[all classes] [models]

Coverage Summary for Class: Product (models)

Class	Class, %	Method, %	Line, %
Product	100% (1/ 1)	83.8% (31/ 37)	80.5% (136/ 169)

```
 1  package models;
 2
 3  import java.sql.Connection;
 4  import java.sql.DriverManager;
 5  import java.sql.SQLException;
 6  import java.sql.Statement;
 7  import java.util.Date;
 8  import java.sql.ResultSet;
 9
10  /**
11   * Created by akshay on 10/29/2016.
12   */
13  public class Product {
14      private long id;
15      private String uploadedBy;
16      private String imagePath;
17      private float price;
18      private String description;
19      private Date dateUploaded;
20      private Date dateSold;
21      private float priceBought;
22      private String onlineLink;
23      private float soldPrice;
24      private int condition;
25      private int months;
26      private int category;
27      private String location;
28
29      public Product() {
30
31      }
32
33      public Product(long id, String uploadedBy, String imagePath, float price, String description,
34                     Date dateUploaded, Date dateSold, float priceBought, String onlineLink,
35                     float soldPrice, int condition, int months, int category, String location) {
36          System.out.println("Got to product constructor");
37          this.id = id;
38          this.uploadedBy = uploadedBy;
39          this.imagePath = imagePath;
40          this.price = price;
41          this.description = description;
42          this.dateUploaded = dateUploaded;
43          this.dateSold = dateSold;
44          this.priceBought = priceBought;
45          this.onlineLink = onlineLink;
46          this.soldPrice = soldPrice;
47          this.condition = condition;
48          this.months = months;
49          this.category = category;
50          this.location = location;
51      }
52
53      public long getId() {
54
55          return id;
56      }
57
58      public void setId(long id) {
59          this.id = id;
60      }
61
62      public String getImagePath() {
63          return imagePath;
64      }
65
66      public void setImagePath(String imagePath) {
67          this.imagePath = imagePath;
```

```
68      }
69
70      public float getPrice() {
71          return price;
72      }
73
74      public void setPrice(float price) {
75          this.price = price;
76      }
77
78      public String getDescription() {
79          return description;
80      }
81
82      public void setDescription(String description) {
83          this.description = description;
84      }
85
86      public Date getDateUploaded() {
87          return dateUploaded;
88      }
89
90      public void setDateUploaded(Date dateUploaded) {
91          this.dateUploaded = dateUploaded;
92      }
93
94      public Date getDateSold() {
95          return dateSold;
96      }
97
98      public void setDateSold(Date dateSold) {
99          this.dateSold = dateSold;
100     }
101
102     public String getOnlineLink() {
103         return onlineLink;
104     }
105
106     public void setOnlineLink(String onlineLink) {
107         this.onlineLink = onlineLink;
108     }
109
110     public float getSoldPrice() {
111         return soldPrice;
112     }
113
114     public void setSoldPrice(float soldPrice) {
115         this.soldPrice = soldPrice;
116     }
117
118     public int getCondition() {
119         return condition;
120     }
121
122     public void setCondition(int condition) {
123         this.condition = condition;
124     }
125
126     public int getMonths() {
127         return months;
128     }
129
130     public String mapMonthsToString() {
131         String[] months = new String[4];
132         months[0] = "Less than 3 months";
133         months[1] = "3-6 months";
134         months[2] = "6 months - 3 years";
135         months[3] = ">3 years";
136         return months[this.getMonths()-1];
137     }
138
139     public void setMonths(int months) {
140         this.months = months;
141     }
142     public float getPriceBought() {
143         return priceBought;
144     }
145
146     public void setPriceBought(float priceBought) {
```

```
147         this.priceBought = priceBought;
148     }
149
150     public String getUploadedBy() {
151         return uploadedBy;
152     }
153
154     public void setUploadedBy(String uploadedBy) {
155         this.uploadedBy = uploadedBy;
156     }
157     public int getCategory() {
158         return category;
159     }
160
161     public void setCategory(int category) {
162         this.category = category;
163     }
164
165     public String getLocation() {
166         return location;
167     }
168
169     public void setLocation(String location) {
170         this.location = location;
171     }
172
173     public boolean checkConditions(){
174         if(price > 999999 || price < 0 || imagePath.length()>100 || imagePath.length() == 0 || priceBought > 999999
175                 || priceBought < 0 || description.length() == 0 || description.length() > 65535 ||
176             category > 4 || category < 1 || onlineLink.length() > 255 || condition > 5 || condition < 1 ||
177                 months > 4 || months < 1 || location.length()>255 || location.length() == 0)
178             return false;
179         return true;
180     }
181
182     public boolean addProductToDatabase() throws ClassNotFoundException {
183         return addProductToDatabase(false);
184     }
185
186     public boolean addProductToDatabase(boolean isTest) throws ClassNotFoundException {
187         String myDriver = null;
188         String myURL = null;
189         Connection conn = null;
190         boolean returnVal = true;
191         if(isTest) {
192             myDriver = "com.mysql.jdbc.Driver";
193             myURL = "jdbc:mysql://localhost:3306/mydatabase?zeroDateTimeBehavior=convertToNull";
194         }
195         else{
196             myDriver = "com.mysql.jdbc.Driver";
197             myURL = "jdbc:mysql://lionmart.cvkcqiaoutkr.us-east-1.rds.amazonaws.com:3306/lionmart";
198         }
199         try {
200             Class.forName(myDriver);
201             if(isTest)
202             {
203                 conn = DriverManager.getConnection(myURL, "root", "");
204             }
205             else
206             {
207                 conn = DriverManager.getConnection(myURL, "lionadmin", "lionlynx42");
208             }
209             Statement st = conn.createStatement();
210             st.executeUpdate("CREATE TABLE IF NOT EXISTS product (id INT PRIMARY KEY, price DECIMAL(8,2), imagepath VARCHAR(100),category INT NOT NULL,price_bought DECIMAL(8,2) NOT NULL,description TEXT NOT NULL,date_upl
211
212             java.sql.Timestamp product_timestamp = new java.sql.Timestamp(this.getDateUploaded().getTime());
213             //Check conditions before actually attempting to insert into database
214             boolean shouldInsert = checkConditions();
215             if (shouldInsert)
216                 st.executeUpdate("INSERT INTO product(id,imagepath, price, category, price_bought, description, date_upload,online_link,price_sold,product_condition,months_used,location,user_id) VALUES ("+this.getId()+",
217             else
218                 return false;
219             //Confirm that product is, in fact, inserted into DB.
220             ResultSet rs = st.executeQuery("SELECT * from product where id = "+id);
221             if(!rs.next())
222                 returnVal = false;
223             if(isTest)
224                 st.executeUpdate("DROP TABLE product;");
225             conn.close();
```

```
226         } catch (SQLException e) {
227             e.printStackTrace();
228             return false;
229         }
230         return returnVal;
231     }
232     // the below method is made for specific test cases and is not redundant.  DO NOT DELETE.
233     public boolean addProductToDatabase2(boolean isTest) throws ClassNotFoundException {
234         String myDriver = null;
235         String myURL = null;
236         Connection conn = null;
237         boolean returnVal = true;
238         if(isTest) {
239             myDriver = "com.mysql.jdbc.Driver";
240             myURL = "jdbc:mysql://localhost:3306/mydatabase?zeroDateTimeBehavior=convertToNull";
241         }
242         else{
243             myDriver = "com.mysql.jdbc.Driver";
244             myURL = "jdbc:mysql://lionmart.cvkcqiaoutkr.us-east-1.rds.amazonaws.com:3306/lionmart";
245         }
246         try {
247             Class.forName(myDriver);
248             if(isTest)
249             {
250                 conn = DriverManager.getConnection(myURL, "root", "");
251             }
252             else
253             {
254                 conn = DriverManager.getConnection(myURL, "lionadmin", "lionlynx42");
255             }
256             Statement st = conn.createStatement();
257             st.executeUpdate("CREATE TABLE IF NOT EXISTS product (id INT PRIMARY KEY, price DECIMAL(8,2), imagepath VARCHAR(100),category INT NOT NULL,price_bought DECIMAL(8,2) NOT NULL,description TEXT NOT NULL,date_upl
258             //TODO img path
259             java.sql.Timestamp product_timestamp = new java.sql.Timestamp(this.getDateUploaded().getTime());
260             //Check conditions before actually attempting to insert into database
261             boolean shouldInsert = checkConditions();
262             if (shouldInsert)
263                 st.executeUpdate("INSERT INTO product(id,imagepath, price, category, price_bought, description, date_upload,online_link,price_sold,product_condition,months_used,location,user_id) VALUES ("+this.getId()+",
264             else
265                 return false;
266             //Confirm that product is, in fact, inserted into DB.
267             ResultSet rs = st.executeQuery("SELECT * from product where id = "+id);
268             if(!rs.next())
269                 returnVal = false;
270             conn.close();
271         } catch (SQLException e) {
272             e.printStackTrace();
273             return false;
274         }
275         return returnVal;
276     }
277
278     public boolean updateProductInDatabase() throws ClassNotFoundException {
279         return updateProductInDatabase(false);
280     }
281
282     public boolean updateProductInDatabase(boolean isTest) throws ClassNotFoundException {
283         String myDriver = null;
284         String myURL = null;
285         Connection conn = null;
286         boolean returnVal = true;
287         if(isTest) {
288             myDriver = "com.mysql.jdbc.Driver";
289             myURL = "jdbc:mysql://localhost:3306/mydatabase?zeroDateTimeBehavior=convertToNull";
290         }
291         else{
292             myDriver = "com.mysql.jdbc.Driver";
293             myURL = "jdbc:mysql://lionmart.cvkcqiaoutkr.us-east-1.rds.amazonaws.com:3306/lionmart";
294         }
295         try {
296             Class.forName(myDriver);
297             if(isTest)
298             {
299                 conn = DriverManager.getConnection(myURL, "root", "");
300             }
301             else
302             {
303                 conn = DriverManager.getConnection(myURL, "lionadmin", "lionlynx42");
304             }
```

```
305             Statement st = conn.createStatement();
306             st.executeUpdate("CREATE TABLE IF NOT EXISTS product (id INT PRIMARY KEY, price DECIMAL(8,2), imagepath VARCHAR(100),category INT NOT NULL,price_bought DECIMAL(8,2) NOT NULL,description TEXT NOT NULL,date_upl
307
308             java.sql.Timestamp product_timestamp = new java.sql.Timestamp(this.getDateUploaded().getTime());
309             //Check conditions before actually attempting to update into database
310             boolean shouldInsert = checkConditions();
311
312             if (shouldInsert) {
313                 System.out.println("shouldinsert is True!");
314                 st.executeUpdate("UPDATE product SET price=" + this.getPrice() + ",category='" + this.getCategory() + "',price_bought=" + this.getPriceBought() + ",description='" + this.getDescription() + "',date_upload=
315             }
316             else
317             {
318                 System.out.println("product upload check conditions failed");
319                 return false;
320             }
321
322             //Confirm that product is, in fact, inserted into DB.
323             ResultSet rs = st.executeQuery("SELECT * from product where id = "+id);
324             if(!rs.next()){
325                 System.out.println("updated product not found!");
326                 returnVal = false;
327             }
328             if(isTest)
329                 st.executeUpdate("DROP TABLE product;");
330             conn.close();
331         } catch (SQLException e) {
332             e.printStackTrace();
333             return false;
334         }
335         return returnVal;
336     }
337 }
```

generated on 2016-12-10 16:50