DP2-1920-GI-02: Sprint 4 June 1, 2020

REFACTORING AND SONARCLOUD

In part I of this document, refactorings (improvements in the quality on the code) have been carried out based on a detected code smell. In part II, improvements in the quality of the code are evidenced by reducing code smell and we can see using SonarCloud screenshots.

PART I: REFACTORING

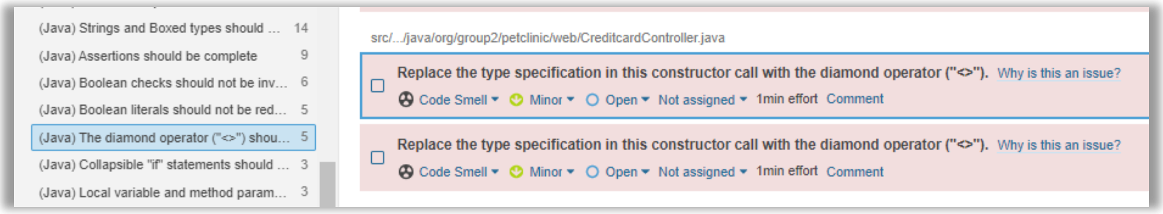
REFACTORING 1

Typo of code smell:

Composition Methods – Extractin Method – Duplicated Code.

Problem:

It was detected an error in the CreditcardController.java class twice. It has been detected twice because the code is duplicated in the get and post methods. Both request want to perform the same functionality: create a list of number equivalent to months to be used as a dropdown.

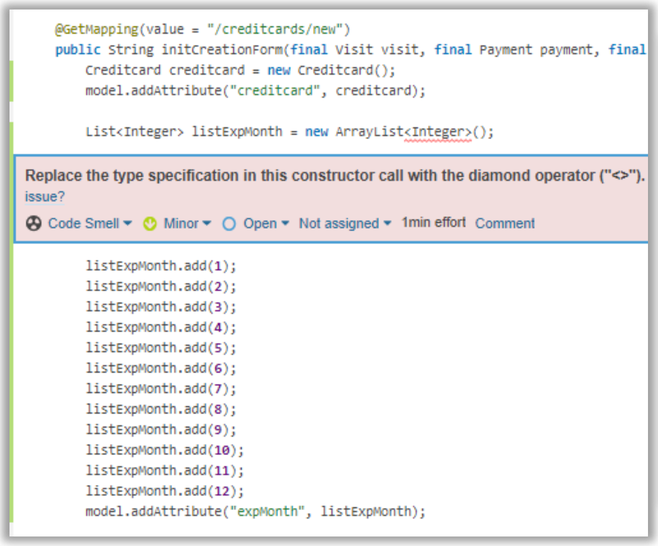


(Java) Strings and Boxed types should ...	14
(Java) Assertions should be complete	9
(Java) Boolean checks should not be inv...	6
(Java) Boolean literals should not be red...	5
(Java) The diamond operator ("<>") shou...	5
(Java) Collapsible "if" statements should ...	3
(Java) Local variable and method param...	3

src/.../java/org/group2/petclinic/web/CreditcardController.java

Replace the type specification in this constructor call with the diamond operator ("<>"). Why is this an issue?
Code Smell Minor Open Not assigned 1min effort Comment

Replace the type specification in this constructor call with the diamond operator ("<>"). Why is this an issue?
Code Smell Minor Open Not assigned 1min effort Comment

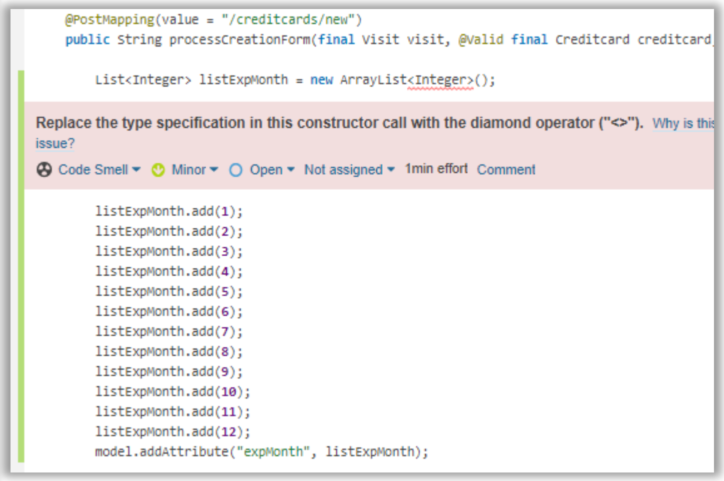


```
    @GetMapping(value = "/creditcards/new")
    public String initCreationForm(final Visit visit, final Payment payment, final
        Creditcard creditcard = new Creditcard();
        model.addAttribute("creditcard", creditcard);

        List<Integer> listExpMonth = new ArrayList<Integer>();
```

Replace the type specification in this constructor call with the diamond operator ("<>"). Why is this an issue?
Code Smell Minor Open Not assigned 1min effort Comment

```
        listExpMonth.add(1);
        listExpMonth.add(2);
        listExpMonth.add(3);
        listExpMonth.add(4);
        listExpMonth.add(5);
        listExpMonth.add(6);
        listExpMonth.add(7);
        listExpMonth.add(8);
        listExpMonth.add(9);
        listExpMonth.add(10);
        listExpMonth.add(11);
        listExpMonth.add(12);
        model.addAttribute("expMonth", listExpMonth);
```



```
    @PostMapping(value = "/creditcards/new")
    public String processCreationForm(final Visit visit, @Valid final Creditcard creditcard

        List<Integer> listExpMonth = new ArrayList<Integer>();
```

Replace the type specification in this constructor call with the diamond operator ("<>"). Why is this issue?
Code Smell Minor Open Not assigned 1min effort Comment

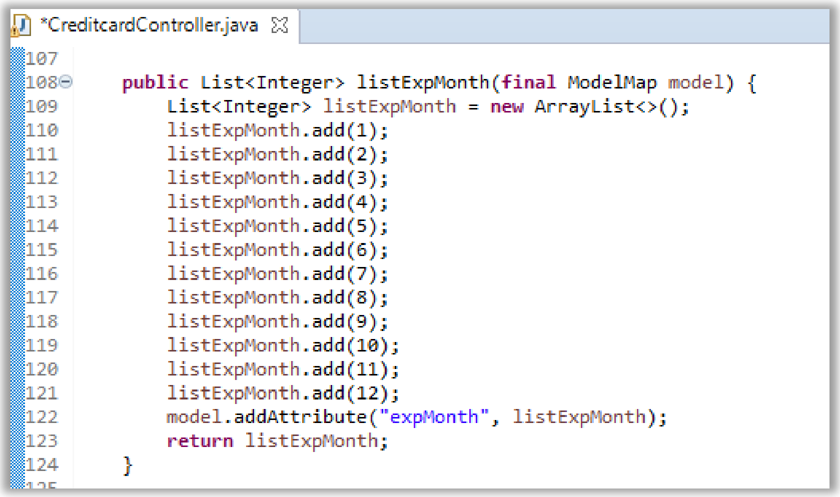
```
        listExpMonth.add(1);
        listExpMonth.add(2);
        listExpMonth.add(3);
        listExpMonth.add(4);
        listExpMonth.add(5);
        listExpMonth.add(6);
        listExpMonth.add(7);
        listExpMonth.add(8);
        listExpMonth.add(9);
        listExpMonth.add(10);
        listExpMonth.add(11);
        listExpMonth.add(12);
        model.addAttribute("expMonth", listExpMonth);
```

Solution:

The solution has been to create a public method that collects the duplicate code. In this method we have corrected the error that SonarCloud marked us: put a diamond operator. Then this method will be called in the corresponding place of the get and post method.

This reduces the percentage of duplicate code that exists in our application.

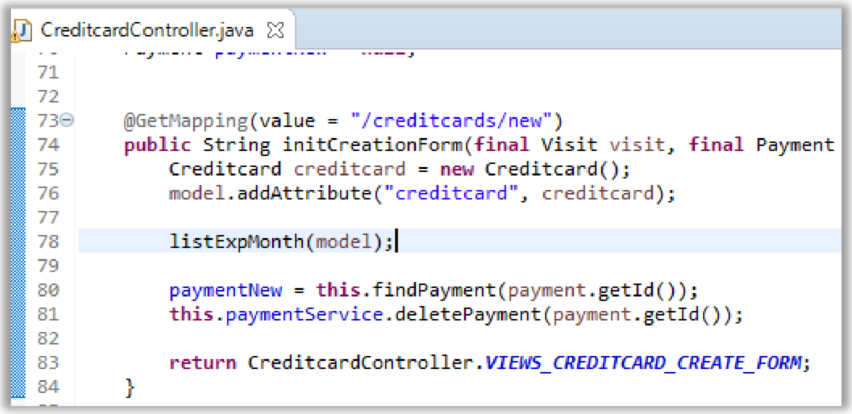
The new public method:



*CreditcardController.java

```java
    public List<Integer> listExpMonth(final ModelMap model) {
        List<Integer> listExpMonth = new ArrayList<>();
        listExpMonth.add(1);
        listExpMonth.add(2);
        listExpMonth.add(3);
        listExpMonth.add(4);
        listExpMonth.add(5);
        listExpMonth.add(6);
        listExpMonth.add(7);
        listExpMonth.add(8);
        listExpMonth.add(9);
        listExpMonth.add(10);
        listExpMonth.add(11);
        listExpMonth.add(12);
        model.addAttribute("expMonth", listExpMonth);
        return listExpMonth;
    }
```

Calls to that method:



CreditcardController.java

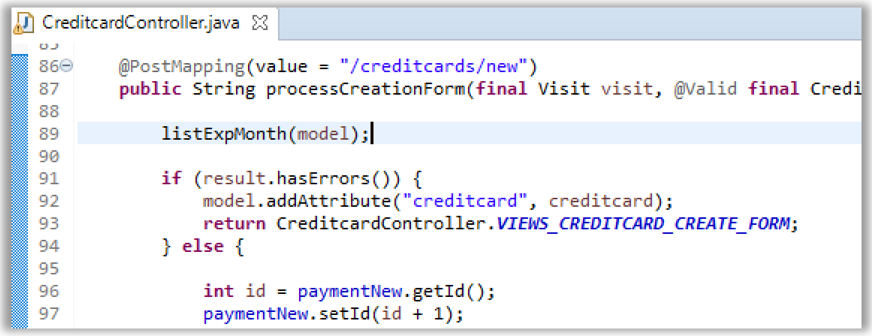
```java

    @GetMapping(value = "/creditcards/new")
    public String initCreationForm(final Visit visit, final Payment
        Creditcard creditcard = new Creditcard();
        model.addAttribute("creditcard", creditcard);

        listExpMonth(model);

        paymentNew = this.findPayment(payment.getId());
        this.paymentService.deletePayment(payment.getId());

        return CreditcardController.VIEWS_CREDITCARD_CREATE_FORM;
    }
```



CreditcardController.java

```java
    @PostMapping(value = "/creditcards/new")
    public String processCreationForm(final Visit visit, @Valid final Credi

        listExpMonth(model);

        if (result.hasErrors()) {
            model.addAttribute("creditcard", creditcard);
            return CreditcardController.VIEWS_CREDITCARD_CREATE_FORM;
        } else {

            int id = paymentNew.getId();
            paymentNew.setId(id + 1);
```

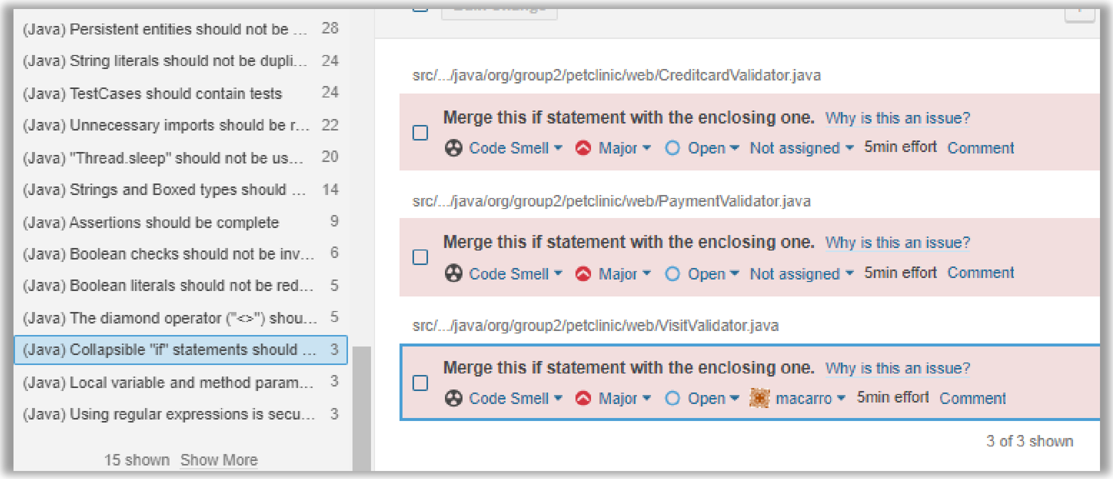
REFACTORING 2

Typo of code smell:

Simplifying Conditionals – Long method.

Problem:

A code smell has been detected in wich there are two if statemets in a row, so it's unnecesary. This repeats three times in differents classes.



(Java) Persistent entities should not be ...	28
(Java) String literals should not be dupli...	24
(Java) TestCases should contain tests	24
(Java) Unnecessary imports should be r...	22
(Java) "Thread.sleep" should not be us...	20
(Java) Strings and Boxed types should ...	14
(Java) Assertions should be complete	9
(Java) Boolean checks should not be inv...	6
(Java) Boolean literals should not be red...	5
(Java) The diamond operator ("<>") shou...	5
(Java) Collapsible "if" statements should ...	3
(Java) Local variable and method param...	3
(Java) Using regular expressions is secu...	3

15 shown Show More

src/.../java/org/group2/petclinic/web/CreditcardValidator.java

Merge this if statement with the enclosing one. Why is this an issue?
Code Smell Major Open Not assigned 5min effort Comment

src/.../java/org/group2/petclinic/web/PaymentValidator.java

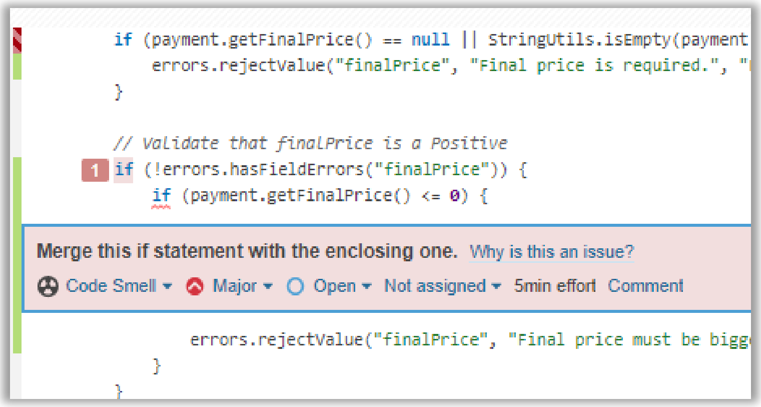
Merge this if statement with the enclosing one. Why is this an issue?
Code Smell Major Open Not assigned 5min effort Comment

src/.../java/org/group2/petclinic/web/VisitValidator.java

Merge this if statement with the enclosing one. Why is this an issue?
Code Smell Major Open macarro 5min effort Comment

3 of 3 shown

Example of how it looks in error inside a class:



```
        if (payment.getFinalPrice() == null || StringUtils.isEmpty(payment
            errors.rejectValue("finalPrice", "Final price is required.", "
        }

        // Validate that finalPrice is a Positive
      1 if (!errors.hasFieldErrors("finalPrice")) {
            if (payment.getFinalPrice() <= 0) {
```

Merge this if statement with the enclosing one. Why is this an issue?
Code Smell Major Open Not assigned 5min effort Comment

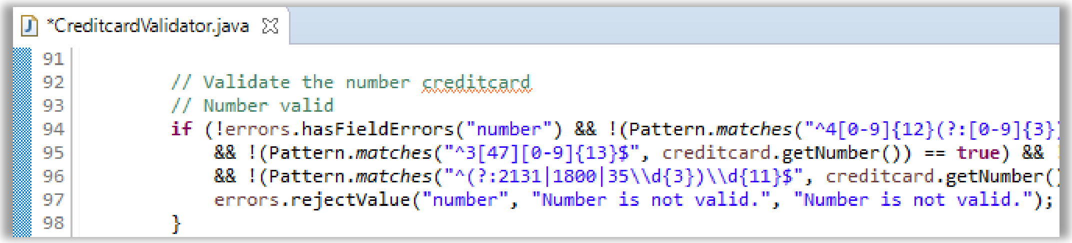
```
                errors.rejectValue("finalPrice", "Final price must be bigg
            }
        }
```

<u>Solution</u>:

For each of the classes where it was in error: one if statement has been removed and his condition has been included in the other if statement.

This creates a cleaner and more summarized code. Shorten the method.

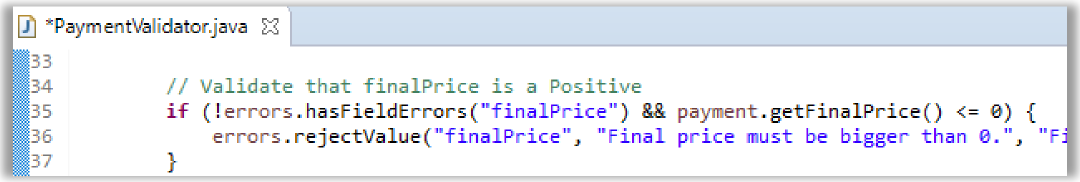
For CreditcardValidator.java:



*CreditcardValidator.java

```
        // Validate the number creditcard
        // Number valid
        if (!errors.hasFieldErrors("number") && !(Pattern.matches("^4[0-9]{12}(?:[0-9]{3}
            && !(Pattern.matches("^3[47][0-9]{13}$", creditcard.getNumber()) == true) &&
            && !(Pattern.matches("^(?:2131|1800|35\\d{3})\\d{11}$", creditcard.getNumber(
            errors.rejectValue("number", "Number is not valid.", "Number is not valid.");
        }
```

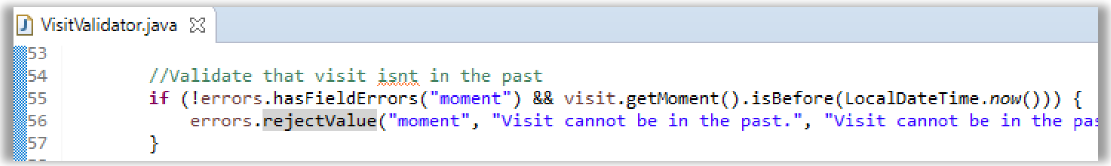
For PaymentValidator.java:



*PaymentValidator.java

```
        // Validate that finalPrice is a Positive
        if (!errors.hasFieldErrors("finalPrice") && payment.getFinalPrice() <= 0) {
            errors.rejectValue("finalPrice", "Final price must be bigger than 0.", "Fi
        }
```

For VisitValidator.java:



VisitValidator.java

```
        //Validate that visit isnt in the past
        if (!errors.hasFieldErrors("moment") && visit.getMoment().isBefore(LocalDateTime.now())) {
            errors.rejectValue("moment", "Visit cannot be in the past.", "Visit cannot be in the pa
        }
```

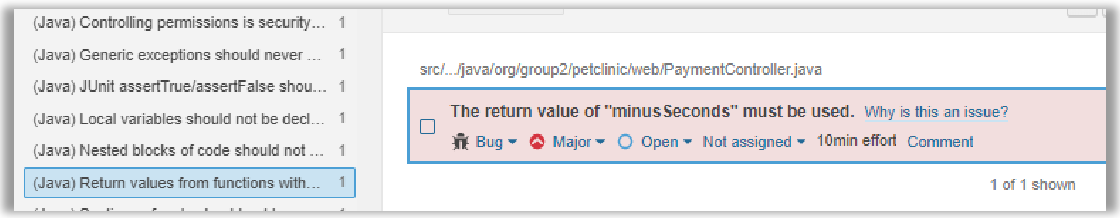
REFACTORING 3

Typo of code smell:

Organizing Data – Temporary Field.

Problem:

A code smell has been detected in wich we see that a method is called to perform an operation but the value it returns is not stored in any variable. This can cause problems that the method does not do the operation that we want to use later.

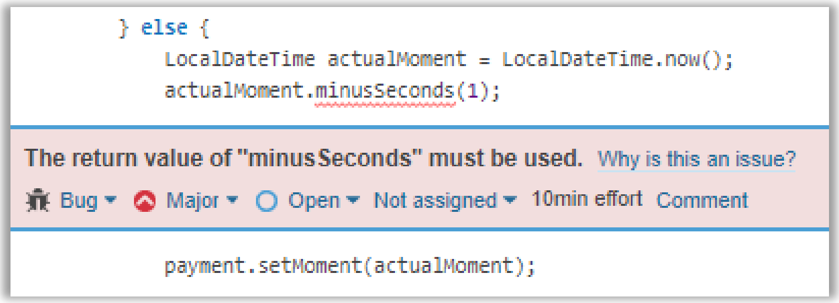


```
(Java) Controlling permissions is security...  1
(Java) Generic exceptions should never ...  1
(Java) JUnit assertTrue/assertFalse shou...  1
(Java) Local variables should not be decl...  1
(Java) Nested blocks of code should not ...  1
(Java) Return values from functions with...  1
```

src/.../java/org/group2/petclinic/web/PaymentController.java

The return value of "minusSeconds" must be used. Why is this an issue?
Bug Major Open Not assigned 10min effort Comment

1 of 1 shown



```
        } else {
            LocalDateTime actualMoment = LocalDateTime.now();
            actualMoment.minusSeconds(1);
```

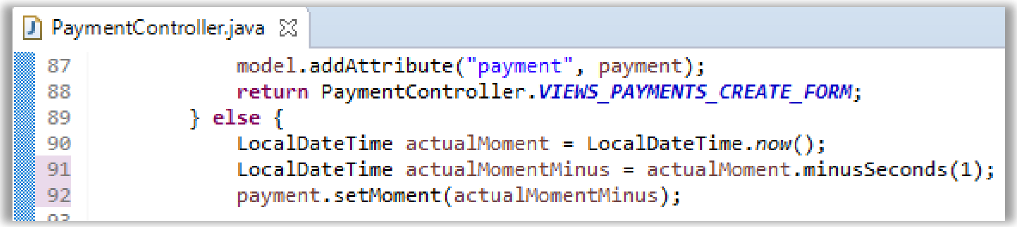
The return value of "minusSeconds" must be used. Why is this an issue?
Bug Major Open Not assigned 10min effort Comment

```
            payment.setMoment(actualMoment);
```

Solution:

It has been solved by storing in an variable the operation that this method performs. Then this variable is the one used.

This makes the value of the variable available at any time and we make sure that the operation is carried out.



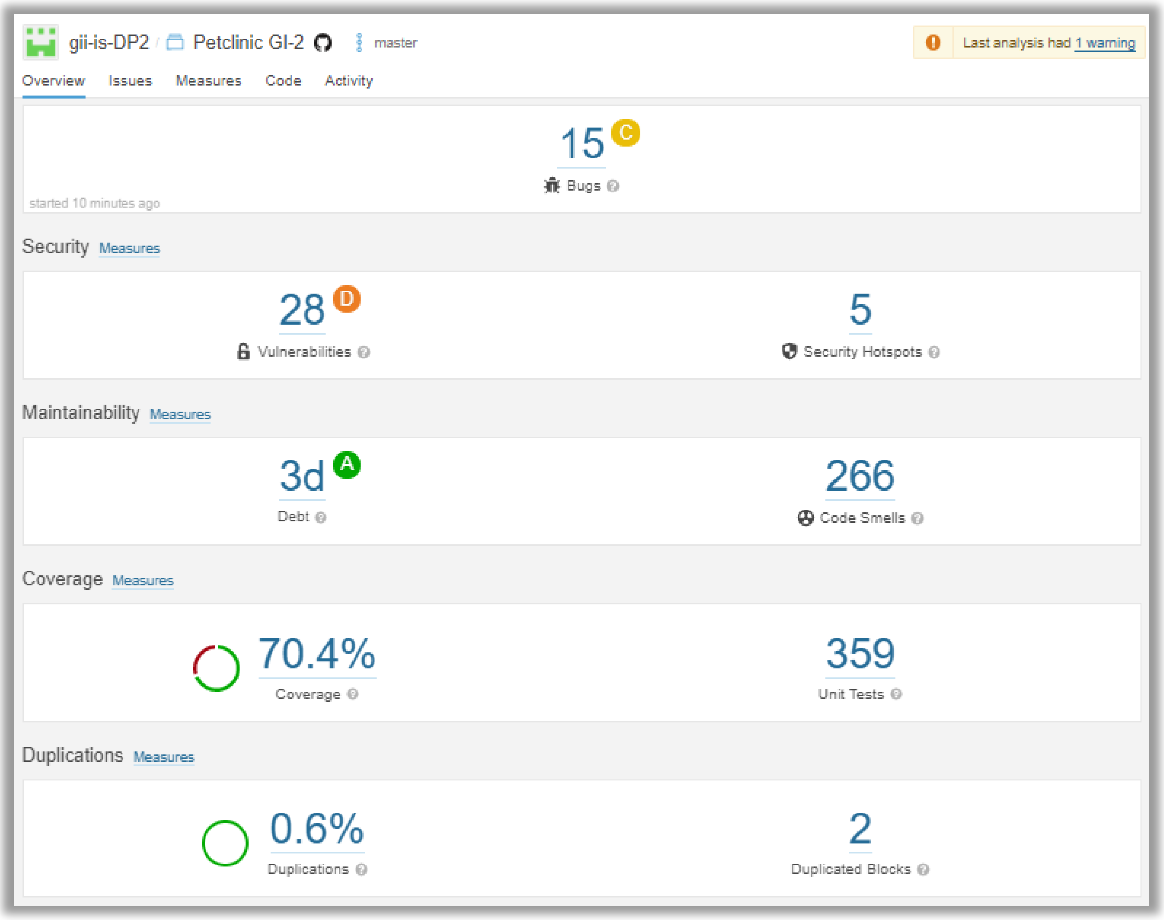
PaymentController.java

```
            model.addAttribute("payment", payment);
            return PaymentController.VIEWS_PAYMENTS_CREATE_FORM;
        } else {
            LocalDateTime actualMoment = LocalDateTime.now();
            LocalDateTime actualMomentMinus = actualMoment.minusSeconds(1);
            payment.setMoment(actualMomentMinus);
```

PART II: SONARCLOUD

SONAR CLOUD BEFORE REFACTORINGS

You can see that it is made up of **266** code smells.



gii-is-DP2 / Petclinic GI-2 master

Last analysis had 1 warning

Overview Issues Measures Code Activity

started 10 minutes ago

15 C
Bugs

Security Measures

28 D
Vulnerabilities

5
Security Hotspots

Maintainability Measures

3d A
Debt

266
Code Smells

Coverage Measures

70.4%
Coverage

359
Unit Tests

Duplications Measures

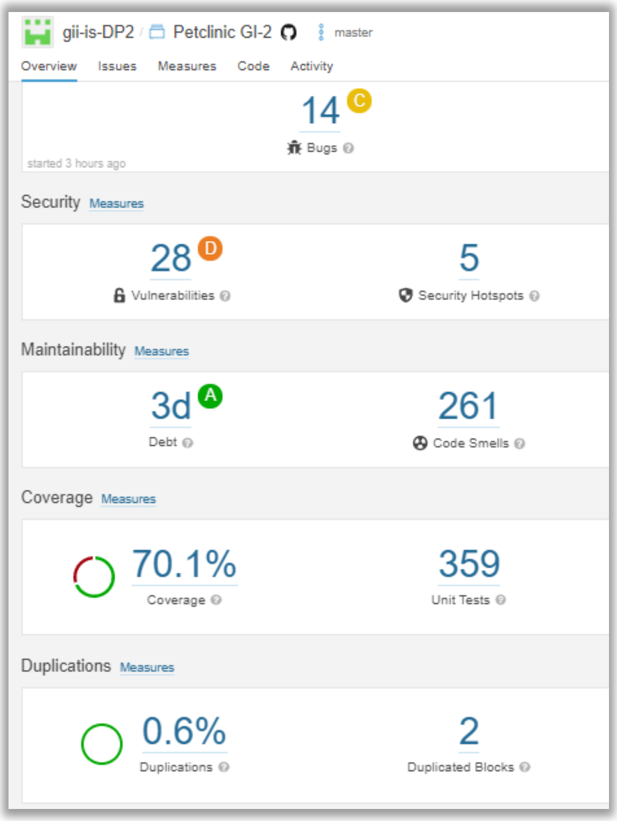
0.6%
Duplications

2
Duplicated Blocks

SONAR CLOUD AFTER REFACTORINGS

You can see that it is made up of **261** code smells.

So we have improved our code quality by correcting 5 code smells.


gii-is-DP2 / Petclinic GI-2 master

Overview Issues Measures Code Activity

started 3 hours ago

14 C Bugs

Security Measures

28 D Vulnerabilities

5 Security Hotspots

Maintainability Measures

3d A Debt

261 Code Smells

Coverage Measures

70.1% Coverage

359 Unit Tests

Duplications Measures

0.6% Duplications

2 Duplicated Blocks