UNIVERSIDAD DE LAS FUERZAS ARMADAS

ESPE

Object-oriented programming

NRC: 9642

Team members:

1. Pamela Chipe
2. Joan Cobeña
3. Anabel Davila
4. Andres Espin

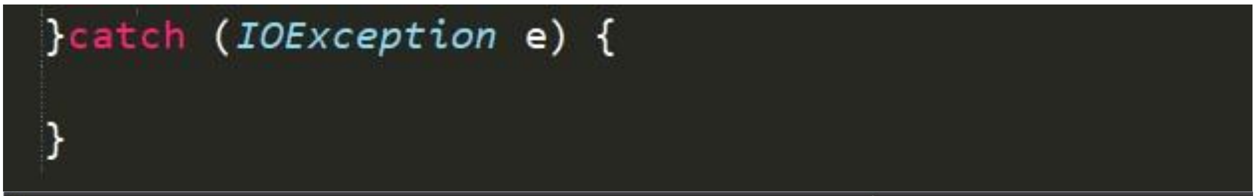
Topic: Project Review

ING. Jorge Edison Lascano.

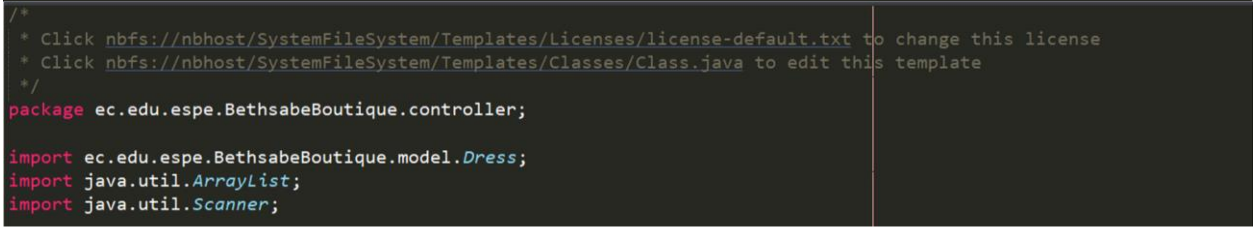
Clean Code Inspection to Team 3 / Software Juniors

By: Team 2 KillChain

Recommendations & corrections.

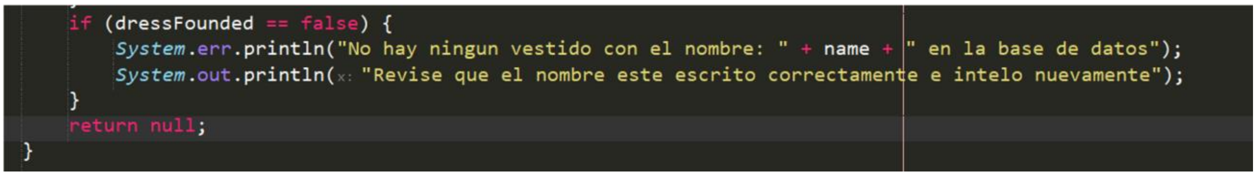


```
}catch (IOException e) {

}
```



```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package ec.edu.espe.BethsabeBoutique.controller;

import ec.edu.espe.BethsabeBoutique.model.Dress;
import java.util.ArrayList;
import java.util.Scanner;
```

BillingSystem



```
        if (dressFounded == false) {
            System.err.println("No hay ningun vestido con el nombre: " + name + " en la base de datos");
            System.out.println(x: "Revise que el nombre este escrito correctamente e intelo nuevamente");
        }
        return null;
    }
```

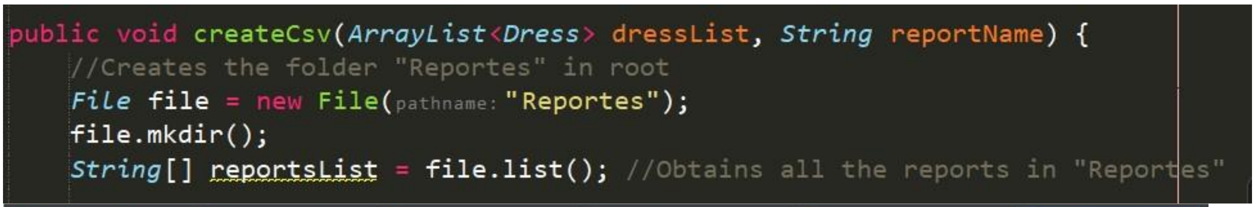
No message for the exception.

DressManager

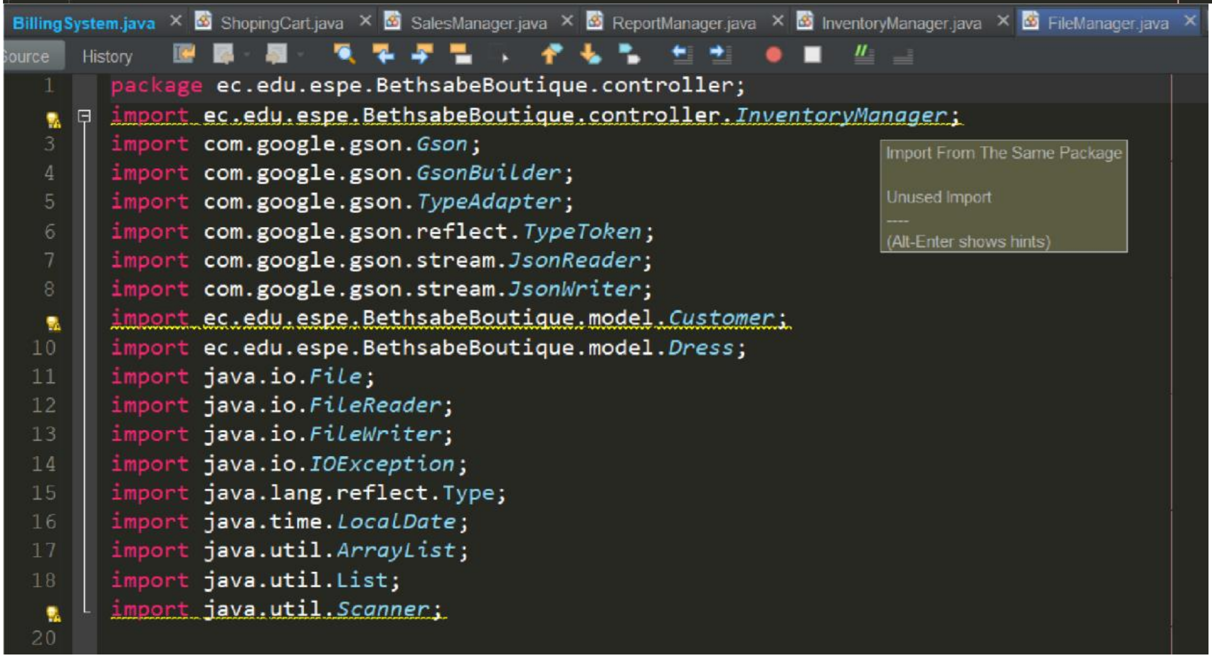
Default header.

Is not recommended to return null, but the first thing to correct here is the message that has an spelling error.

FileManager



```java
public void createCsv(ArrayList<Dress> dressList, String reportName) {
    //Creates the folder "Reportes" in root
    File file = new File(pathname: "Reportes");
    file.mkdir();
    String[] reportsList = file.list(); //Obtains all the reports in "Reportes"
```



```java
package ec.edu.espe.BethsabeBoutique.controller;
import ec.edu.espe.BethsabeBoutique.controller.InventoryManager;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import com.google.gson.TypeAdapter;
import com.google.gson.reflect.TypeToken;
import com.google.gson.stream.JsonReader;
import com.google.gson.stream.JsonWriter;
import ec.edu.espe.BethsabeBoutique.model.Customer;
import ec.edu.espe.BethsabeBoutique.model.Dress;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.lang.reflect.Type;
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
```

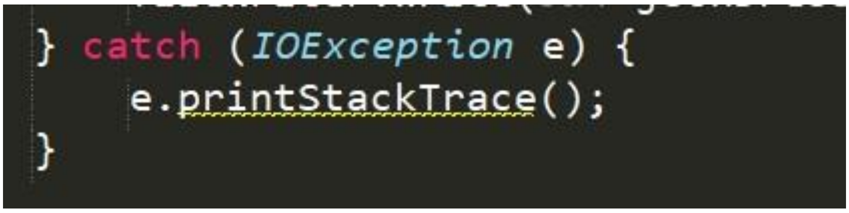
Import From The Same Package

Unused Import

(Alt-Enter shows hints)

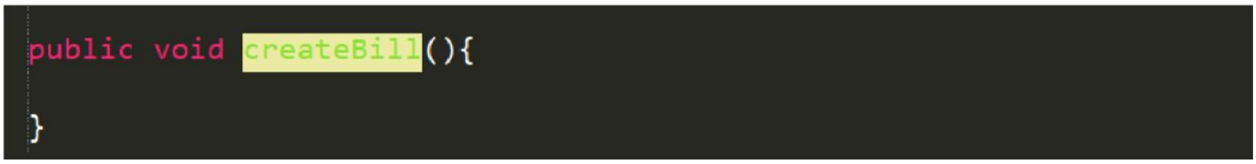
Imported classes without use.

Some comments that are not obligatory.



```java
} catch (IOException e) {
    e.printStackTrace();
}
```

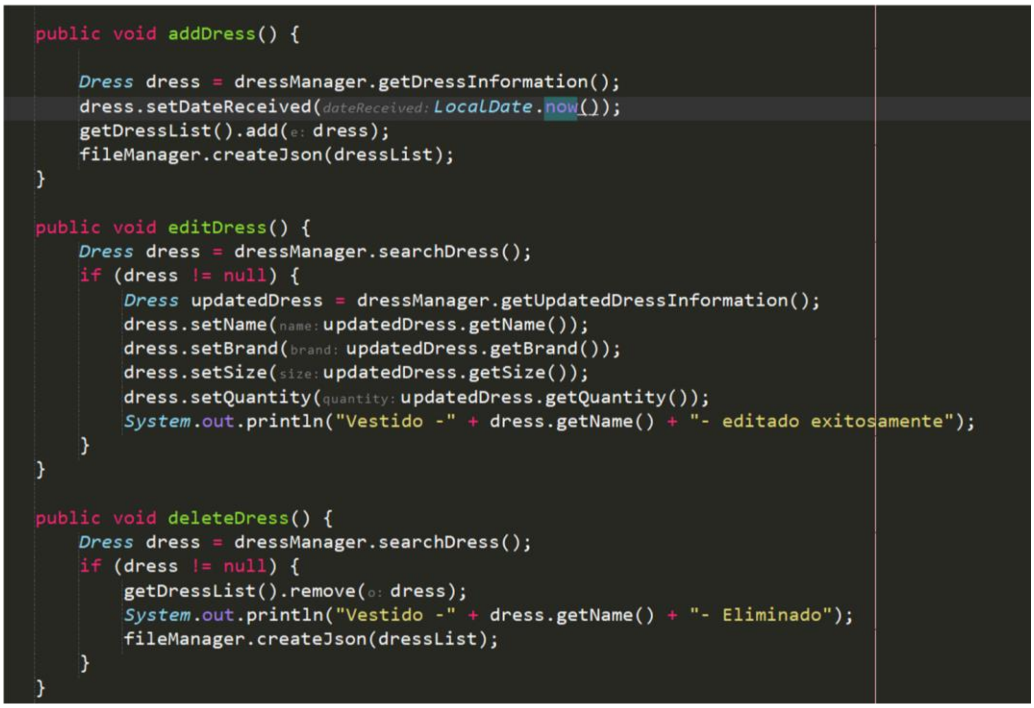
Is better to put a message that the customer can understand.



```
public void createBill(){

}
```

Useless function.

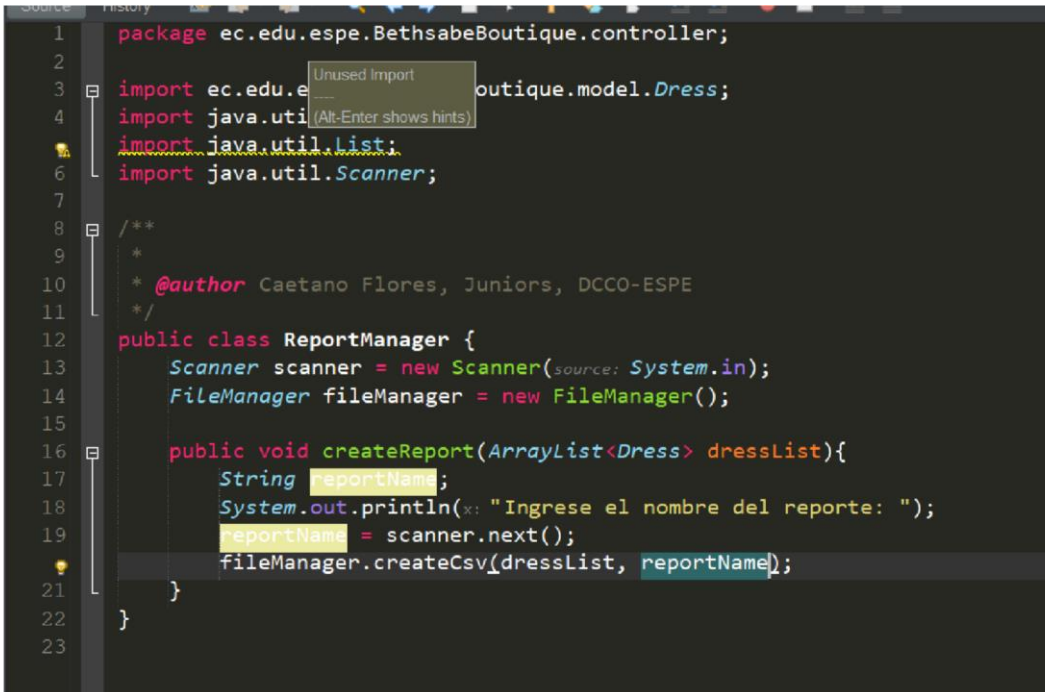
InventoryManager



```
public void addDress() {

    Dress dress = dressManager.getDressInformation();
    dress.setDateReceived(dateReceived: LocalDate.now());
    getDressList().add(e: dress);
    fileManager.createJson(dressList);
}

public void editDress() {
    Dress dress = dressManager.searchDress();
    if (dress != null) {
        Dress updatedDress = dressManager.getUpdatedDressInformation();
        dress.setName(name: updatedDress.getName());
        dress.setBrand(brand: updatedDress.getBrand());
        dress.setSize(size: updatedDress.getSize());
        dress.setQuantity(quantity: updatedDress.getQuantity());
        System.out.println("Vestido -" + dress.getName() + "- editado exitosamente");
    }
}

public void deleteDress() {
    Dress dress = dressManager.searchDress();
    if (dress != null) {
        getDressList().remove(o: dress);
        System.out.println("Vestido -" + dress.getName() + "- Eliminado");
        fileManager.createJson(dressList);
    }
}
```

Declaration of a Dress object multiple times in the same class instead of declaring one global.

ReportManager

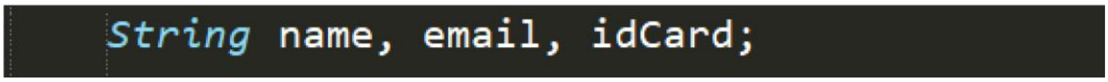


```
package ec.edu.espe.BethsabeBoutique.controller;

import ec.edu.e                   outique.model.Dress;
import java.uti
import java.util.List;
import java.util.Scanner;

/**
 *
 * @author Caetano Flores, Juniors, DCCO-ESPE
 */
public class ReportManager {
    Scanner scanner = new Scanner(source: System.in);
    FileManager fileManager = new FileManager();

    public void createReport(ArrayList<Dress> dressList){
        String reportName;
        System.out.println(x: "Ingrese el nombre del reporte: ");
        reportName = scanner.next();
        fileManager.createCsv(dressList, reportName);
    }
}
```

Unused Import
(Alt-Enter shows hints)

Unused import.

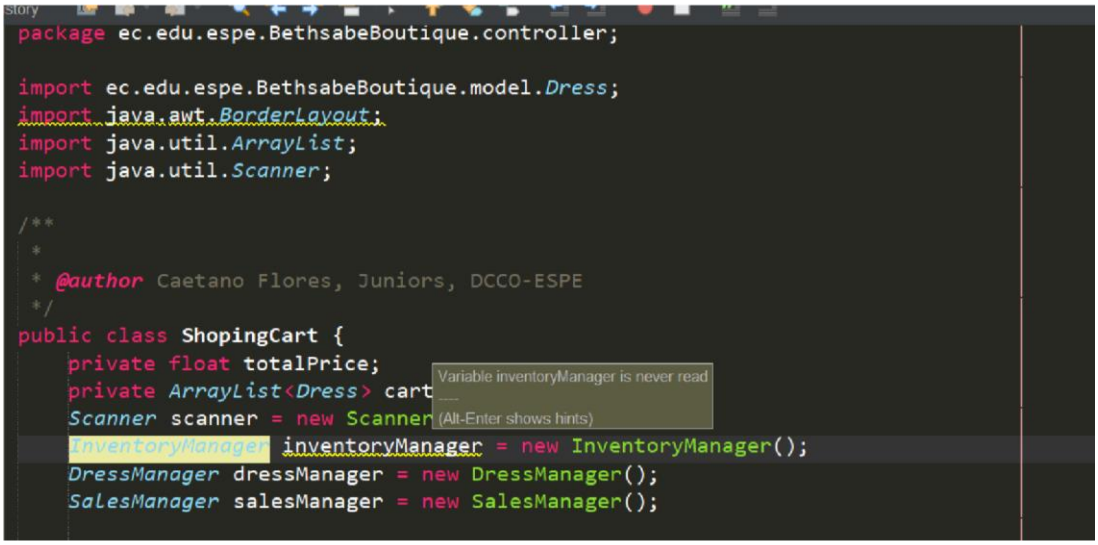
SalesManager



```
    String name, email, idCard;
```

More than one variable declared in the same line.

ShoppingCart

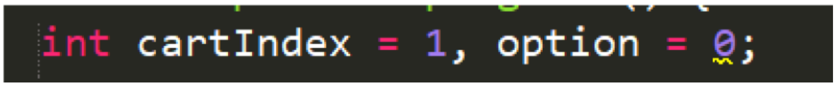


```
package ec.edu.espe.BethsabeBoutique.controller;

import ec.edu.espe.BethsabeBoutique.model.Dress;
import java.awt.BorderLayout;
import java.util.ArrayList;
import java.util.Scanner;

/**
 *
 * @author Caetano Flores, Juniors, DCCO-ESPE
 */
public class ShopingCart {
    private float totalPrice;
    private ArrayList<Dress> cart
    Scanner scanner = new Scanner
    InventoryManager inventoryManager = new InventoryManager();
    DressManager dressManager = new DressManager();
    SalesManager salesManager = new SalesManager();
```

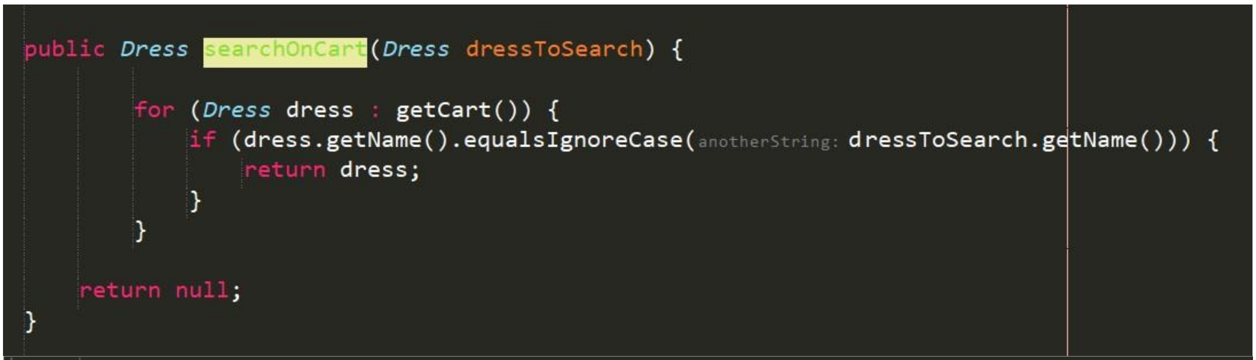
Variable inventoryManager is never read
(Alt-Enter shows hints)

Class & Import without use.

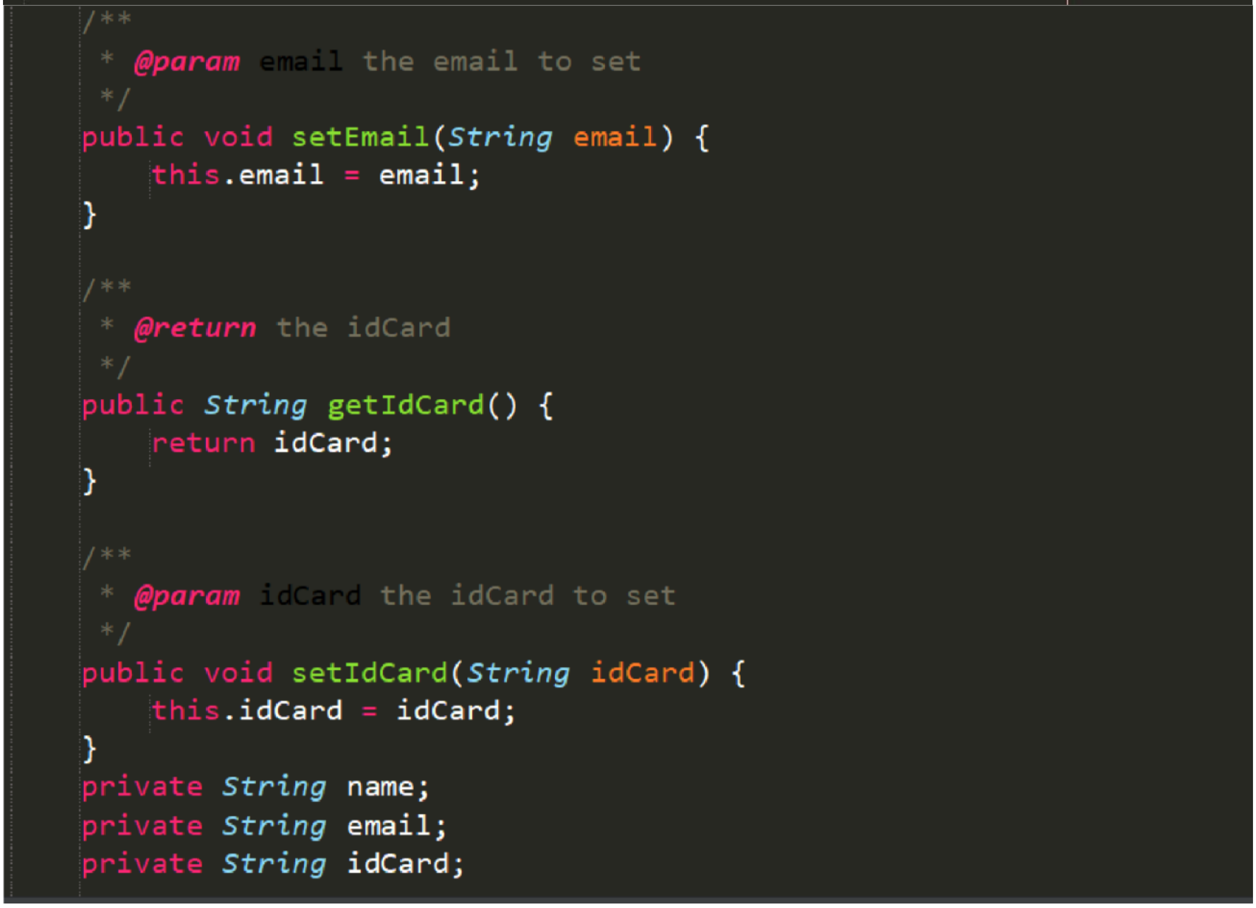
Bad Indentation.



```
int cartIndex = 1, option = 0;
```



```java
public Dress searchOnCart(Dress dressToSearch) {

    for (Dress dress : getCart()) {
        if (dress.getName().equalsIgnoreCase(anotherString: dressToSearch.getName())) {
            return dress;
        }
    }

    return null;
}
```



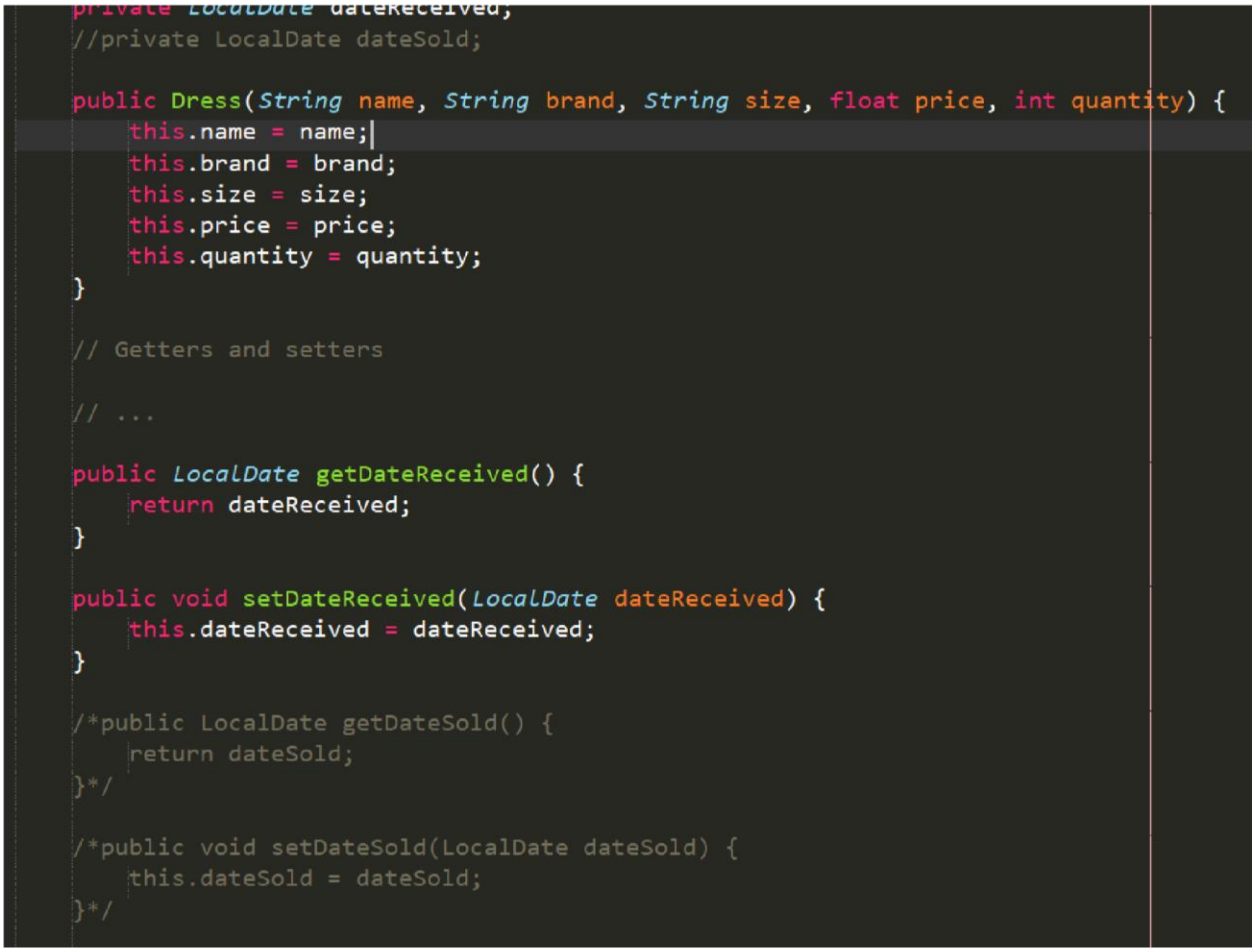
```java
    /**
     * @param email the email to set
     */
    public void setEmail(String email) {
        this.email = email;
    }

    /**
     * @return the idCard
     */
    public String getIdCard() {
        return idCard;
    }

    /**
     * @param idCard the idCard to set
     */
    public void setIdCard(String idCard) {
        this.idCard = idCard;
    }
    private String name;
    private String email;
    private String idCard;
```

More than 1 variable declared in the same line.

Customer

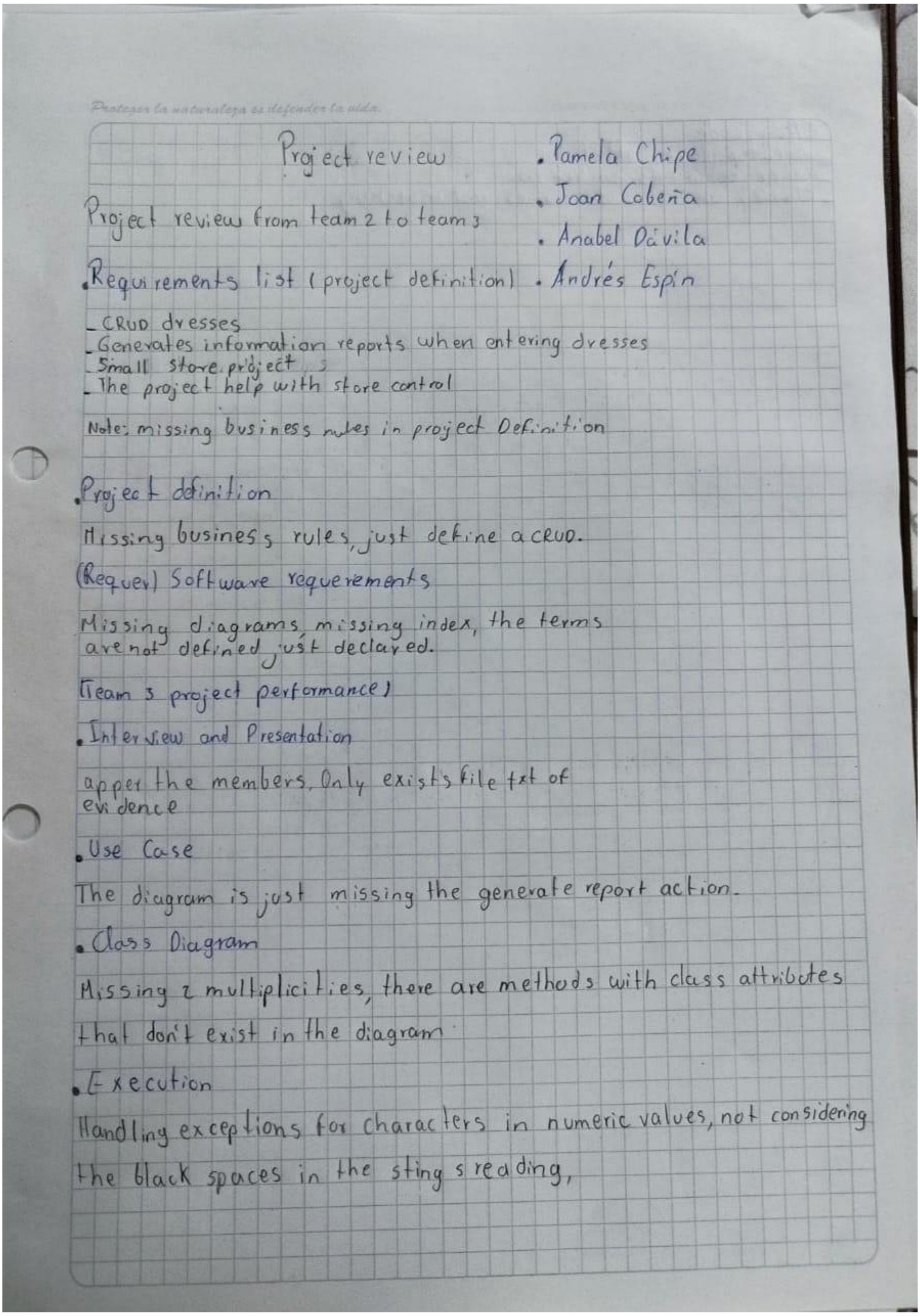
Declaration of attributes after the methods.

Dress



```
    private LocalDate dateReceived;
    //private LocalDate dateSold;

    public Dress(String name, String brand, String size, float price, int quantity) {
        this.name = name;
        this.brand = brand;
        this.size = size;
        this.price = price;
        this.quantity = quantity;
    }

    // Getters and setters

    // ...

    public LocalDate getDateReceived() {
        return dateReceived;
    }

    public void setDateReceived(LocalDate dateReceived) {
        this.dateReceived = dateReceived;
    }

    /*public LocalDate getDateSold() {
        return dateSold;
    }*/

    /*public void setDateSold(LocalDate dateSold) {
        this.dateSold = dateSold;
    }*/
```

Commented code & unnecessary comment about getters and setters.



Project review

- Pamela Chipe
- Joan Cobeña
- Anabel Dávila
- Andrés Espín

Project review from team 2 to team 3

- Requirements list (project definition)

_CRUD dresses
_Generates information reports when entering dresses
_Small store project
_The project help with store control

Note: missing business rules in project Definition

- Project definition

Missing business rules, just define a CRUD.

(Requer) Software requerements

Missing diagrams, missing index, the terms are not defined just declared.

(Team 3 project performance)

- Interview and Presentation

apper the members, Only exists file txt of evidence

- Use Case

The diagram is just missing the generate report action.

- Class Diagram

Missing 2 multiplicities, there are methods with class attributes that don't exist in the diagram

- Execution

Handling exceptions for characters in numeric values, not considering the black spaces in the sting s reading,

Clean code

- unencapsulated attributes
- Other comments on line 20
- Classes and attribute names not matching