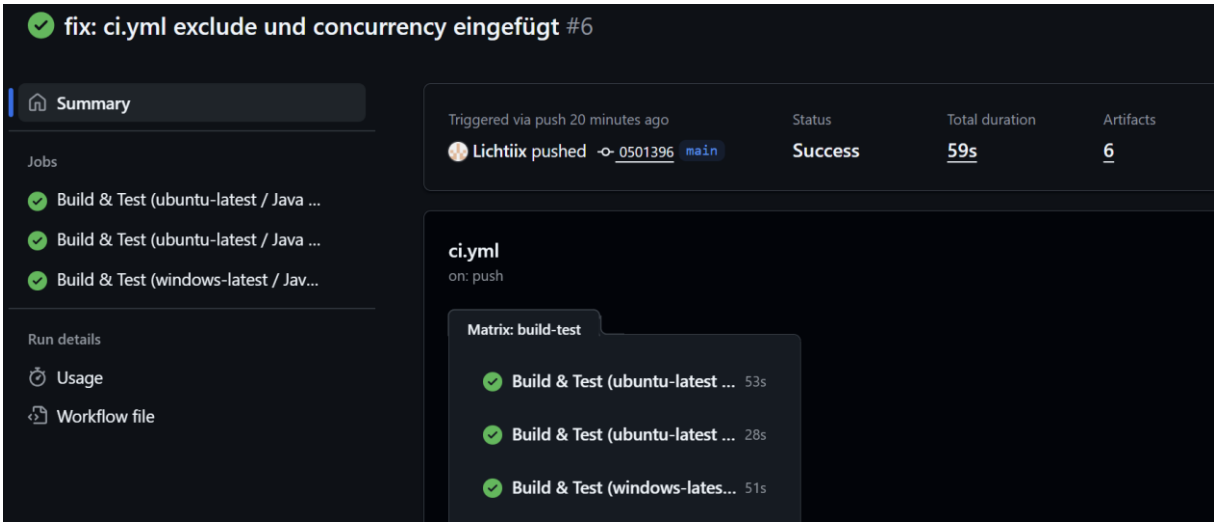
CICD Ex02 Lichtenberger

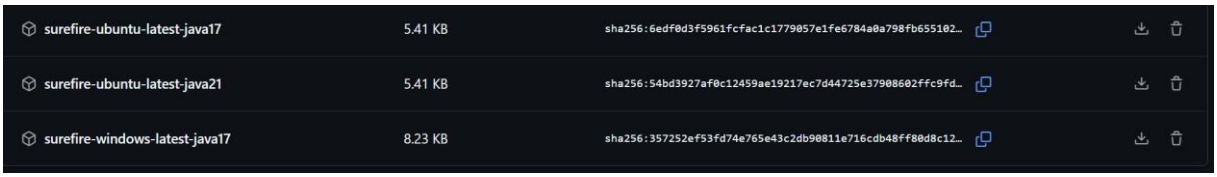
What you deliver

1. **GitHub repo link** (same repo as UE01, continued).

https://github.com/Lichtiix/cicd-BA-uebung01-Lichtenberger

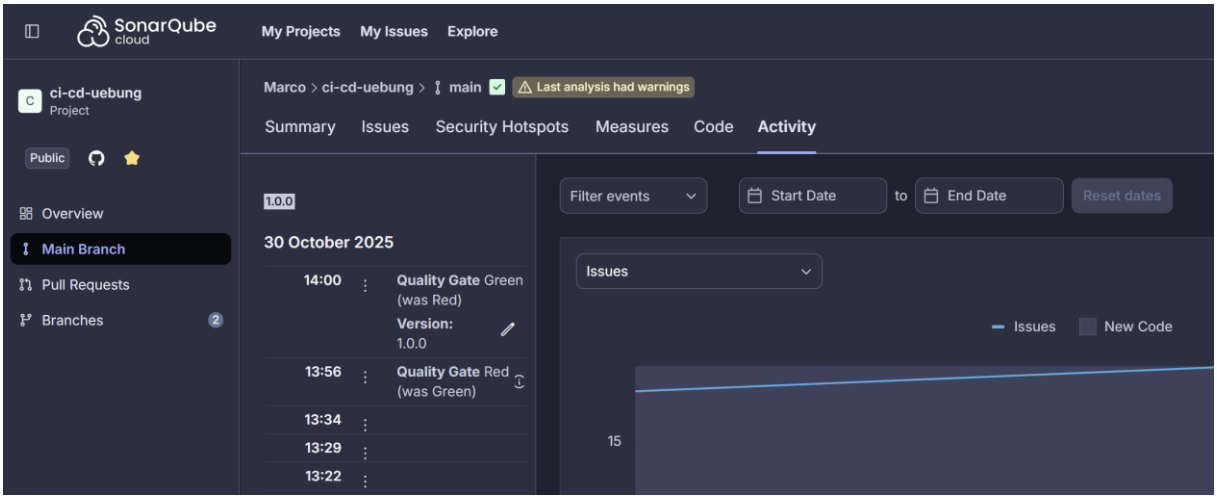
2. A short **PDF report** with screenshots and brief comments:
   - Successful CI runs (matrix view).
   - Artifacts (Surefire reports) for each matrix variant.



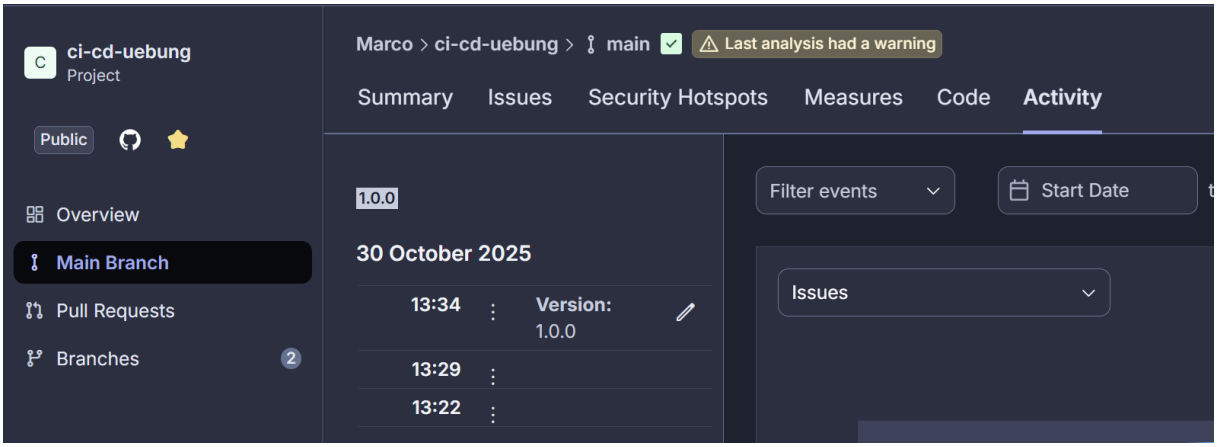


- **Coverage**
  - > JaCoCo erzeugt einen **XML-Report** am erwarteten Ort (z. B. target/site/jacoco/jacoco.xml).
  - > Coverage ist nach der Analyse in SonarCloud sichtbar (≠ 0 %, sofern Tests Code abdecken).

○ SonarCloud project dashboard / PR decoration / Quality Gate status.



- **SonarCloud**
  - > Analyse erfolgreich bei jedem Push auf main.



- The changed code parts you fixed for Sonar issues (before/after).

- **Issue-Bearbeitung**
  > Mindestens **zwei** sinnvolle Issues behoben (z. B. echter Bug, Problem bei Reliability/Maintainability/Security).

1. Issue

```
public class App {
    public static void main(String[] args) {
        Calculator calc = new Calculator();
        int sum = calc.add(2, 3);
        System.out.println("Sum(2,3) = " + sum);

        List<Integer> numbers = Arrays.asList(1, 2, 3, 4);
        int s1 = calc.sumUp(numbers);
        int s2 = calc.addAll(numbers);
        System.out.println("sumUp=" + s1 + ", addAll=" + s2);

        System.out.println("isPalindrome('Anna')? " + TextUtils.isPalindrome("Anna"));
        System.out.println("safeParseInt('42'): " + TextUtils.safeParseInt("42"));
        System.out.println("safeParseInt('x'): " + TextUtils.safeParseInt("x"));
    }
}
```

Replace this use of System.out by a logger.

```
public class App {
    Run | Debug
    public static void main(String[] args) {
        Calculator calc = new Calculator();
        int sum = calc.add(a:2, b:3);
        System.out.println("Sum(2,3) = " + sum);

        List<Integer> numbers = Arrays.asList(...a:1, 2, 3, 4);
        int s1 = calc.sumUp(numbers);
        int s2 = calc.addAll(numbers);
        System.out.println("sumUp=" + s1 + ", addAll=" + s2);

        System.out.println("isPalindrome('Anna')? " + TextUtils.isPalindrome(input:"Anna"));
        System.out.println("safeParseInt('42'): " + TextUtils.safeParseInt(s:"42"));
        System.out.println("safeParseInt('x'): " + TextUtils.safeParseInt(s:"x"));
    }
}
```

Ausbesserung:

```
public class App {
    private static final Logger logger = Logger.getLogger(App.class.getName());

    Run | Debug
    public static void main(String[] args) {
        Calculator calc = new Calculator();
        int sum = calc.add(a:2, b:3);
        logger.info("Sum(2,3) = " + sum);

        List<Integer> numbers = Arrays.asList(...a:1, 2, 3, 4);
        int s1 = calc.sumUp(numbers);
        int s2 = calc.addAll(numbers);
        logger.info("sumUp=" + s1 + ", addAll=" + s2);

        logger.info("isPalindrome('Anna')? " + TextUtils.isPalindrome(input:"Anna"));
        logger.info("safeParseInt('42'): " + TextUtils.safeParseInt(s:"42"));
        logger.info("safeParseInt('x'): " + TextUtils.safeParseInt(s:"x"));
    }
}
```

2. Issue

```
public class Calculator {

    public static int MAX_OPERANDS = 100;
```

Make MAX_OPERANDS a static final constant or non-public and provide accessors if needed.

```
public class Calculator {

    public static int MAX_OPERANDS = 100;
```

Ausbesserung:

```
private static final int MAX_OPERANDS = 100;
```

```
public static int getMaxOperands() {
    return MAX_OPERANDS;
}
```

3. Updated **README.md** in the repo with a **build badge** pointing to your CI workflow.

- **README-Badge**
  > Ein **Status-Badge** für den Workflow ist **oben** in der README sichtbar.