Software Engineering
10636312

Masa Rateb Abu Aisheh
12027844

Nancy Fayez Sawalmeh
12029645

Cleaning Service System

In our project we made a 10 features

Nancy Sawalmeh

1- record full information for customers (e.g. name , phone, address, etc..).

2- generate invoice for customer including delivery information price , address , items he

give to clean.

3- Give a discount option to customers based on your design option, for example if the

customer frequently uses the service with an amount of money >400 NIS, he can get a

10% discount. Please feel free to set an acceptable discount rate.

4- Generate statistics about, for example, total delivered items , total cash , total paid, total

debts.

5- and added new feature for log in

Masa Abu Aisheh

1- record full information for the product (e.g. name, picture, description[dimension,material) , is required special treatment, etc..)

2- generate customized reports about business.

3- Track the orders and the status [waiting, in treatment, complete]

4- Distribute the orders on the available workers.

5- Notify the customer by sending email when the order is complete.

The system include 3 menus

admins menu:

1. prodect

2. customer

3. worker

4.reports_about_business

5.all order

6.logout

users menu:

1. Add prodect in shoppingcart

2.show all prodect in shoppingcart

3. Remove prodect in shoppingcar

4.check out

5.Messages

6.logout

workers menu:

1. show order

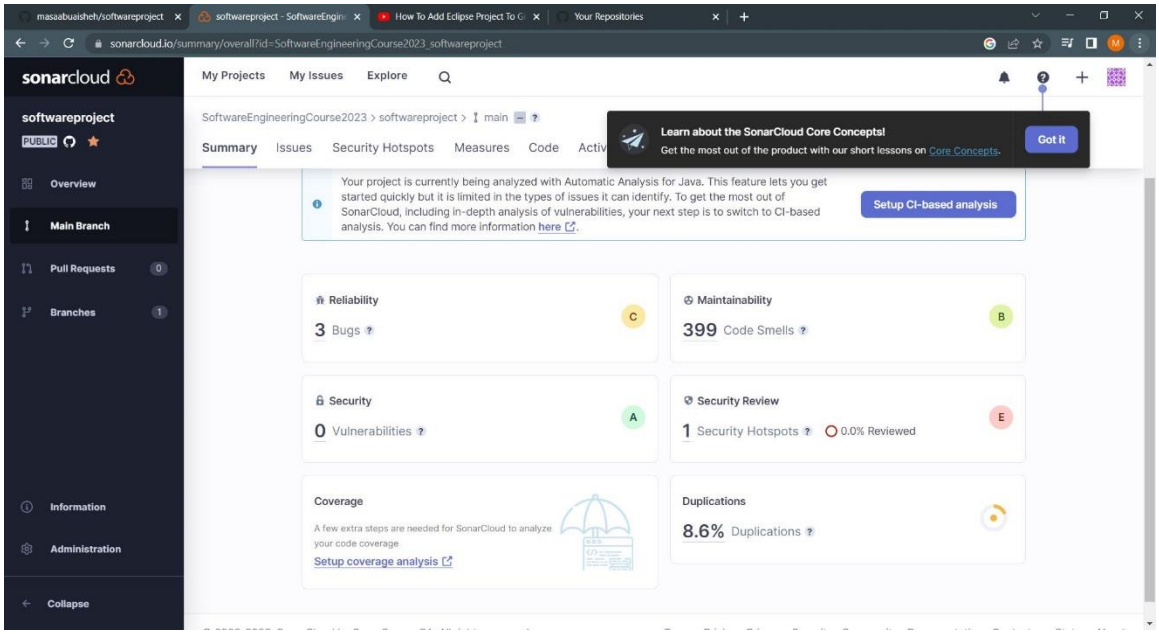
2. select situation

3. logou

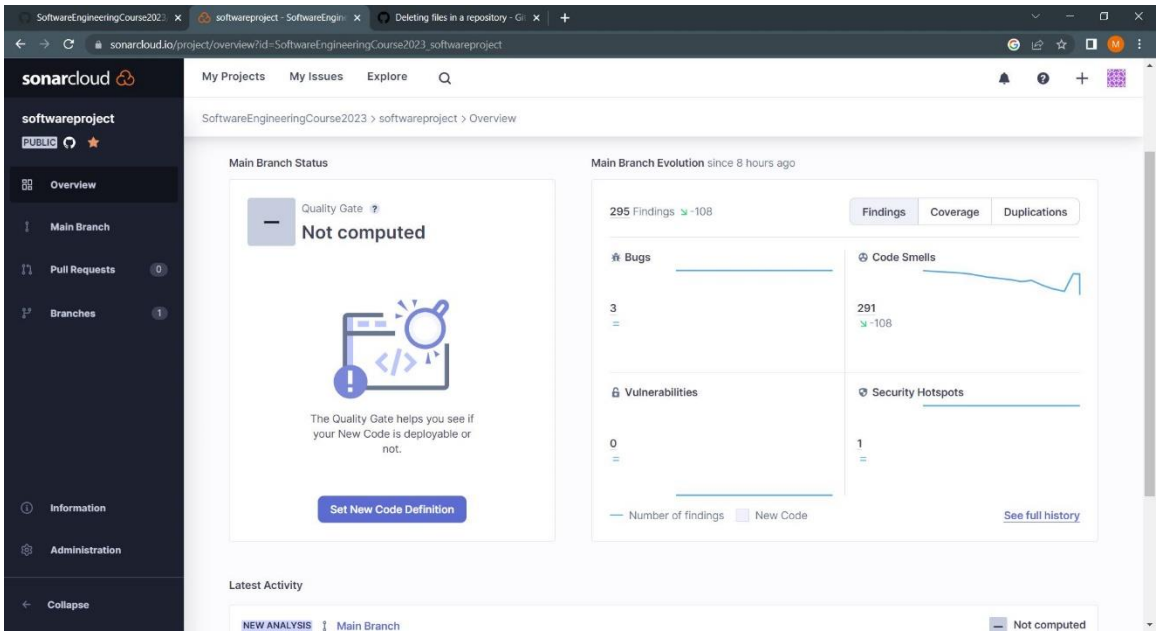
We have established a new organization on Github with the purpose of transferring our repository to it. To ensure the quality and integrity of our code, we have integrated our organization with SonarCloud, a platform that runs a comprehensive set of rules covering various aspects of code, including maintainability, reliability, and security. This integration allows us to automatically check our code against these rules on each pull request, thus improving the overall quality and security of our codebase.

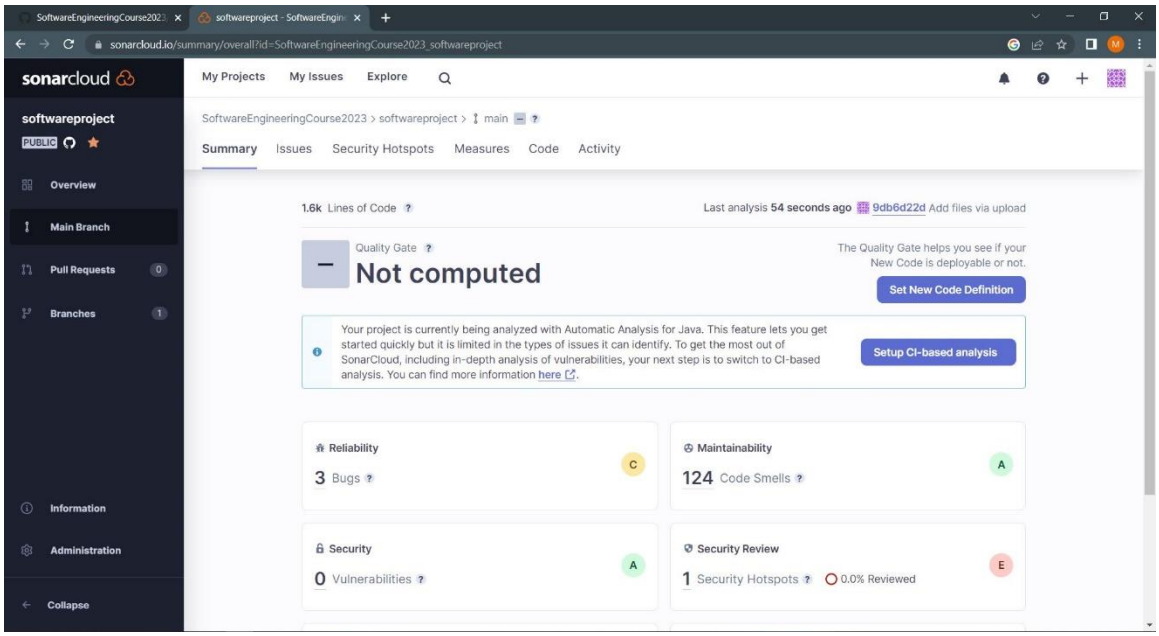
In the first analysis

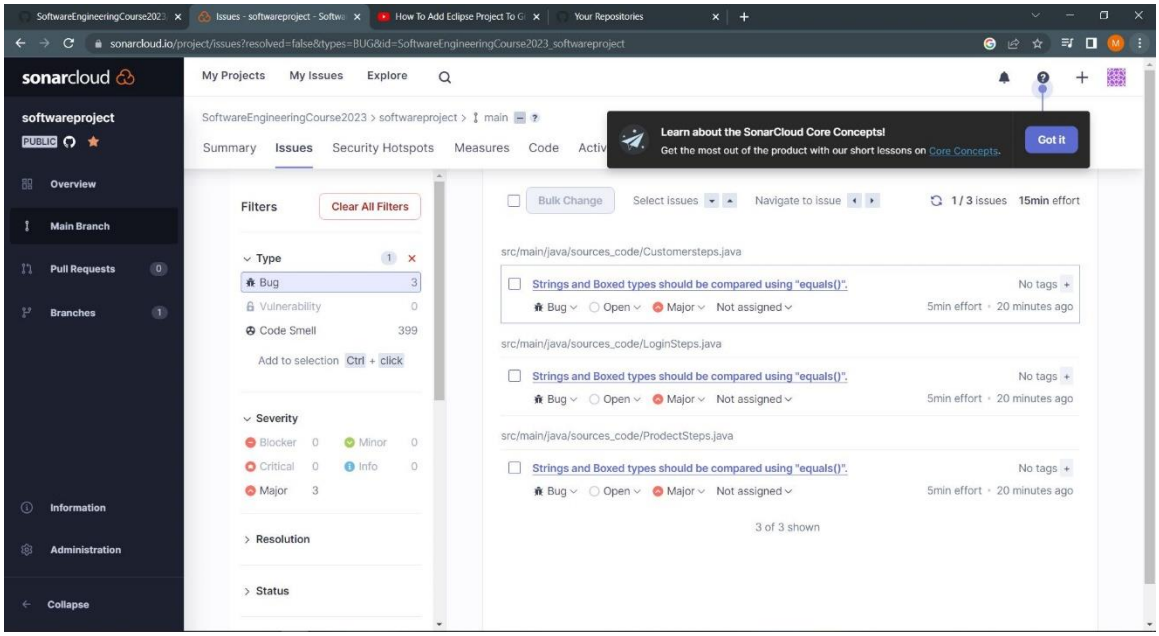
Main Branch Evolution



Then we refactoring our project to decrease the number of code smells and bugs







We refactoring this bug
For example :

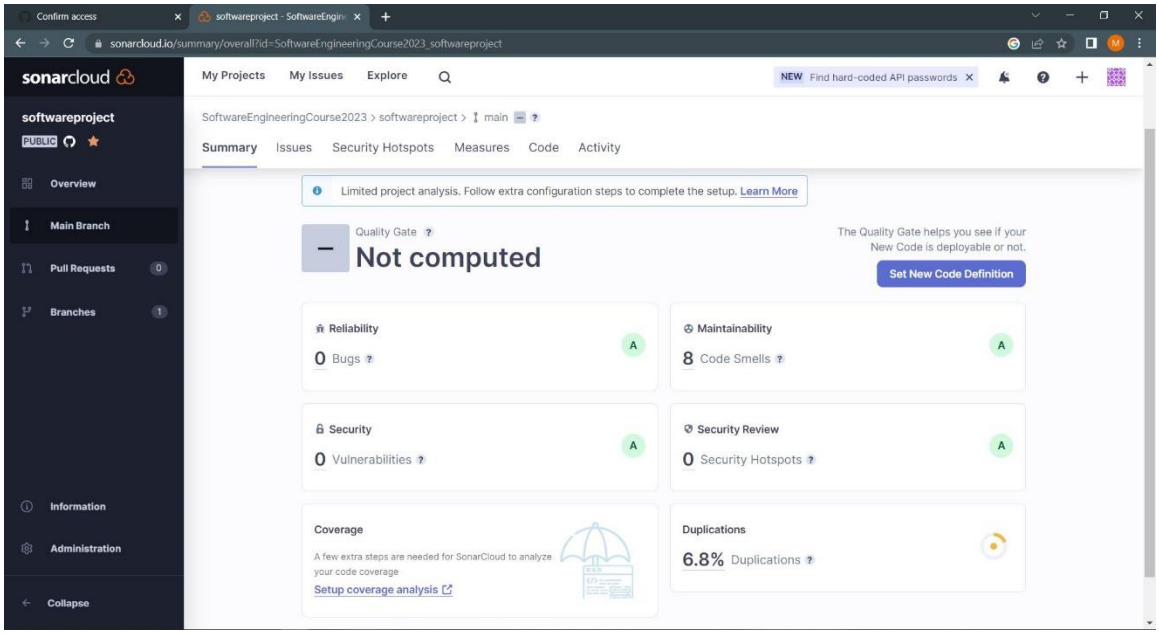
Before:

```
220
221    public static void  usernotfound () {
222
223        if(isLoginonuser==true) {
224            x=-1;
225        }
226
227
228    }
```

After:

```
228
229    public static void  usernotfound () {
230
231        if(isLoginonuser) {
232            setxnan(-1);
233        }
234
235
236    }
237
```

We completed cleaning code like this method until we come to this result



Some example for things we refactored

1- System.Out.println(“”);

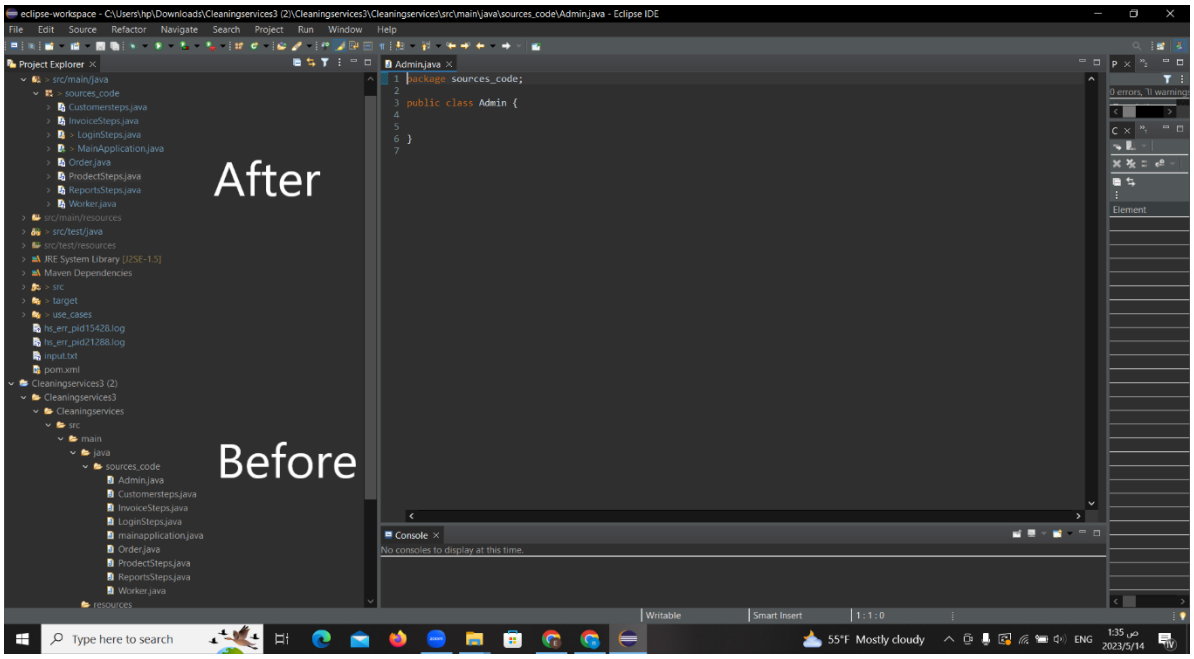
Before:

```
102
103     public  void  addnewlogin() {
104         //boolean e= cus.getissave();
105         //System.out.print(e);1274
106         if(Customersteps.issave) {
107          System.out.println("Enter Username : ");
108         input3 = new Scanner(System.in);
109         String username=input3.nextLine();
110
```

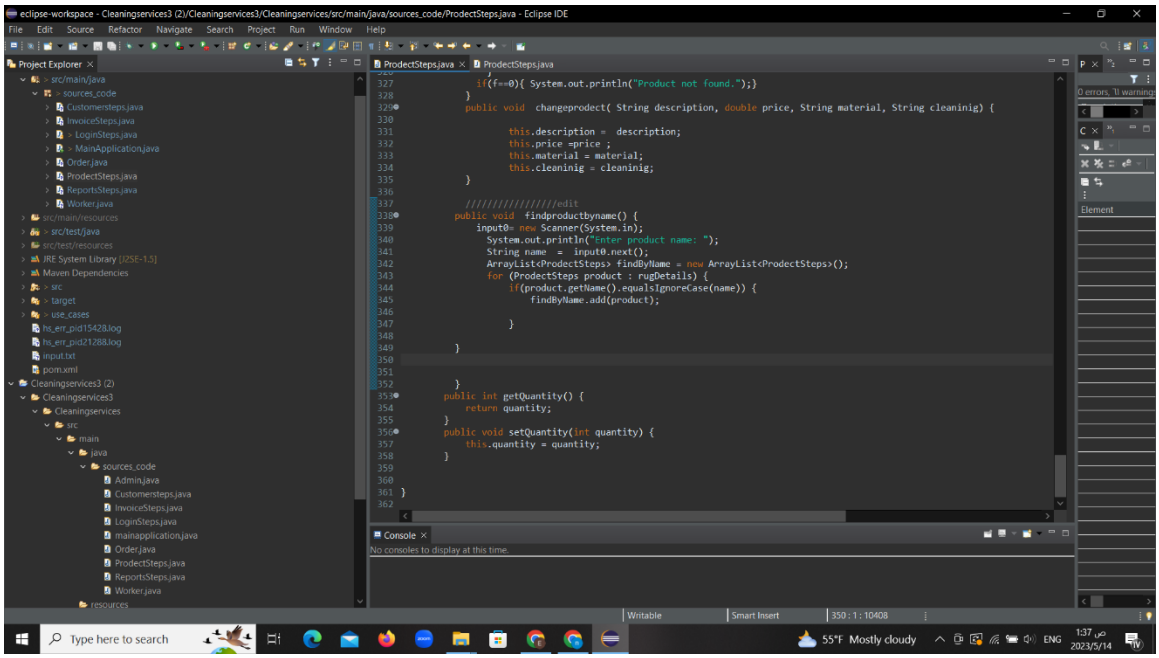
After:

```
110
111    public  void  addnewlogin() {
112
113        if(Customersteps.isIssave()) {
114              LOGGER.log(Level.INFO,"Enter Username: ");
115            }
116            input3 = new Scanner(System.in);
117        String usernameprod=input3.nextLine();
118
```

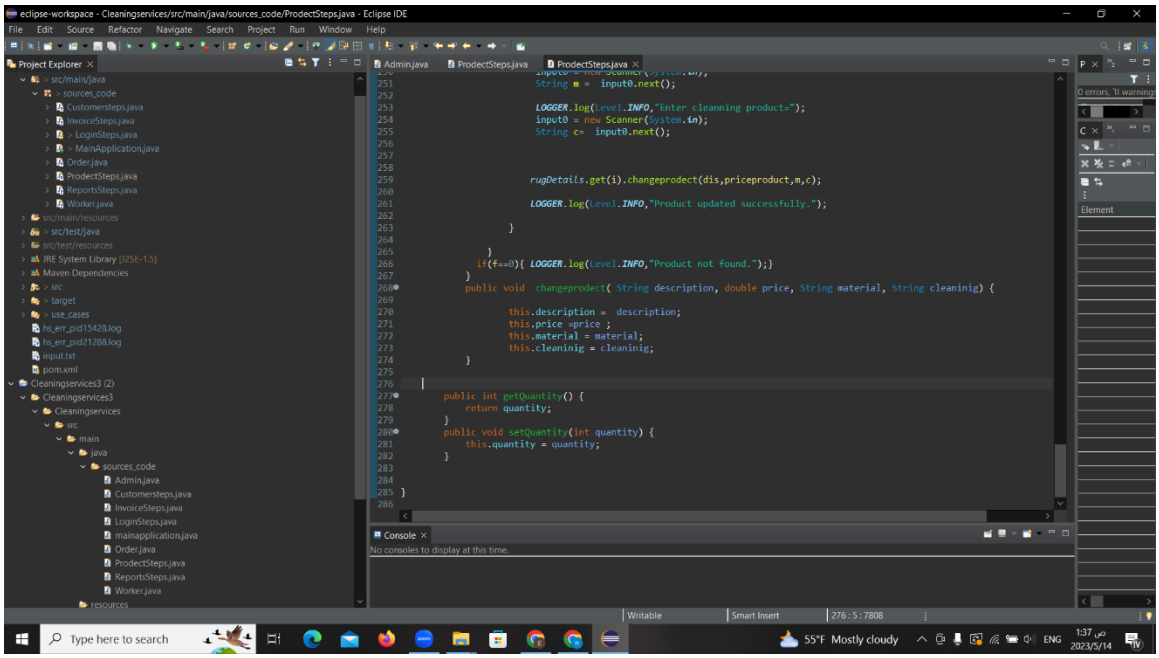
2- Remove unuse class called admin and functions



Before:



After:



3- Duplicate in code

Before:

```
        System.out.println("----------------------------------==Cleaning Services Application ==----------------------------------------------\n");
        LoginSteps.inti();
        LoginSteps.Whos_the_user();
        int x=LoginSteps.x;

           log.setx(x);


        if(log.getisLoginadmin()==true){

            System.out.println("----------------------------------==Cleaning Services Application ==----------------------------------------------\n");
            System.out.println("welacome to admin: "+LoginSteps. userslogin.get(x).getName());
            admin();
        }
        else if (log.getisLoginworker()==true) {
            System.out.println("----------------------------------==Cleaning Services Application ==----------------------------------------------\n");

            System.out.println("welacome to worker: "+LoginSteps. userslogin.get(x).getName());
            worker();

        }
        else {
            System.out.println("----------------------------------==Cleaning Services Application ==----------------------------------------------\n");
            System.out.println("welacome to customer: "+LoginSteps. userslogin.get(x).getName());
            customer();
        }
    }
```

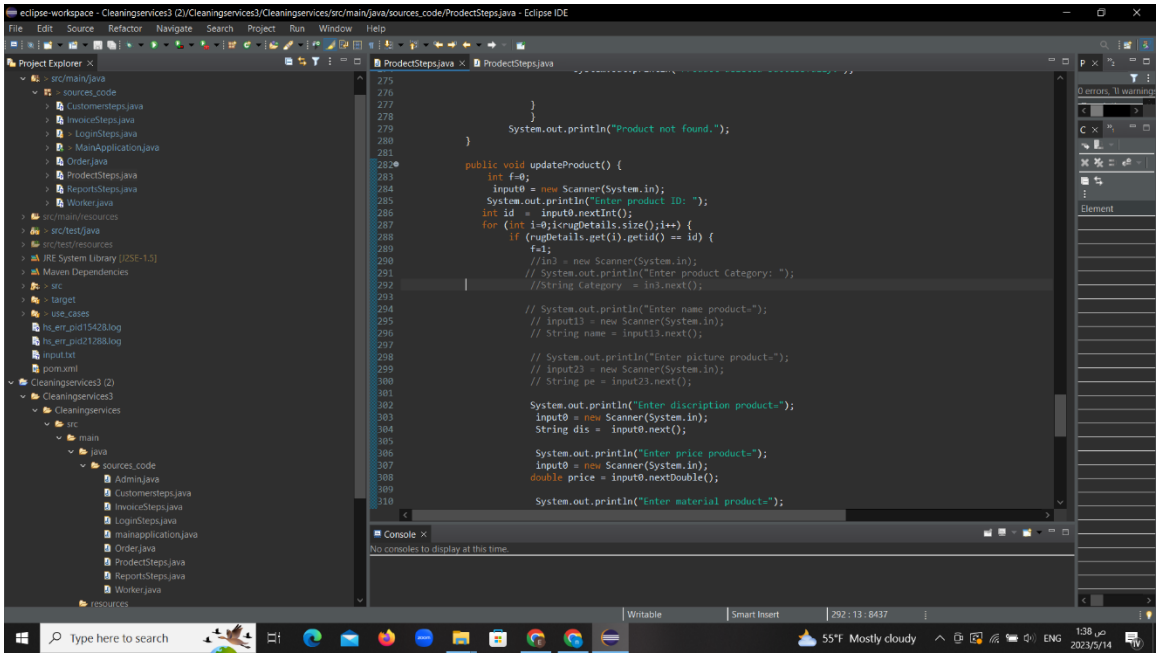
After:

```
    private static final Logger LOGGER = Logger.getLogger(MainApplication.class.getName());

    private static final String MAIN = "----------------------------------==Cleaning Services Application ==----------------------------------------------\n";
    private static final String WELCCUST =   "welacome to customer: {0}";
    private static final String WELC =    "welacome to worker:{0} ";
    private static final String WELCADMIN =     "welacome to admin:{0} ";
    private static final String DEFAULT =    "Default! ";
```

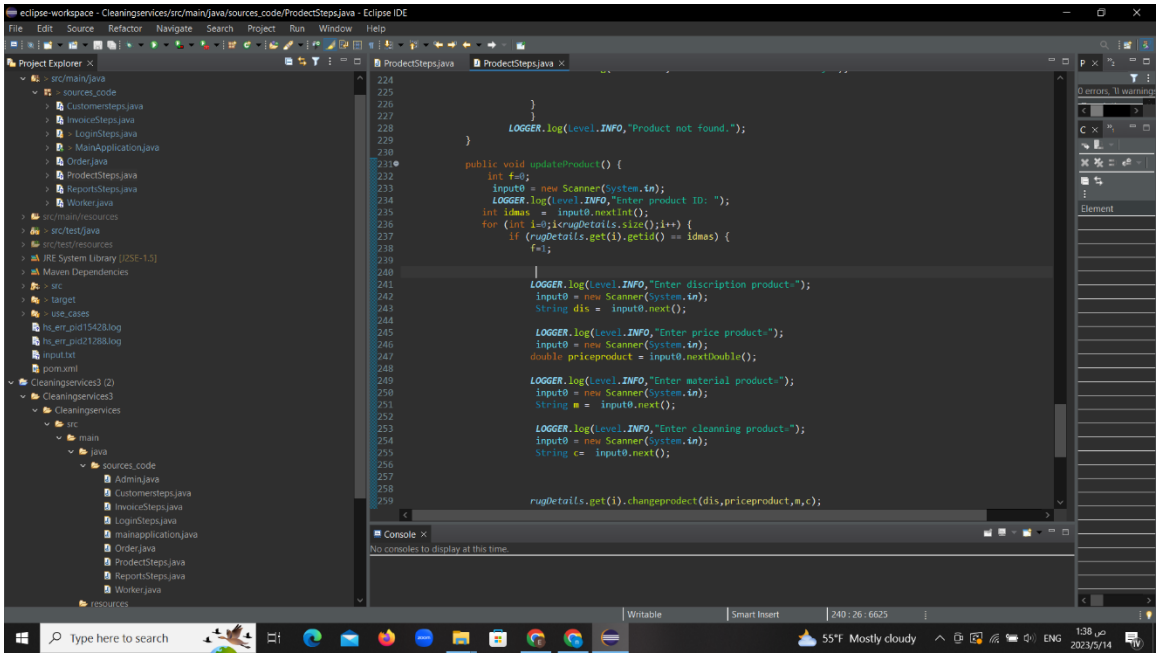
```

        LOGGER.log(Level.INFO,MAIN);
        LoginSteps.inti();
        LoginSteps.whostheuser();
        int x=LoginSteps.getxmas();

           LoginSteps.setx(x);



        if(log.getisLoginadmin()){

            LOGGER.log(Level.INFO,MAIN);
            LOGGER.log(Level.INFO,WELCADMIN,LoginSteps. getUserslogin().get(x).getName());
            admin();
        }
        else if (log.getisLoginworker()) {
            LOGGER.log(Level.INFO,MAIN);

            LOGGER.log(Level.INFO,WELC,LoginSteps. workerlogin.get(x).getName());
            worker();

        }
        else {
            LOGGER.log(Level.INFO,MAIN);
            LOGGER.log(Level.INFO,WELCCUST,LoginSteps. getUserslogin().get(x).getName());
            customer();
        }
    }
```

4- Delete the comments because it gives code smells

Before:



After:



5- Convert array list to protected or private final static (array list in public is code smell)

Before:

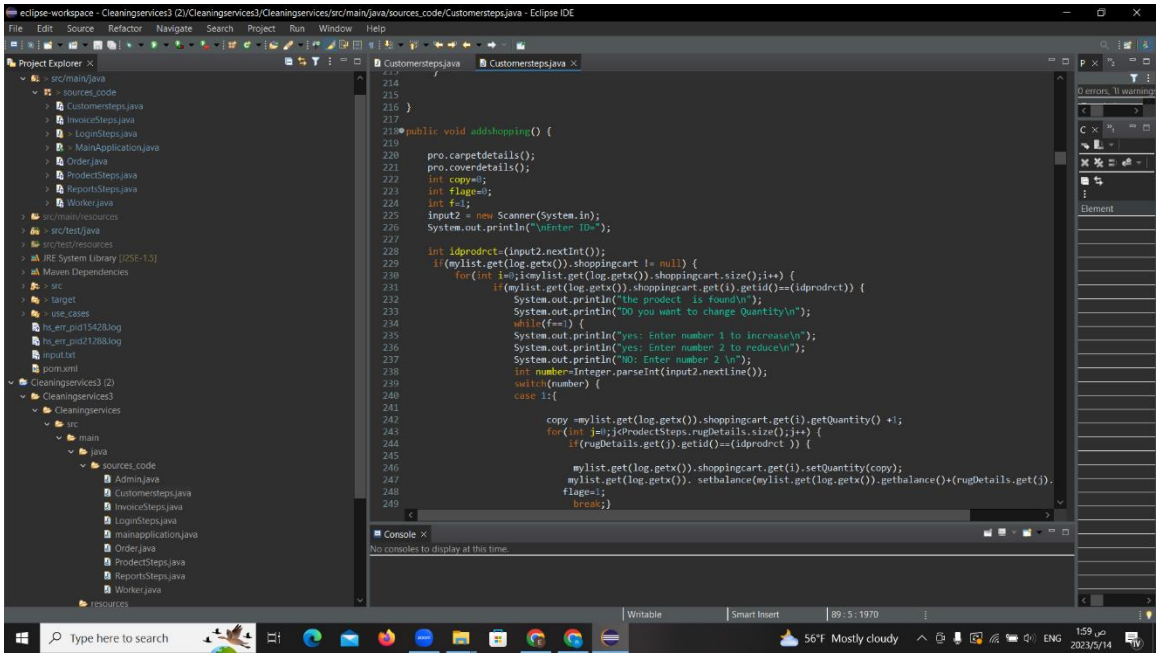
```
87      }
88
89      public ArrayList <Customersteps> mylist =new ArrayList<Customersteps>();
90
```

After:

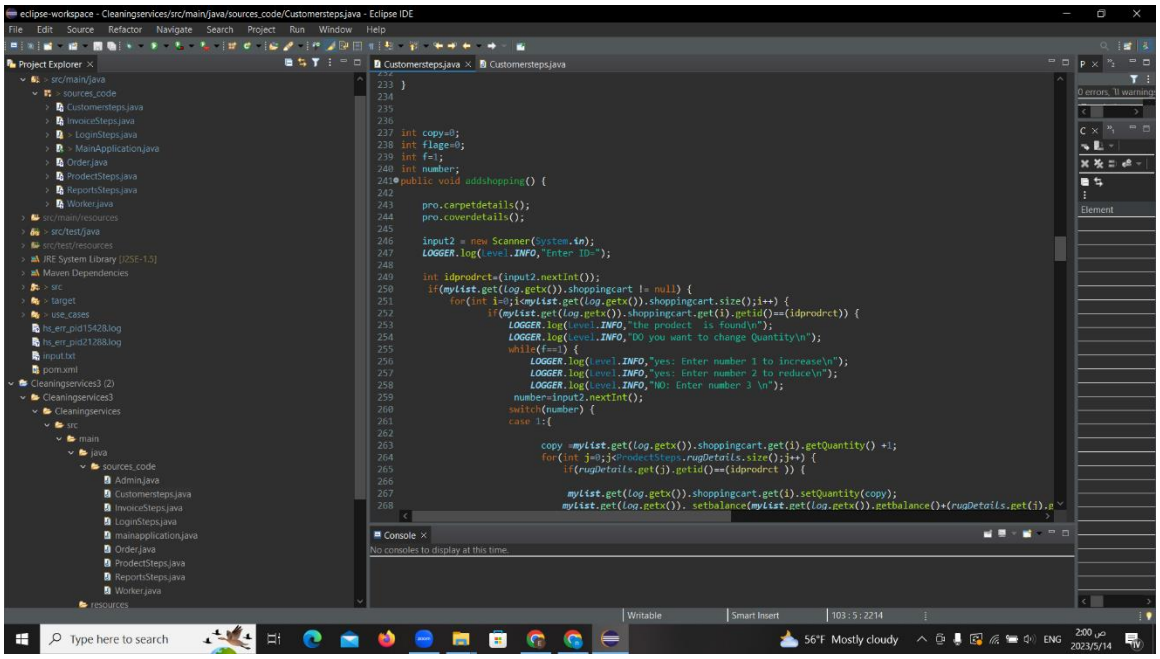
```
    private static final  List <Customersteps> mylist =new ArrayList<Customersteps>();
    public static List <Customersteps> getMylist() {
        return mylist;
    }
```

6- Decrease the number of variables in methods

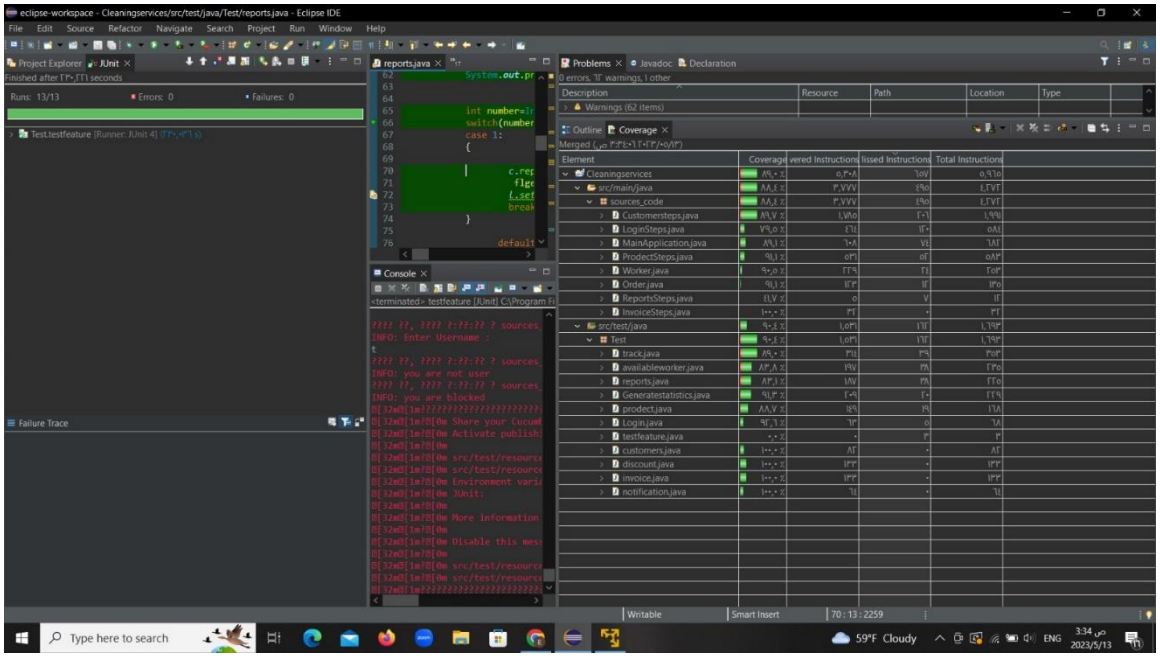
Before:



After:



After we test our project from test cases, main sources the coverage of code is 89.0



Then we build the project in Jenkins

The console output: