An-Najah National University

Faculty of Engineering

Computer Engineering Department

Preparing By

Sama' Ghazi Yasin

&

Tala Mohammad Yaseen

Supervisor

Dr.Haya Samaana

Sakancom project

In our project we made 10 features.

Tala Yaseen achievements:

1. Everything has to do with the admin (Features, Testing and source code).

2. 70% of the Tenant feature.

Sama' Yasin achievements:

1. Everything has to do with the owner (Features, Testing and The source code).

2. log in feature.

3. 30% of the Tenant feature.

Our system includes 3 menus:

- **Admin's menu**

1. View requests for advertisement
2. Accept a housing advertisement
3. Reject a housing advertisement
4. View reservations
5. Add and advertise a housing unit
6. Modify housing data
7. Exit.

- **Owner's menu**

1. Add a private residence

2. View listed residences

3. display the number of tenants and floors in specific residence

4. The names of the tenants and their contact information and the number of bathrooms and bedrooms, and balcony availability should be displayed

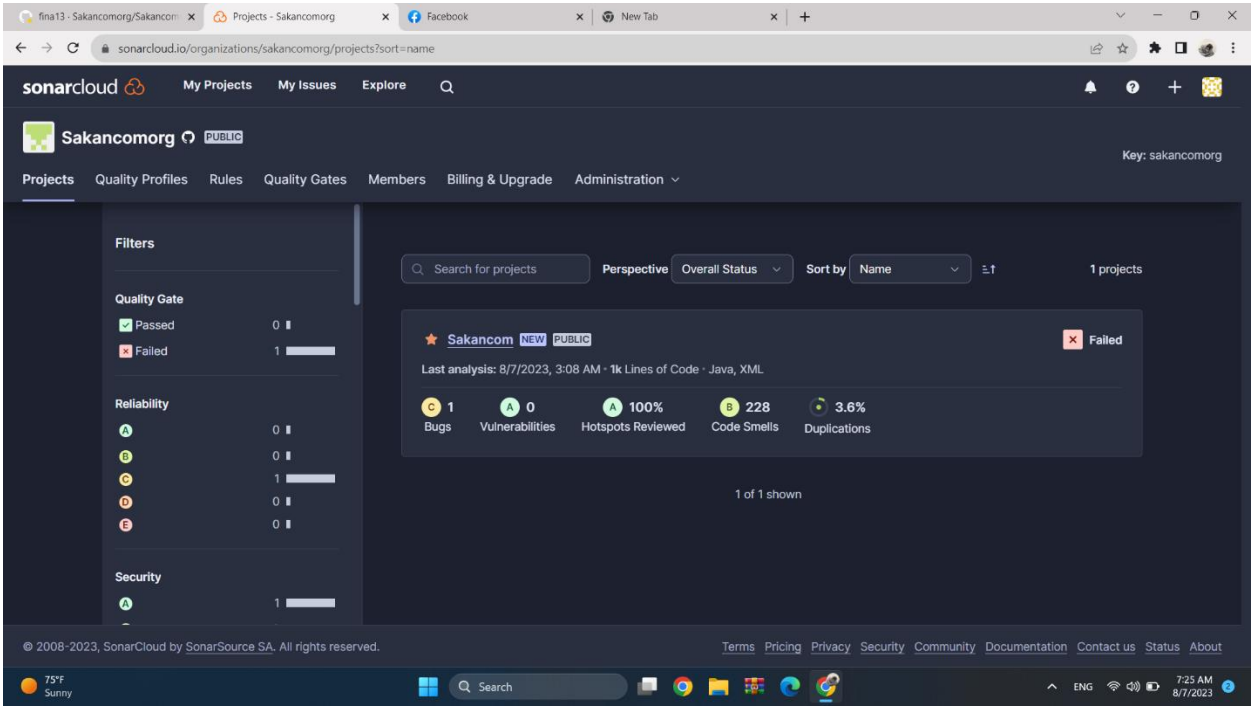
5. Exit.

- **Tenant's menu**
  1. View available housing.
  2. View housing details and pictures.
  3. Book accommodation.
  4. View information about housemates.
  5. Advertise used furniture for sale.
  6. View tenant control panel.
  7. Exit.

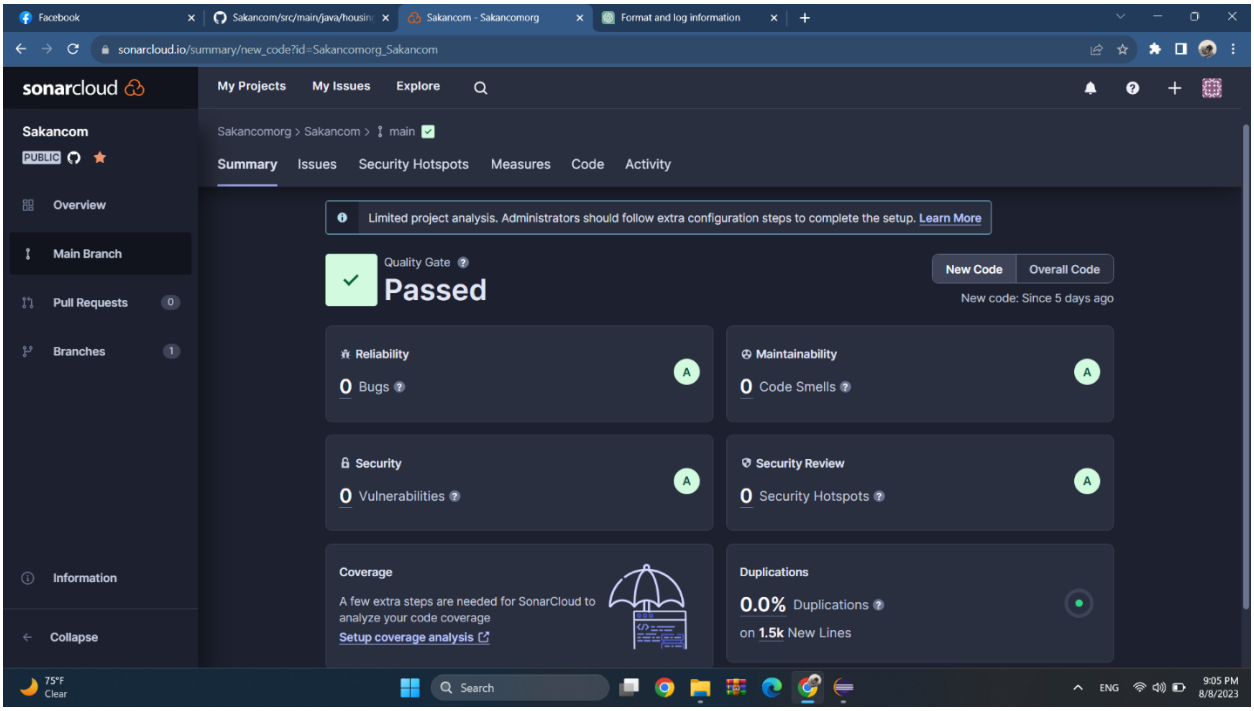
We have created an account on the GitHub (is a platform and cloud-based service for software development and version control using Git, allowing developers to store and manage their code), and we add repository to it to push our project, in order to ensure the quality of the project, and we have connected it with the sonar cloud to check the bad smells in our project, so that we can fix it, and it checks maintainability, reliability, and security for it thus improving the overall quality for project.

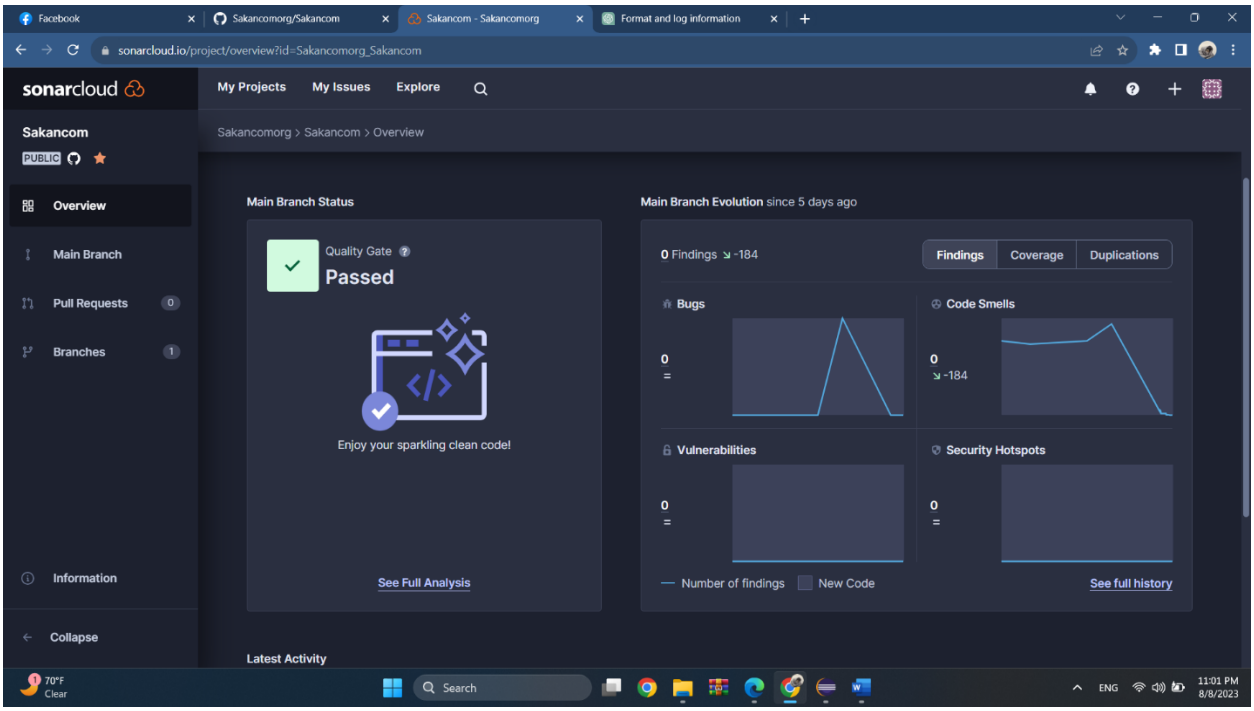
In the first analysis:

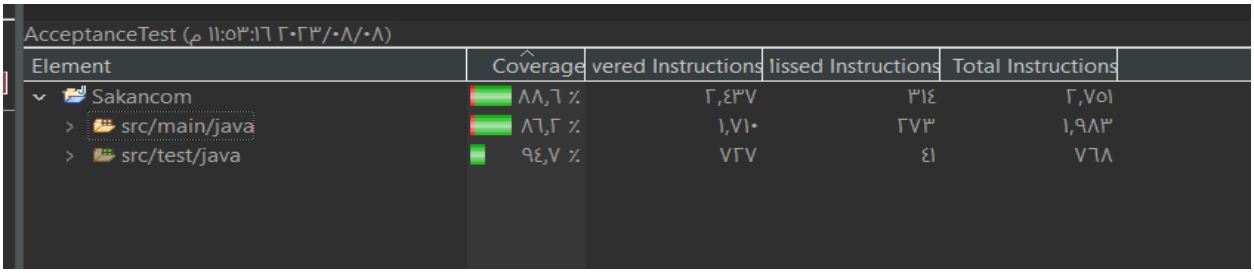
Main Branch Evolution:



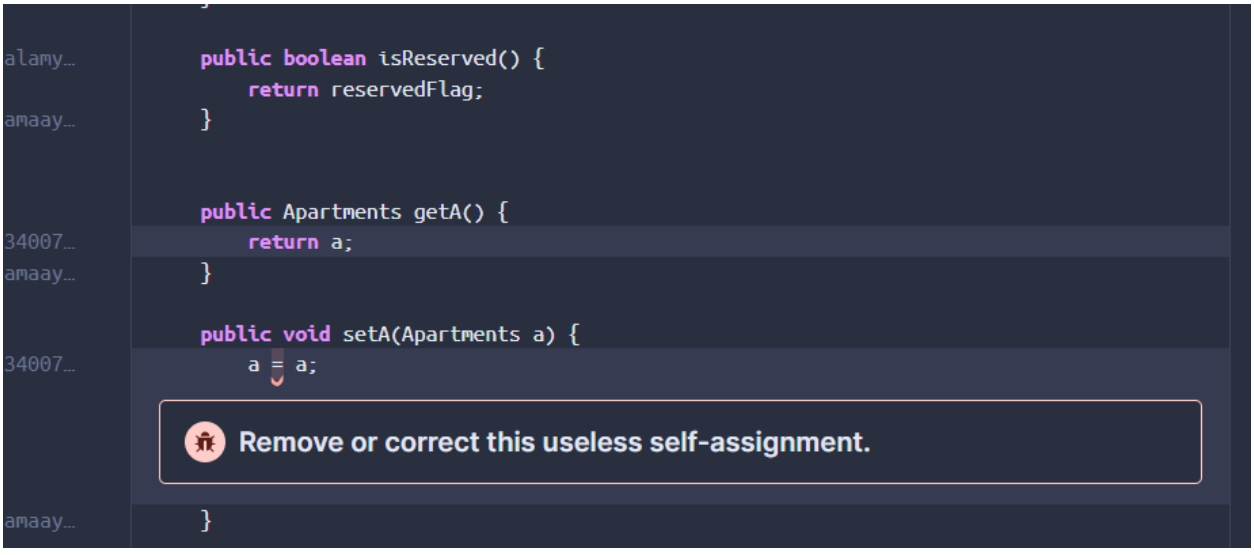
Then we refactoring our project to decrease the number of code smells and Bugs:







We have a bug:



We solve this bug like this:

```
        return a;
    }

    public void setA(Apartments a) {
        this.a = a;
    }
```

Some examples for bad smells we refactored

1. The parameters in constructer more than 7.

Before:

```
public HousingUnit(String location, String photo, double rent, int numberOfTenants, int number
                   String availableServices, boolean reservedFlag, Apartments A, boolean stud
    this.location = location;
    this.photo = photo;
    this.rent = rent;
    this.numberOfTenants = numberOfTenants;
    this.numberOfFloors = numberOfFloors;
    this.availableServices = availableServices;
    this.reservedFlag = reservedFlag;
    this.A = A;
    this.studentHouse = studentHouse;
    this.id = id;
}
```

After:

```
    public HousingUnit(LocationInfo locationInfo, Apartments apartmentInfo,
                       boolean reservedFlag, boolean studentHouse,int numoffloors,String availableSe
        this.locationInfo = locationInfo;
        this.apartmentInfo = apartmentInfo;
        this.reservedFlag = reservedFlag;
        this.studentHouse = studentHouse;
        this.numoffloors=numoffloors;
        this.availableServices=availableServices;
    }
```

2. Delete the comments because it gives code smells

Before:

```
public void initarrayvalues() {
  /*  userinfo.add(new User("admin","samaa","samaa@gmail.com","4545",1));
    userinfo.add(new User("tenant","malak","m@gmail.com","1234",1));
    userinfo.add(new User("admin","tala","tala@gmail.com","7676",1));*/
     userinfo.add(new User("admin","samaa","samaa@gmail.com","4545",1));
        userinfo.add(new User("tenant","malak","m@gmail.com","1234",1));
        userinfo.add(new User("owner","nancy","N@gmail.com","4444",1));
        userinfo.add(new User("admin","tala","tala@gmail.com","7676",1));
}
```

After:

```
    }
     public void initarrayvalues() {
          userinfo.add(new User("admin","samaa","samaa@gmail.com","4545",1));
             userinfo.add(new User("tenant","malak","m@gmail.com","1234",1));
             userinfo.add(new User("owner","nancy","N@gmail.com","4444",1));
             userinfo.add(new User("admin","tala","tala@gmail.com","7676",1));
     }

    public static ArrayList<HousingUnit> ReservationsList() {
```

3. System.Out.println(“ ”);

Before:

```
        System.out.println("========= Housing Owner Menu =========");
        System.out.println("1. Add a private residence");
        System.out.println("2. View listed residences");
        System.out.println("3. display the number of tenants and floors in specific residence ");
        System.out.println("4. The names of the tenants and their contact information and the numb
        System.out.println("5. Exit");
        System.out.print("Enter your choice: ");
```

After:

```
logger.info("========= Housing Owner Menu =========");
logger.info("1. Add a private residence");
logger.info("2. View listed residences");
logger.info("3. Display the number of tenants and floors in specific residence");
logger.info("4. Display tenant information and apartment details");
logger.info("5. Exit");
logger.info("Enter your choice: ");
```

4. Convert Array list to list

Before:

```
public ArrayList<String> getTenantNames() {
    return tenantNames;
}

public void setTenantNames(ArrayList<String> tenantNames) {
    this.tenantNames = tenantNames;
}

public ArrayList<String> getMeansOfCommunication() {
    return meansOfCommunication;
}

public void setMeansOfCommunication(ArrayList<String> meansOfCommunication) {
    this.meansOfCommunication = meansOfCommunication;
}
private int floor;
```

After:

```java
public List<String> getTenantNames() {
    return tenantNames;
}

public void setTenantNames(List<String> tenantNames) {
    this.tenantNames = tenantNames;
}

public List<String> getMeansOfCommunication() {
    return meansOfCommunication;
}

public void setMeansOfCommunication(List<String> meansOfCommunication) {
    this.meansOfCommunication = meansOfCommunication;
}
```

5. Conver public variable to private with setter and getter

Before:

```java
package housing.unit;

public class User {
    public String name;
    public String emaill;
    public String pa;
    String role;
    public int s;

    User(String role,String name ,String emaill , String pa,int s){
        this.name=name;
        this.emaill=emaill;
        this.pa=pa;
        this.s=s;
        this.role=role;
    }

}
```

After:

```java
package housing.unit;

public class User {
    private String name;
    private String emaill;
    private String pa;
    String role;
    private int s;

    User(String role,String name ,String emaill , String pa,int s){
        this.name=name;
        this.emaill=emaill;
        this.pa=pa;
        this.s=s;
        this.role=role;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getEmaill() {
        return emaill;
    }

    public void setEmaill(String emaill) {
        this.emaill = emaill;
    }

    public String getPa() {
        return pa;
```

6. Add private constructer to hide implicit one

   Before:

```java
import java.util.ArrayList;

public class HouseExist {

    Add a private constructor to hide the implicit public one.

    public static boolean cheack(HousingUnit enteredhouse, ArrayList<HousingUnit>
HousingUnitList) {
        // TODO Auto-generated method stub
        for (HousingUnit house : HousingUnitList) {
            if(house.equals(enteredhouse))
                return true;
        }

        return false;
    }
```

After:

```java
package housing.unit;

import java.util.List;

public class HouseExist {
    private HouseExist() {
    }

    public static boolean cheack(HousingUnit enteredhouse,List<HousingUnit> housingunitlist) {
        for (HousingUnit house : housingunitlist) {
             if(house.equals(enteredhouse)) {
                   return true;
             }
     }

        return false;
    }

}

```

7. built-in formatting to construct the argument for the logger.info

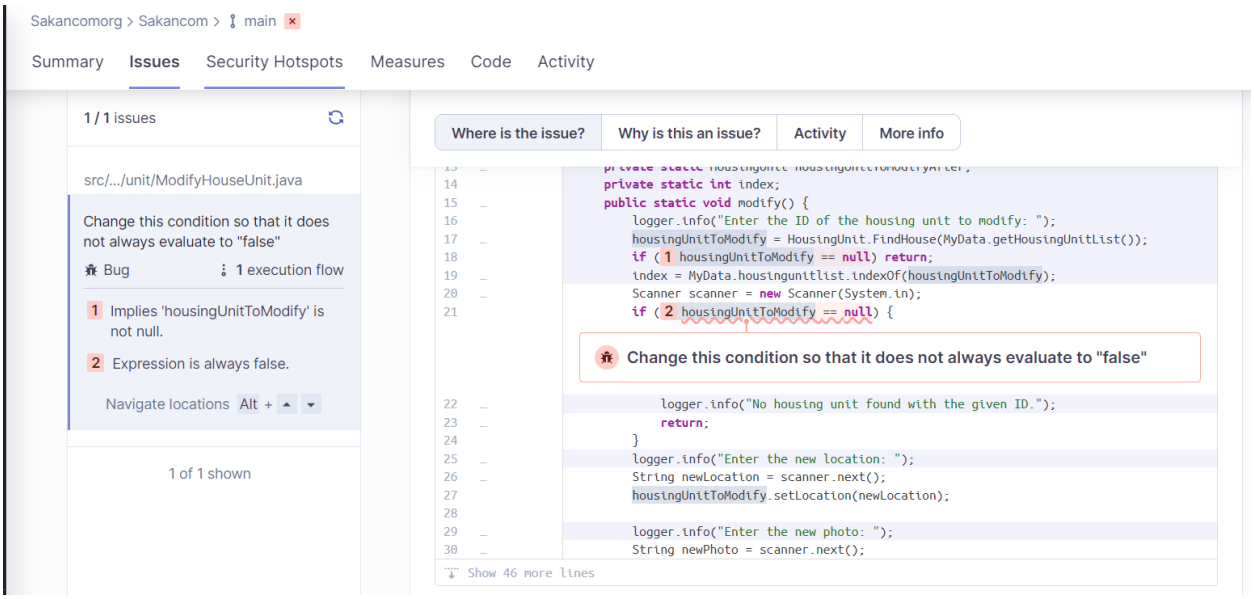
Before:

```java
            for (int i = 0; i < tenantNames.size(); i++) {
                logger.info("Tenant Name: " + tenantNames.get(i));
```

After:

```java
        for (int i = 0; i < tenantNames.size(); i++) {
               String tenantInfo = String.format("Tenant Name: %s, Means of Communication: %s",
                                                 tenantNames.get(i), meansOfCommunication.get(i));
               logger.info(tenantInfo);
           }
```

8. Before:

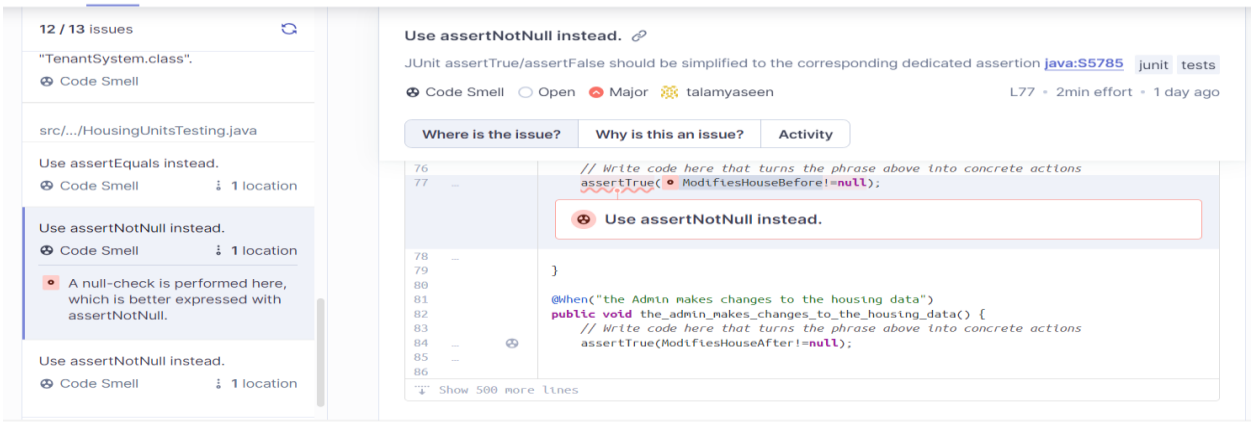


After:

```
logger.info("Enter the ID of the housing unit to modify: ");
housingUnitToModify = HousingUnit.FindHouse(MyData.getHousingUnitList());
if (housingUnitToModify == null) {
         logger.info("No housing unit found with the given ID.");
        return;
}

index = MyData.housingunitlist.indexOf(housingUnitToModify);
Scanner scanner = new Scanner(System.in);
```

9. Before:

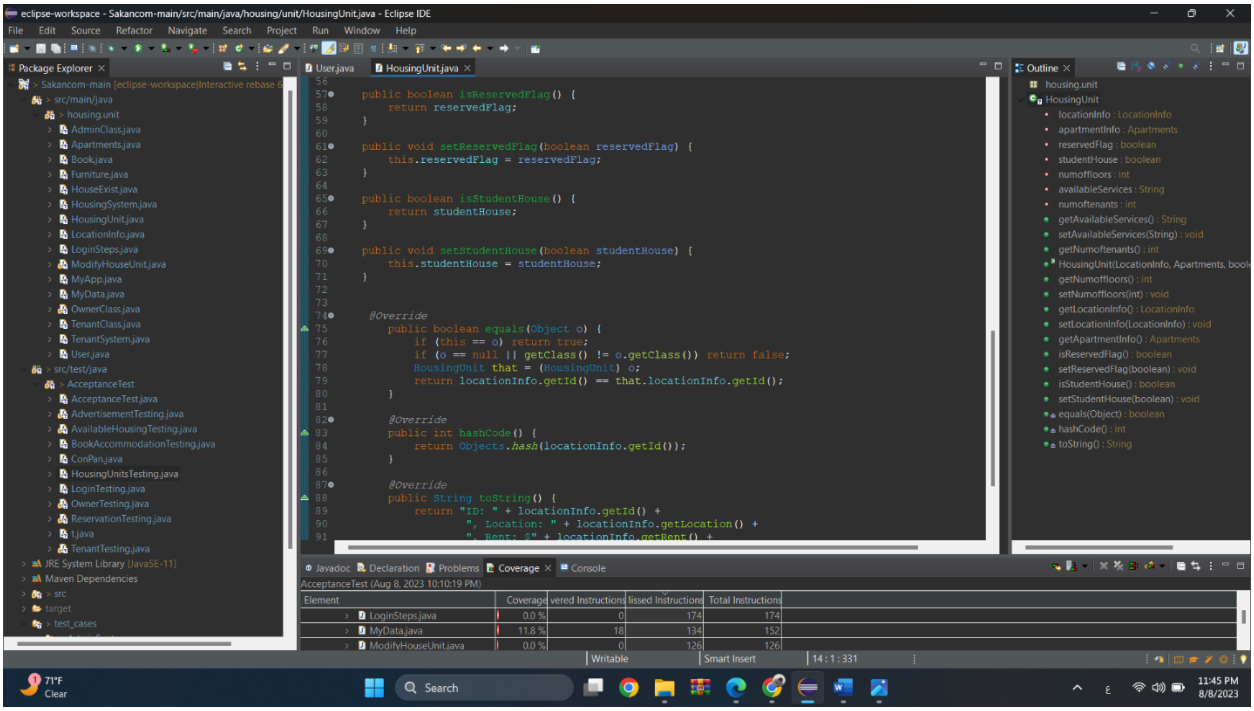


After:

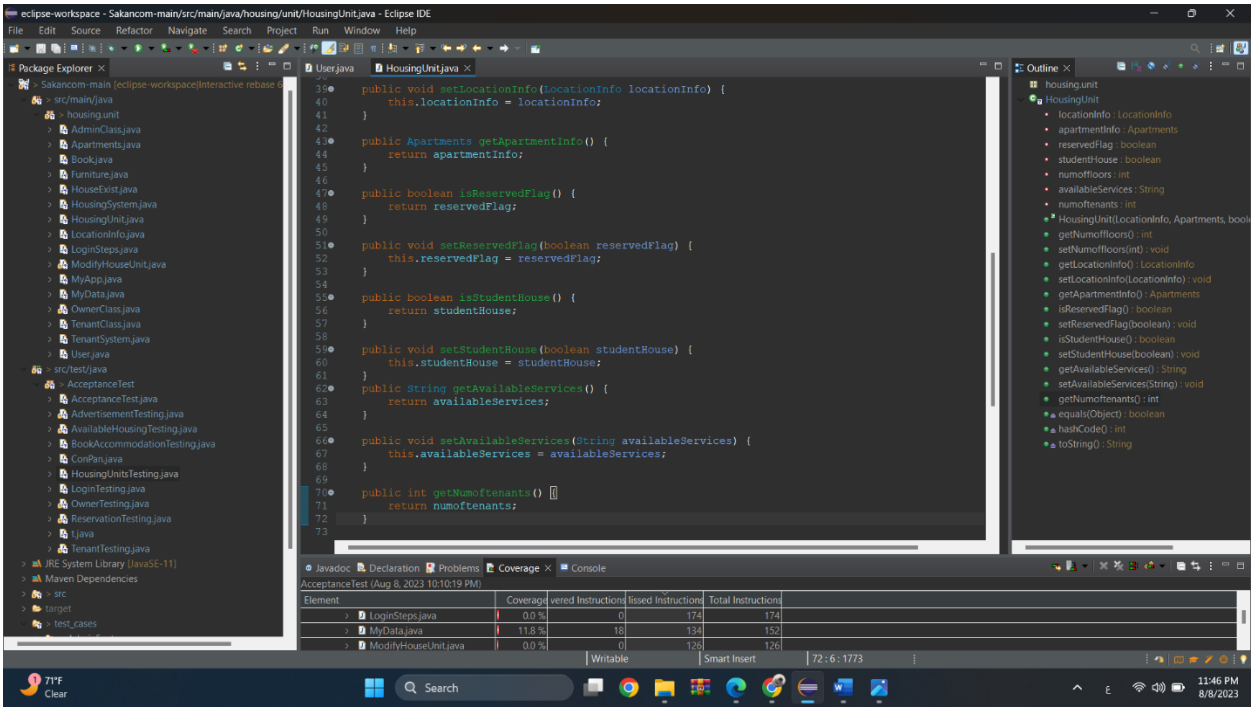
```
76
77      ModifyHouseUnit.modifyAdmin();
78       this.ModifiesHouseBefore= ModifyHouseUnit.getHousingUnitToModify();
79      assertNotNull(ModifiesHouseBefore);
80
81  }
82
```

We remove duplicate in code :

Before:



After:



Then we build the project in Jenkins

The console output:

```
[INFO] argLine set to -
javaagent:C:\\Windows\\system32\\config\\systemprofile\\.m2\\repository\\org\\jacoco\\org.jacoco.agent\\0.8.7\\org.jacoco.agent-0.8.7-
runtime.jar=destfile=C:\\ProgramData\\Jenkins\\.jenkins\\workspace\\Sakancom\\target\\jacoco.exec
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ Sakancom ---
[WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory C:\ProgramData\Jenkins\.jenkins\workspace\Sakancom\src\main\resources
[INFO]
[INFO] --- compiler:3.11.0:compile (default-compile) @ Sakancom ---
[INFO] Changes detected - recompiling the module! :source
[WARNING] File encoding has not been set, using platform encoding Cp1252, i.e. build is platform dependent!
[INFO] Compiling 16 source files with javac [debug target 11] to target\classes
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  5.964 s
[INFO] Finished at: 2023-08-08T21:07:06+03:00
[INFO] ------------------------------------------------------------------------
```