

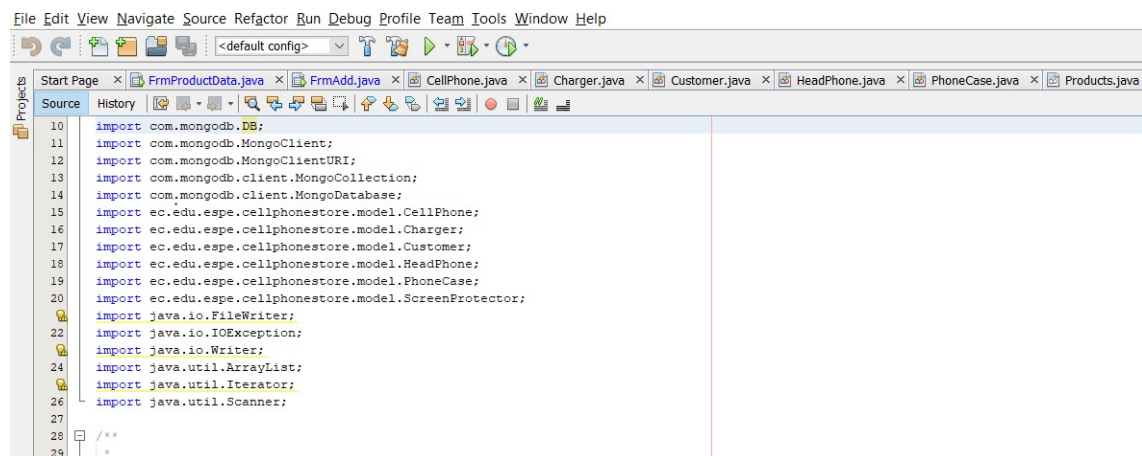
## Inspection to group 3

**Inspectors:** The Programmers; Mena Paul, Mortensen Eduardo, Guaman Byron

**Project:** Innova Code

**From:** Escobar Rivadeneira, Isaac Alejandro, Felix Paredes, Cristian David González Hurtado, Ariel Adrián, Gualotuña Paucar, Richard Fabian

## SCREENS:



The screenshot shows an IDE window with the following imports in a Java file:

```
import com.mongodb.DB;
import com.mongodb.MongoClient;
import com.mongodb.MongoClientURI;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import ec.edu.espe.cellphonestore.model.CellPhone;
import ec.edu.espe.cellphonestore.model.Charger;
import ec.edu.espe.cellphonestore.model.Customer;
import ec.edu.espe.cellphonestore.model.HeadPhone;
import ec.edu.espe.cellphonestore.model.PhoneCase;
import ec.edu.espe.cellphonestore.model.ScreenProtector;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.Writer;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.Scanner;
```

Existence of libraries, which are not necessary in the code



The screenshot shows a snippet of Java code with a highlighted section of unnecessary imports. The code is as follows:

```
System.out.println("Which data product are you going to watch?");
System.out.println("[1] Cellphone.");
System.out.println("[2] Phone case.");
System.out.println("[3] Charger.");
System.out.println("[4] Headphone.");
System.out.println("[5] Screen Protector.");
System.out.println("[6] Go back.");
option1 = scan.nextInt();
scan.nextLine();
switch (option1) {
    case 1:
        System.out.println("Watching CellPhone Data ");
        for (int i = 0; i < cellphones.size(); i++) {
            System.out.println("CellPhone: "
                + cellphones.get(i).getBrand() + " "
                + cellphones.get(i).getModel() + " "
                + cellphones.get(i).getId() + " "
                + " " + cellphones.get(i).getPrice()
                + " "
                + cellphones.get(i).getColor() + "\n");
        }
        CellPhone.showCollection();
        break;
    case 2:
        System.out.println("Watching Phone Cases Data ");
        for (int i = 0; i < phoneCases.size(); i++) {
            System.out.println("Phone Cases: "
                + phoneCases.get(i).getBrand() + " "
                + phoneCases.get(i).getModel() + " "
                + phoneCases.get(i).getId() + " "
                + phoneCases.get(i).getPrice() + " "
                + phoneCases.get(i).getColor() + "\n");
        }
}
```

```

art Page x FrmProductData.java x FrmAdd.java x CellPhone.java x Charger.java x Customer.java x HeadPhone.java x Pr
Source History
12 // scan.nextLine();
13 // cellphones.remove(id);
14 // Iterator it = cellphones.iterator();
15 // while(it.hasNext())
16 // System.out.println(it.next());
17
18 //
19 // CellPhone.deleteCellphone(id);
20 // break;
21
22 // case 2:
23 // System.out.println("Delete");
24 // System.out.println("Position of the product you want to delete: ");
25 // id = scan.nextInt();
26 // scan.nextLine();
27 // phoneCases.remove(id);
28 // Iterator it1 = phoneCases.iterator();
29 // while(it1.hasNext())
30 // System.out.println(it1.next());
31 //
32 //
33 // PhoneCase.deletePhoneCase(id);
34 // break;
35
36 // case 3:
37 // System.out.println("Delete");
38 // System.out.println("Position of the product you want to delete: ");
39 // id = scan.nextInt();
40 // scan.nextLine();
41 // chargers.remove(id);
42 // Iterator it2 = chargers.iterator();
43 // while(it2.hasNext())
44 // System.out.println(it2.next());
45 //
46 //
47 //
48 //
49 //
50 //
51 //

```

Existence of code in the form of comments which do not work

```

207
208 String brand = cmbCellphoneBrand.getSelectedItem().toString();
209 String model = txtCellPhoneModel.getText();
210 String color = cmbCellphoneColor.getSelectedItem().toString();
211 float price = Float.parseFloat(txtCellphonePrice.getText());
212 int id = Integer.valueOf(txtCellphoneId.getText());
213 int stock = (int) spnStock.getValue();
214
215 CellPhone.addCellphone(color, model, id, price, brand, stock);
216 }
217
218 private void btnDeleteCellphoneActionPerformed(java.awt.event.ActionEvent evt) {
219     int id = Integer.valueOf(txtCellphoneId.getText());
220     MongoClient connection = createConnection();
221     DB db = connection.getDB("Products");
222     DBCollection collection = db.getCollection("Cellphone");
223     collection.remove(new BasicDBObject().append("Id", id));
224 }
225
226 /**
227  * @param args the command line arguments
228  */
229 public static void main(String args[]) {
230     /* Set the Nimbus look and feel */
231     Look and feel setting code (optional)
232
233     /* Create and display the form */
234     java.awt.EventQueue.invokeLater(new Runnable() {
235
236     }
237 }

```

This part has to be in a function

```

}
private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    int id = Integer.valueOf(txtId.getText());
    MongoClient connection = createConnection();
    DB db = connection.getDB("Products");
    DBCollection collection = db.getCollection("Headphone");
    collection.remove(new BasicDBObject().append("Id", id));
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {

```

code in spanish

```

17 public Login() {
18     initComponents();
19 }
20
21 /**
22  * This method is called from within the constructor to initialize the form.
23  * WARNING: Do NOT modify this code. The content of this method is always
24  * regenerated by the Form Editor.
25  */
26 @SuppressWarnings("unchecked")
27 // Generated Code
28
29 private void btnIngresarActionPerformed(java.awt.event.ActionEvent evt) {
30     // TODO add your handling code here:
31 }

```

buttons without functionality

```

1 /**
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6 package ec.edu.espe.cellphonestore.model;
7
8 import java.util.ArrayList;
9
10 /**
11  *
12  * @author InnovaCode
13  */
14 public class Warehouse {
15     private int id;
16
17     private ArrayList<CellPhone> cellphones = new ArrayList<>();
18     private ArrayList<PhoneCase> phoneCases = new ArrayList<>();
19     private ArrayList<Charger> chargers = new ArrayList<>();
20     private ArrayList<HeadPhone> headPhones = new ArrayList<>();
21     private ArrayList<ScreenProtector> screenProtectors = new ArrayList<>();
22
23     public int remove(int id) {
24         return 0;
25     }
26 }
27
28
29

```

It is not encapsulated and has a function that does not return anything

```

43 }
44
45 public PhoneCase(String color, String model, String material, int id, float price, String brand, int stock) {
46     super(id, price, brand, stock);
47     this.color = color;
48     this.model = model;
49     this.material = material;
50 }
51
52 public String getMaterial() {
53     throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
54 }
55
56 public String getColor() {
57     throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
58 }
59
60 public String getModel() {
61     throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
62 }
63
64 /**
65  * @param color the color to set
66  */
67 public void setColor(String color) {
68     this.color = color;
69 }

```

Unnecessary functions in code

