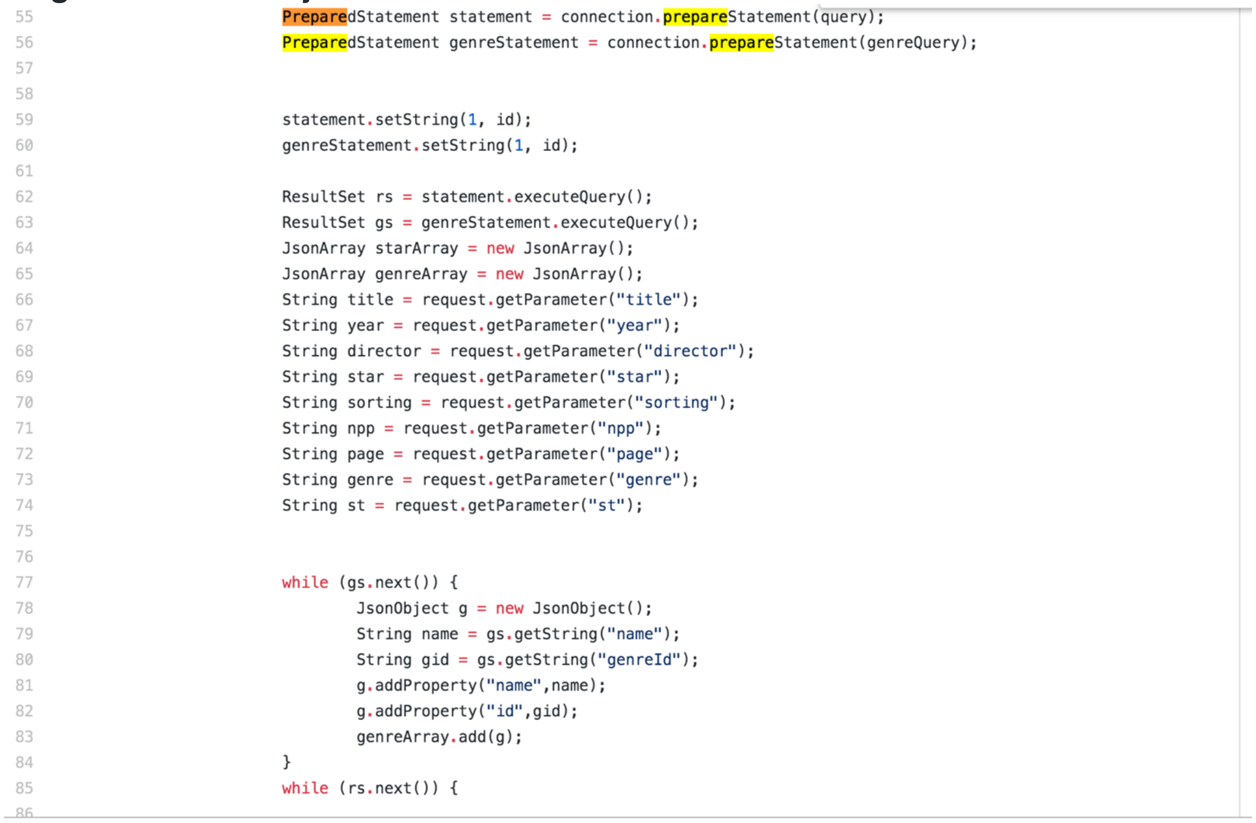
```
54     PreparedStatement statement = connection.prepareStatement(query);
55     statement.setString(1, username);
56     ResultSet rs = statement.executeQuery();
57     while(rs.next())
58     {
59             email = rs.getString("email");
60             pwd = rs.getString("password");
61     }
```

makechange.java: 67~75



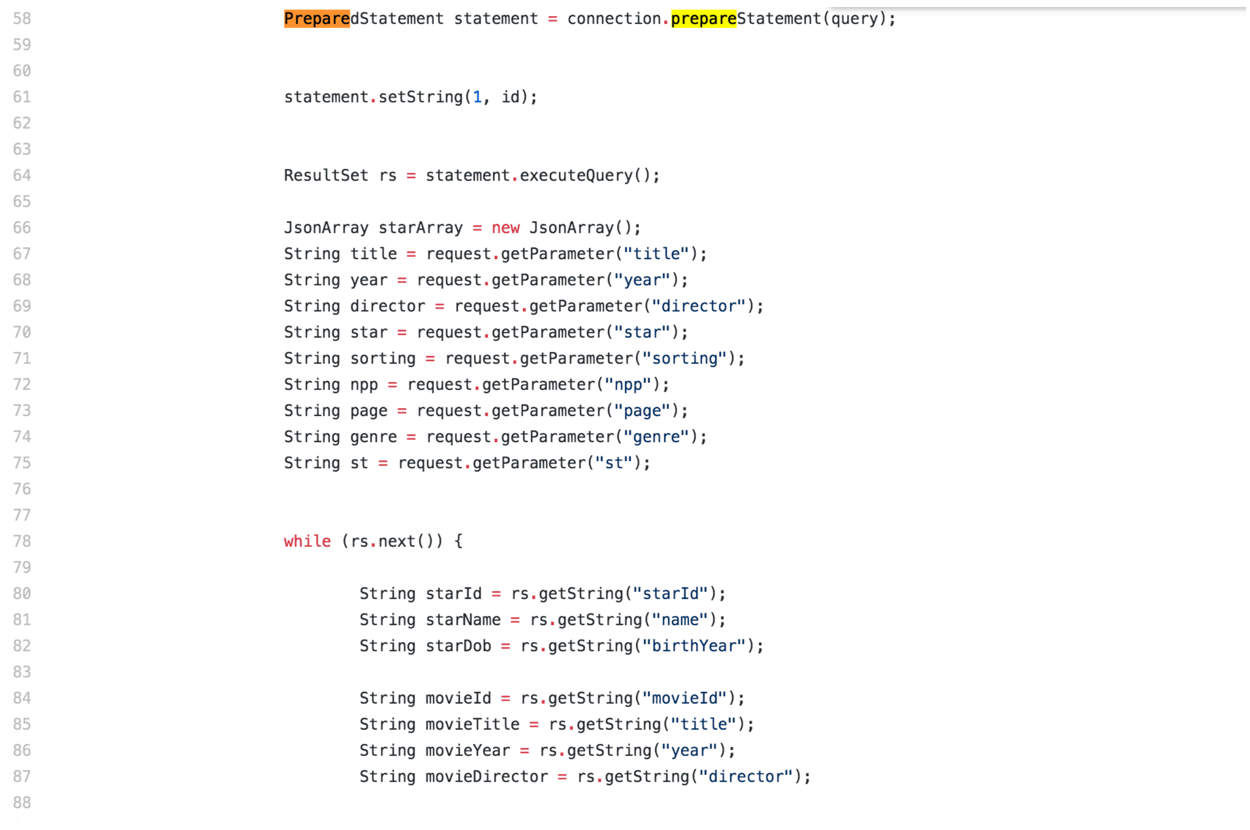
```java
                        PreparedStatement statement = connection.prepareStatement(query);

                        ResultSet rs = statement.executeQuery();

                        JsonArray tableArray = new JsonArray();
//                      //JsonArray genreArray = new JsonArray();

                        while (rs.next()) {
                                String tableName = rs.getString("TABLE_NAME");

                                String queryNext = "Select COLUMN_NAME, DATA_TYPE,IS_NULLABLE " +
                                                "From INFORMATION_SCHEMA.COLUMNS " +
                                                "Where TABLE_NAME Like ?;";

                                PreparedStatement statementNext = connection.prepareStatement(queryNext);

                                statementNext.setString(1, tableName);

                                ResultSet result = statementNext.executeQuery();
```

singleMovieServlet.java: 55~115



```java
                        PreparedStatement statement = connection.prepareStatement(query);
                        PreparedStatement genreStatement = connection.prepareStatement(genreQuery);


                        statement.setString(1, id);
                        genreStatement.setString(1, id);

                        ResultSet rs = statement.executeQuery();
                        ResultSet gs = genreStatement.executeQuery();
                        JsonArray starArray = new JsonArray();
                        JsonArray genreArray = new JsonArray();
                        String title = request.getParameter("title");
                        String year = request.getParameter("year");
                        String director = request.getParameter("director");
                        String star = request.getParameter("star");
                        String sorting = request.getParameter("sorting");
                        String npp = request.getParameter("npp");
                        String page = request.getParameter("page");
                        String genre = request.getParameter("genre");
                        String st = request.getParameter("st");


                        while (gs.next()) {
                                JsonObject g = new JsonObject();
                                String name = gs.getString("name");
                                String gid = gs.getString("genreId");
                                g.addProperty("name",name);
                                g.addProperty("id",gid);
                                genreArray.add(g);
                        }
                        while (rs.next()) {
```

singleStarServlet.java: 58~109



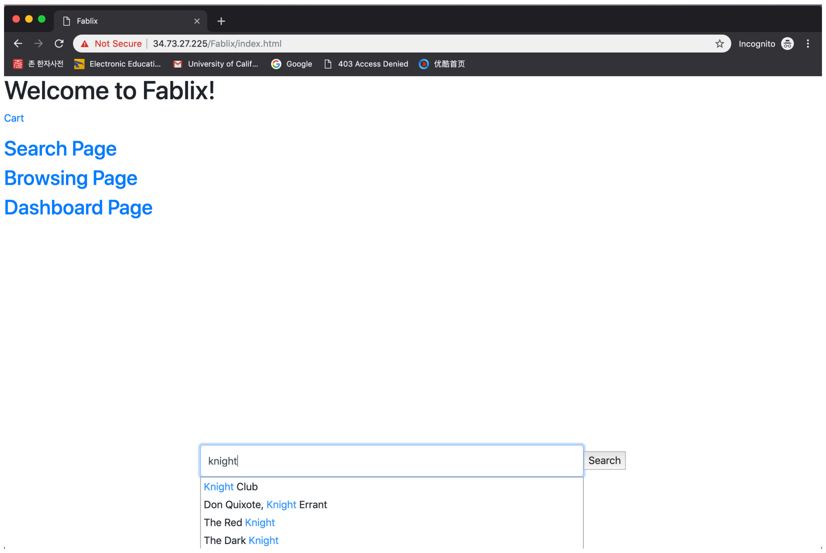
```
PreparedStatement statement = connection.prepareStatement(query);


statement.setString(1, id);


ResultSet rs = statement.executeQuery();

JsonArray starArray = new JsonArray();
String title = request.getParameter("title");
String year = request.getParameter("year");
String director = request.getParameter("director");
String star = request.getParameter("star");
String sorting = request.getParameter("sorting");
String npp = request.getParameter("npp");
String page = request.getParameter("page");
String genre = request.getParameter("genre");
String st = request.getParameter("st");


while (rs.next()) {

        String starId = rs.getString("starId");
        String starName = rs.getString("name");
        String starDob = rs.getString("birthYear");

        String movieId = rs.getString("movieId");
        String movieTitle = rs.getString("title");
        String movieYear = rs.getString("year");
        String movieDirector = rs.getString("director");

```

<u>Task 2</u>

- Address of AWS and Google instances
  AWS Instance 1: 52.53.239.70
  AWS Instance 2(Master): 52.53.210.87
  AWS Instance 3(Slave): 54.219.150.85
  GCP: 34.73.27.225

- Have you verified that they are accessible? Does Fablix site get opened both on Google's 80 port and AWS' 8080 port?
  - Yes, the port 80 for GCP and 80 for AWS Instance1 and 8080 for all 3 AWS instances are open and accessible as in the snapshot. And the name of the website is Fablix.

  - google: url: http://34.73.27.225/Fablix/index.html



Fablix

Not Secure | 34.73.27.225/Fablix/index.html

Welcome to Fablix!

Cart

Search Page

Browsing Page

Dashboard Page

knight

Search

Knight Club

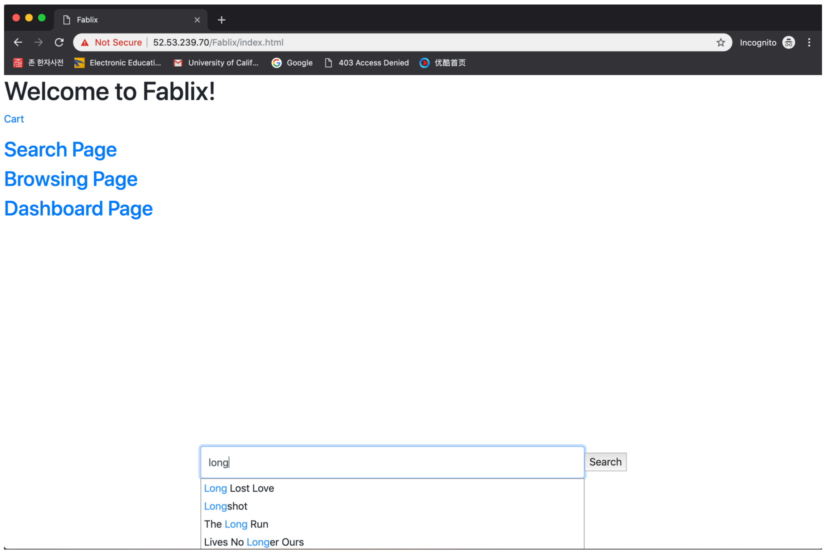
Don Quixote, Knight Errant

The Red Knight

The Dark Knight

-
-
-
-
-
-
-
-

AWS Instance 1: 80 url: http://52.53.239.70/Fablix/index.html



Fablix

Not Secure | 52.53.239.70/Fablix/index.html

Welcome to Fablix!

Cart

Search Page

Browsing Page

Dashboard Page

long

Search

Long Lost Love

Longshot

The Long Run

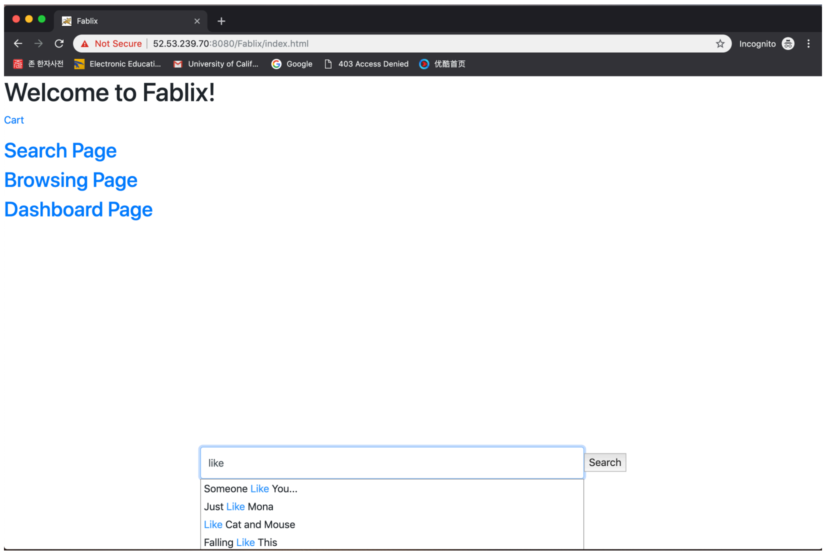
Lives No Longer Ours

-
-
-
-
-
-
-
-
-
-
-
-
-

AWS Instance 1: 8080 url: http://52.53.239.70:8080/Fablix/index.html

-

-
-
-
-
-
-
-
-
-
-
-
-
-
-
-



Welcome to Fablix!

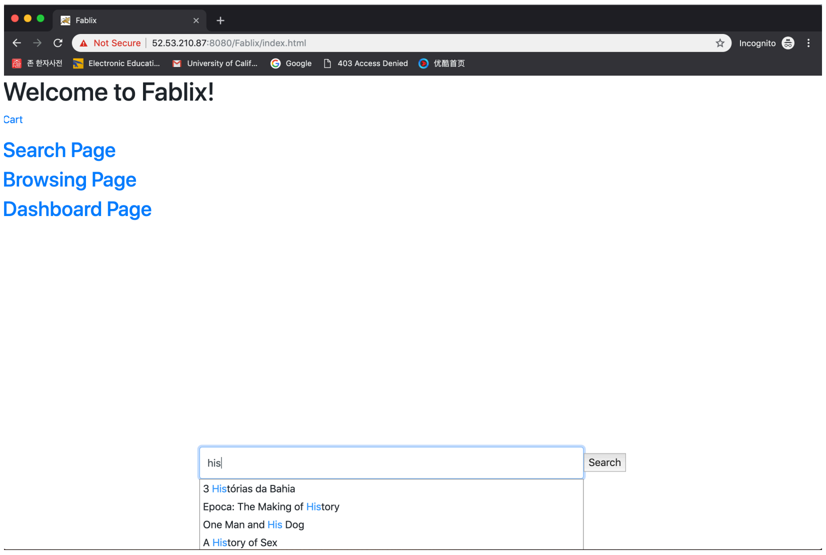
Cart

Search Page
Browsing Page
Dashboard Page

like
Search

Someone Like You...
Just Like Mona
Like Cat and Mouse
Falling Like This

AWS Instance 2: 8080 url: http://52.53.210.87:8080/Fablix/index.html



Welcome to Fablix!

Cart

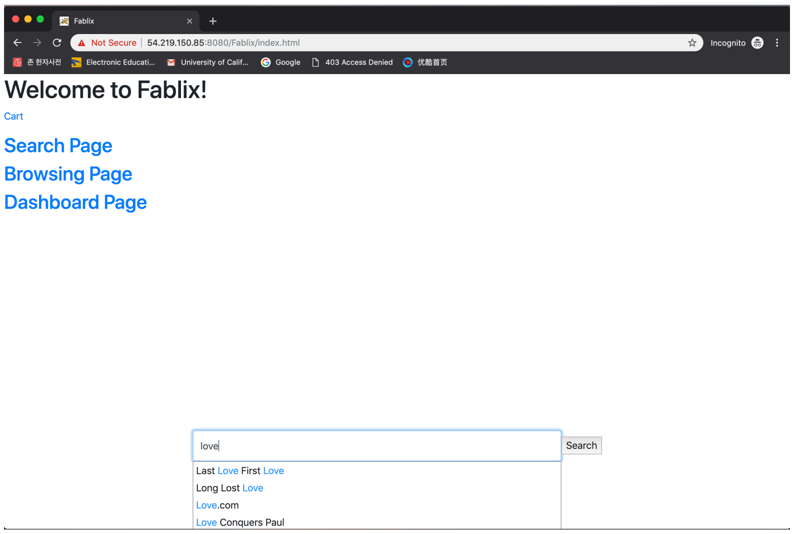
Search Page
Browsing Page
Dashboard Page

his
Search

3 Histórias da Bahia
Epoca: The Making of History
One Man and His Dog
A History of Sex

AWS Instance 3: 8080 url: http://54.219.150.85:8080/Fablix/index.html

-
-
-
-
-
-
-
-
-



Welcome to Fablix!

Cart

Search Page
Browsing Page
Dashboard Page

love
Search

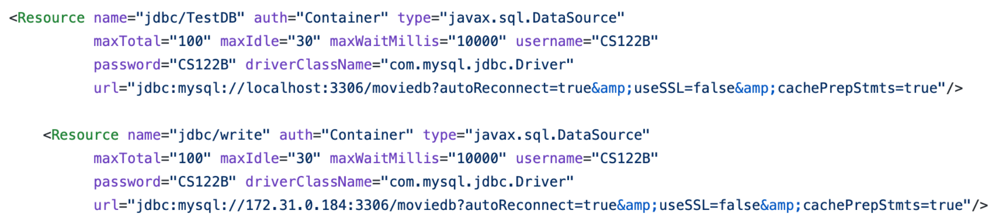
Last Love First Love
Long Lost Love
Love.com
Love Conquers Paul

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

There are 2 resources added into context.xml, one is called testDB and the other one is called write, the testDB has the ip address of localhost, which means whenever the backend is connected and need to read from the database, it will always connect to its own database, but when a backend is connected and when it needs to write something into database, it will always connect to master's database and in order to do this, I also did grant the user on slave to access the database on master.

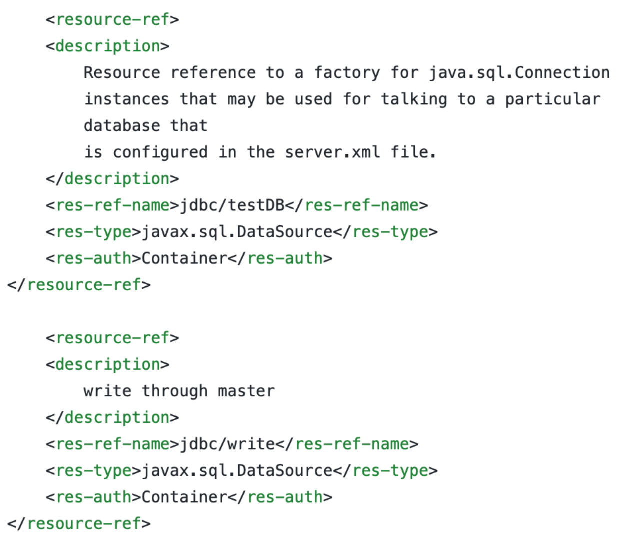
    - File name, line numbers as in Github
      project5/WebContent/META-INF/**context.xml 15-23**
      project5/WebContent/WEB-INF/**web.xml 21-40**

    - Snapshots
    - context.xml



```
<Resource name="jdbc/TestDB" auth="Container" type="javax.sql.DataSource"
          maxTotal="100" maxIdle="30" maxWaitMillis="10000" username="CS122B"
          password="CS122B" driverClassName="com.mysql.jdbc.Driver"
          url="jdbc:mysql://localhost:3306/moviedb?autoReconnect=true&amp;useSSL=false&amp;cachePrepStmts=true"/>

    <Resource name="jdbc/write" auth="Container" type="javax.sql.DataSource"
          maxTotal="100" maxIdle="30" maxWaitMillis="10000" username="CS122B"
          password="CS122B" driverClassName="com.mysql.jdbc.Driver"
          url="jdbc:mysql://172.31.0.184:3306/moviedb?autoReconnect=true&amp;useSSL=false&amp;cachePrepStmts=true"/>
```

- web.xml



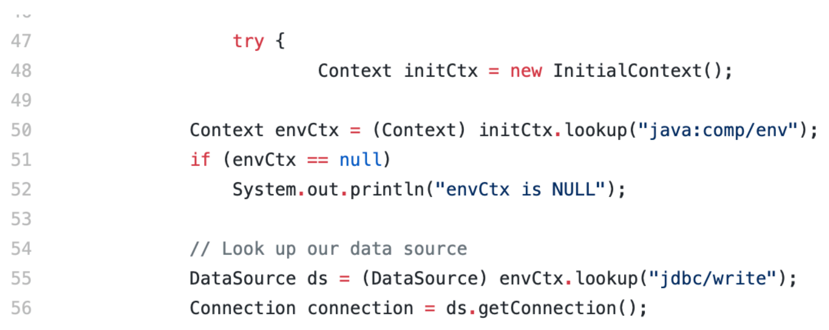
```
    <resource-ref>
    <description>
        Resource reference to a factory for java.sql.Connection
        instances that may be used for talking to a particular
        database that
        is configured in the server.xml file.
    </description>
    <res-ref-name>jdbc/testDB</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
</resource-ref>

    <resource-ref>
    <description>
        write through master
    </description>
    <res-ref-name>jdbc/write</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
</resource-ref>
```

- How read/write requests were routed?
    - When a servlet needs to do a write, it will always look up the "write" connection in the context.xml to connect to database only on master. When a servlet needs to

do a read, it will look up the “testDB” in the context.xml to connect to database on either master or slave

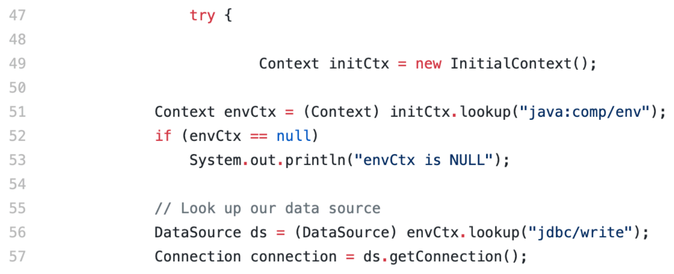
- File name, line numbers as in Github
  /project5/src/makechange.java 50-55
  /project5/src/add_movie.java 49-56
  /project5/src/completecheckoutServlet.java 68-74

- Snapshots
- makechange.java 50-55



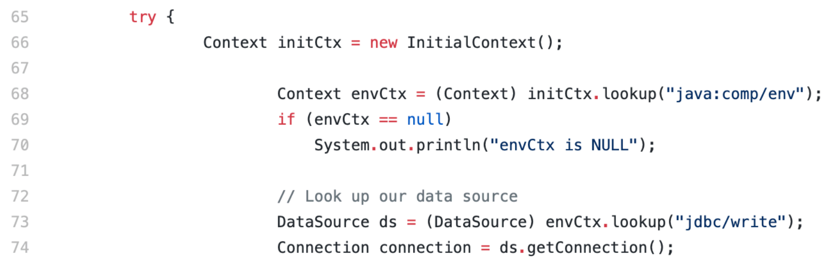
```
47                  try {
48                          Context initCtx = new InitialContext();
49
50              Context envCtx = (Context) initCtx.lookup("java:comp/env");
51              if (envCtx == null)
52                  System.out.println("envCtx is NULL");
53
54              // Look up our data source
55              DataSource ds = (DataSource) envCtx.lookup("jdbc/write");
56              Connection connection = ds.getConnection();
```

add_movie.java 49-56



```
47                  try {
48
49                          Context initCtx = new InitialContext();
50
51              Context envCtx = (Context) initCtx.lookup("java:comp/env");
52              if (envCtx == null)
53                  System.out.println("envCtx is NULL");
54
55              // Look up our data source
56              DataSource ds = (DataSource) envCtx.lookup("jdbc/write");
57              Connection connection = ds.getConnection();
```

completecheckoutServlet.java 68-74



```
65          try {
66                  Context initCtx = new InitialContext();
67
68                          Context envCtx = (Context) initCtx.lookup("java:comp/env");
69                          if (envCtx == null)
70                              System.out.println("envCtx is NULL");
71
72                          // Look up our data source
73                          DataSource ds = (DataSource) envCtx.lookup("jdbc/write");
74                          Connection connection = ds.getConnection();
```

Task 3

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

  Yes, it is in the Time measure folder, and the log file for every test is in its own folder

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

  Yes, it is in the Time measure folder

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

  Yes, it is in the cs122b-winter19-team-2/Time measure/calTime.py

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

  yes, the war file is in the root of our team repo, called Fablix.war