



DevOps Code Quality



Building Competence. Crossing Borders.

Warum ist Code Quality wichtig?

*Measuring programming progress
by lines of code
is like measuring
aircraft building progress by weight.*

Bill Gates

Was gehört zur Code Qualität?

Code Qualität umfasst unter anderem

- automatische Tests (siehe letzte Woche)
- Testing Coverage
- Static Code Analysis (Lint)
- Reporting Tools

Code Coverage

Testing: Coverage

Test Coverage / Line Coverage = Testabdeckung

```
52: @GetMapping("/services/todo")
53: public List<PathListEntry<Integer>> todo() {
54:     var result = new ArrayList<PathListEntry<Integer>>();
55:     for (var todo : this.todos.values()) {
56:         var entry = new PathListEntry<Integer>();
57:         entry.setKey(todo.getId(), "todoKey");
58:         entry.setName(todo.getTitle());
59:         entry.getDetails().add(todo.getDescription());
60:         entry.setTooltip(todo.getDescription());
61:         result.add(entry);
62:     }
63:     return result;
64: }
65:
66: @GetMapping("/services/todo/{key}")
67: public Todo getTodo(@PathVariable Integer key) {
68:     return this.todos.get(key);
69: }
70:
71: @PostMapping("/services/todo")
72: public void createTodo(@RequestBody Todo todo) {
73:     var newKey = this.todos.keySet().stream().max(Comparator.naturalOrder()).orElse(0) + 1;
74:     todo.setId(newKey);
75:     this.todos.put(newKey, todo);
76: }
```

Conditions (if / else)

```
public boolean addAll(int index, Collection c) {
    if(c.isEmpty()) {
        return false;
    } else if(_size == index || _size == 0) {
        return addAll(c);
    } else {
        Listable succ = getListableAt(index);
        Listable pred = (null == succ) ? null : succ.prev();
        Iterator it = c.iterator();
        while(it.hasNext()) {
            pred = insertListable(pred, succ, it.next());
        }
        return true;
    }
}
```

◆ 1 of 2 branches missed.
Press 'F2' for focus

Java Code Coverage

Kann die Testabdeckung in Java analysieren

Gradle Plugin

Es gibt ein JaCoCo-Plugin für Gradle.

Dies kann in build.gradle ergänzt werden.

```
plugins {  
    ...  
    id 'jacoco'  
}  
  
test {  
    finalizedBy jacocoTestReport // report is always generated after tests  
}
```



Tutorial Code Coverage

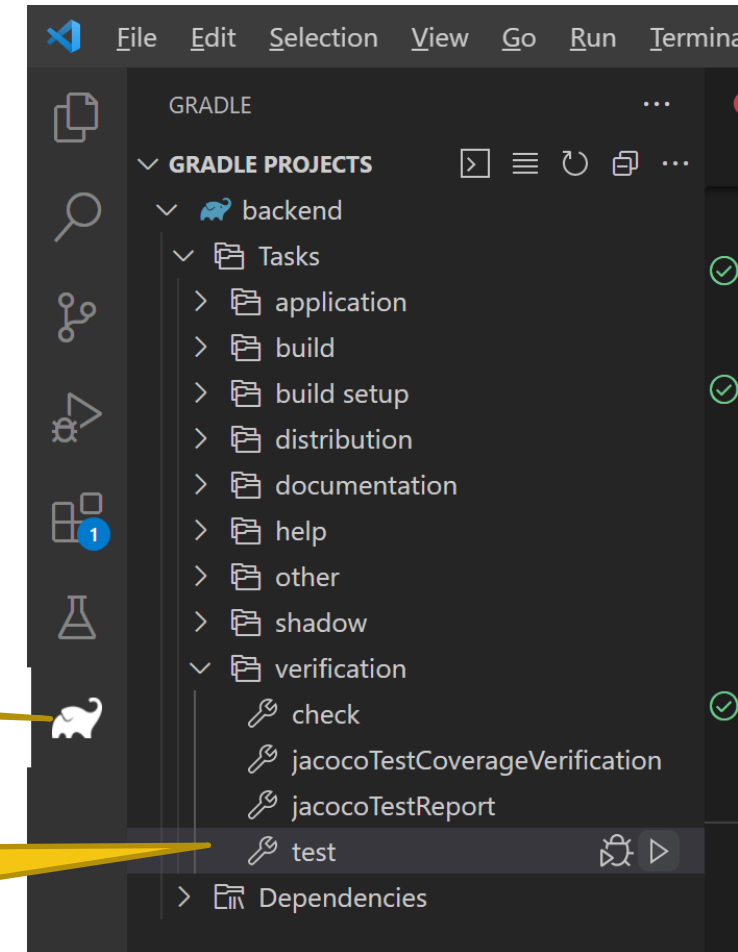
Code Coverage für DevOpsPath (Backend) analysieren

Gradle Task test starten

- Der JaCoCo Agent wird gestartet
- Alle Test laufen durch
- JaCoCo «merkt» sich, welche Codezeilen durchlaufen wurden

1. Gradle Sicht

2. Task verification → test ausführen



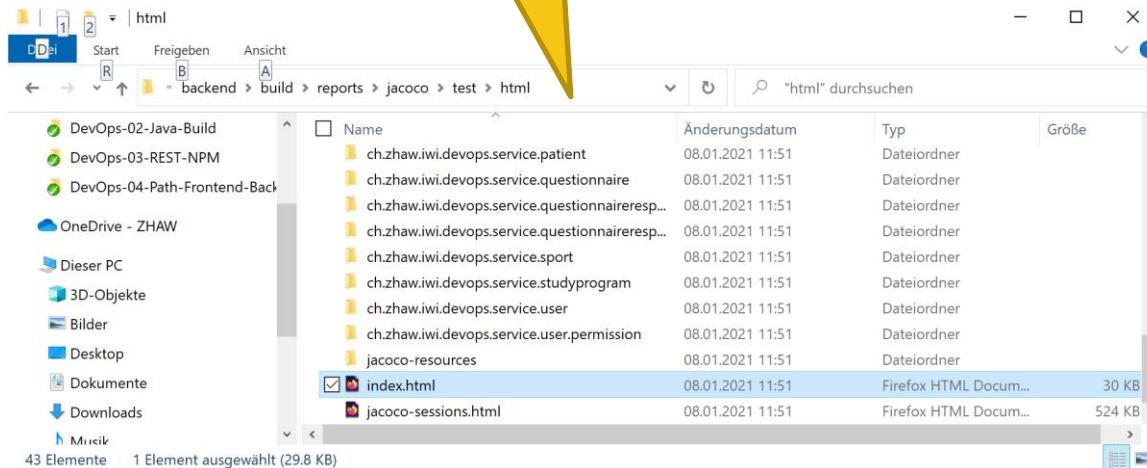
Resultate als HTML Report

Tutorial

JaCoCo erzeugt einen HTML-Report, welche Codezeilen getestet wurden:

backend\build\reports\jacoco\test\html\index.html

1. Verzeichnis öffnen
(Reveal in File Explorer)



backend > ch.zhaw.iwi.devops.fizzbuzz > FizzBuzzConverter.java

FizzBuzzConverter.java

```
1. package ch.zhaw.iwi.devops.fizzbuzz;
2.
3. public class FizzBuzzConverter {
4.
5.     public String convert(int i) {
6.         if (i % 3 == 0 && i % 7 == 0) {
7.             return "FizzBuzz";
8.         } else if (i % 3 == 0) {
9.             return "Fizz";
10.        } else if (i % 7 == 0) {
11.            return "Buzz";
12.        }
13.        return String.valueOf(i);
14.    }
15. }
```

2. In Browser öffnen

Resultate in VisualStudio Code sichtbar machen

Tutorial

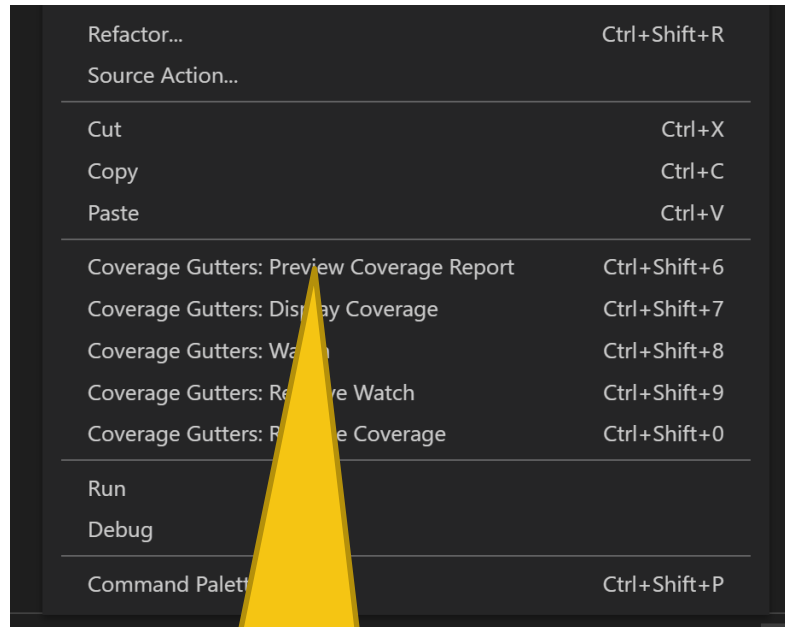
Coverage Gutters Plugin kann Coverage direkt in VisualStudio Code anzeigen. Dazu muss mit Gradle/JaCoCo das entsprechende Datenfile erstellt werden.

build.gradle (in DevOpsDemo bereits vorhanden):

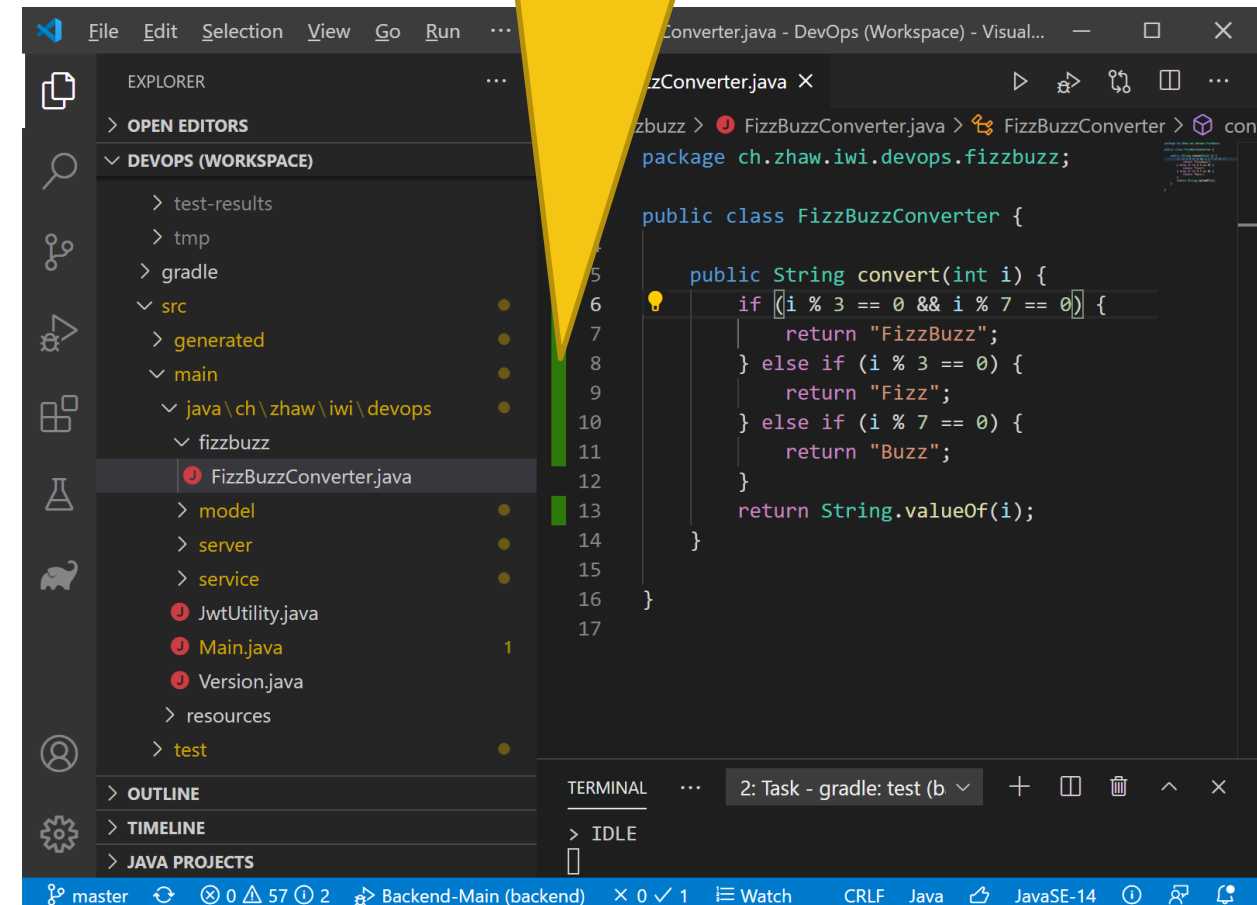
```
jacocoTestReport {  
    reports {  
        xml.required = true  
        csv.required = false  
        xml.destination(file("${rootProject.projectDir}/target/site/jacoco/cov.xml"))  
    }  
}
```

Resultate in VisualStudio Code sichtbar machen

Tutorial



1. Display Coverage auf gewünschter Datei



Lint: Static Code Analysis

Begriff

Der Name Lint leitet sich von der englischen Bezeichnung für unerwünschte Anteile an Fasern und Flaum in Schafwolle ab (englisch für «Fussel»).

Lint / Linten

Lint ist eine Software zur statischen Code-Analyse. Davon abgeleitet hat sich das Verb linten (englisch to lint) für das Durchführen der statischen Code-Analyse etabliert.

Tools

Es gibt eine grosse Anzahl von Linting Tools, deren Funktionalität sich teilweise überschneidet.

Eine Auswahl für **JavaScript**:



Eine Auswahl für **Java**:



Tutorial Lint

Lint für DevOpsPath (Frontend) analysieren

Beispiel: ESLint

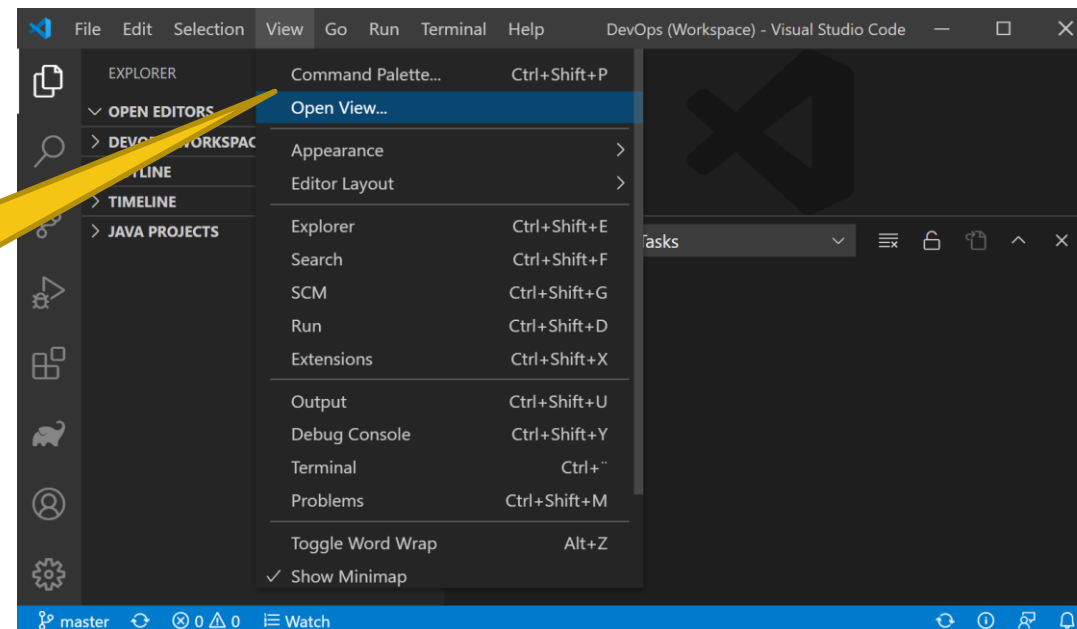
Tutorial

ESLint ist im Path Frontend integriert und kann als NPM-Task ausgeführt werden.

Variante 1: Terminal öffnen, frontend-Ordner, npm run lint

Variante 2: Visual Studio Code kann npm-Scripts anzeigen

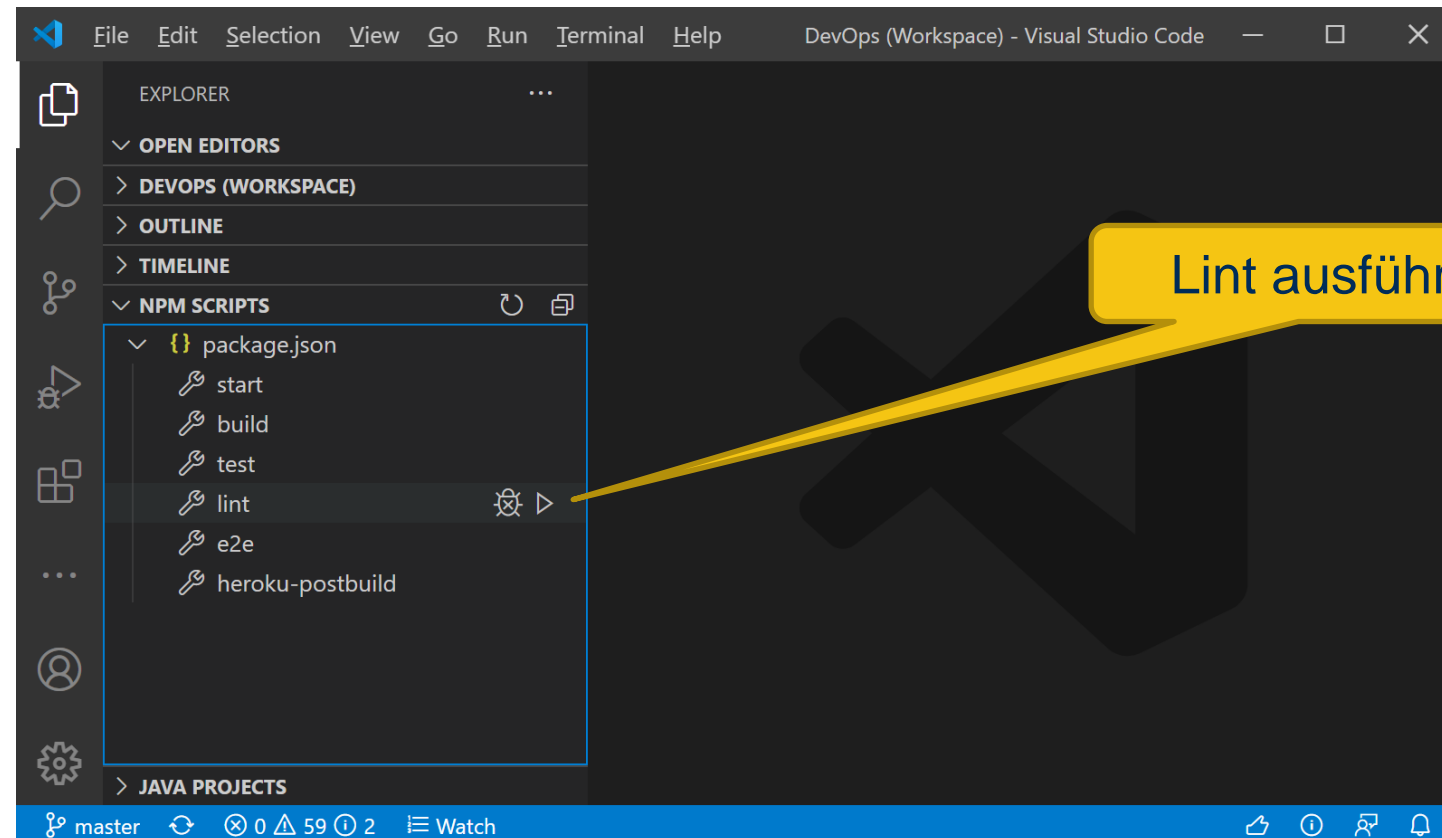
Menü «View»
«Open View...»



Beispiel: ESLint

Tutorial

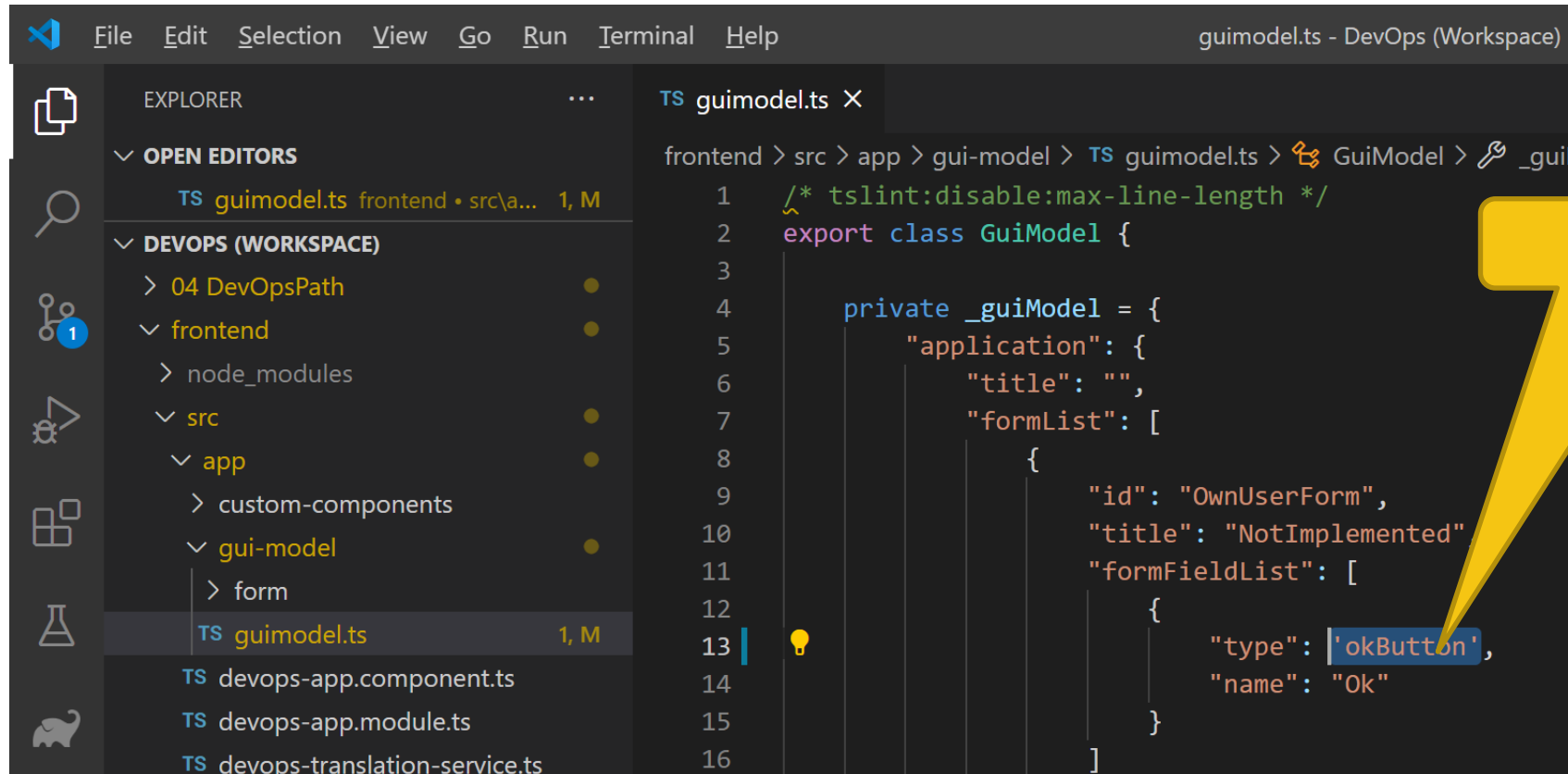
ESLint ist im Path Frontend integriert und kann als NPM-Task ausgeführt werden.



Fehler einbauen (in guimodel.ts) und linten

Tutorial

– F



The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar shows the project structure: 'DEVOPS (WORKSPACE)' containing '04 DevOpsPath', 'frontend', 'node_modules', 'src', 'app', 'custom-components', 'gui-model', and 'form'. The file 'TS guimodel.ts' is selected. The main editor shows the content of 'guimodel.ts' with the following code:

```
1  /* tslint:disable:max-line-length */
2  export class GuiModel {
3
4      private _guiModel = {
5          "application": {
6              "title": "",
7              "formList": [
8                  {
9                      "id": "OwnUserForm",
10                     "title": "NotImplemented"
11                     "formFieldList": [
12                         {
13                             "type": 'okButton',
14                             "name": "Ok"
15                         }
16                     ]
17                 }
18             ]
19         }
20     }
21 }
```

A yellow lightning bolt callout labeled 'Fehler' points to the single quote in the 'type' property on line 13, which is causing a linting error.

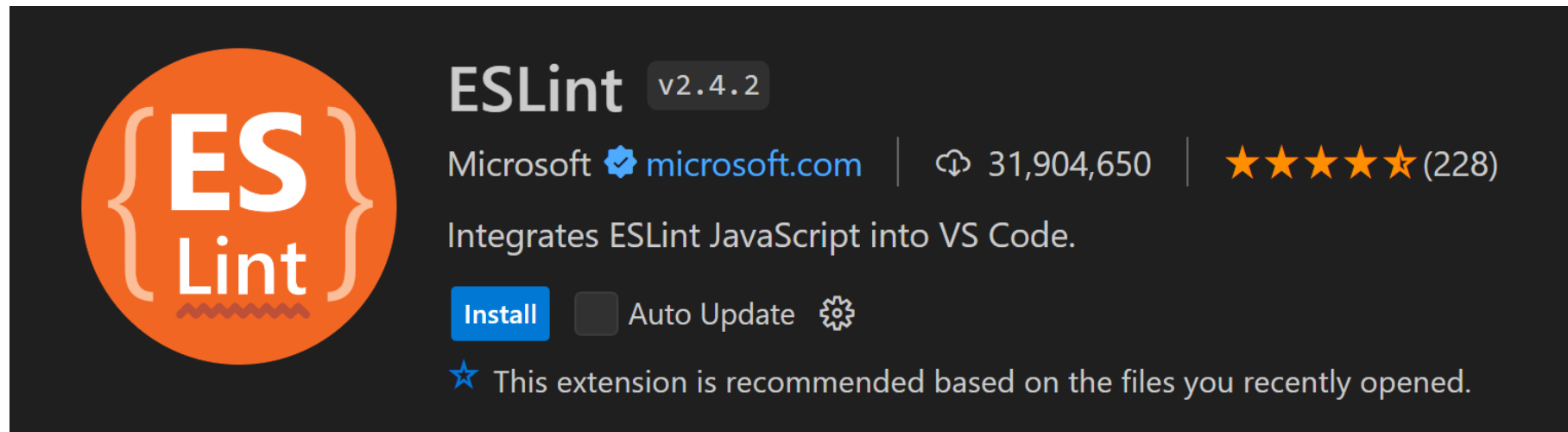
Fehlermeldung:

ERROR: C:/.../DevOpsPath/frontend/src/app/gui-model/guimodel.ts:13:37 - ' should be "

Beispiel: ESLint – Integration in VisualStudio Code

Tutorial

ESLint kann zusätzlich als VisualStudio Code Erweiterung installiert werden:



Beispiel: ESLint – Integration in VisualStudio Code

Tutorial

```
TS guimodel.ts 1, M X
frontend > src > app > gui-model > TS guimodel.ts > GuiModel > _gui
1  /* tslint:disable:max-line-length */
2  export class GuiModel {
3
4      private _guiModel = {
5          "application": {
6              "title": "DevOpsDemo FS2023",
7              "formList": [
8                  {
9                      "id": "OwnUserForm",
10                     "title": "NotImplemented",
11                     "formFieldList": [
12                         {
13                             "type": 'okButton',
14                             "name": "Ok"
15                         }
16                     ]
17                 },
18             ]
19         },
20     };
21 }
```

Linting
Problem
wird
markiert

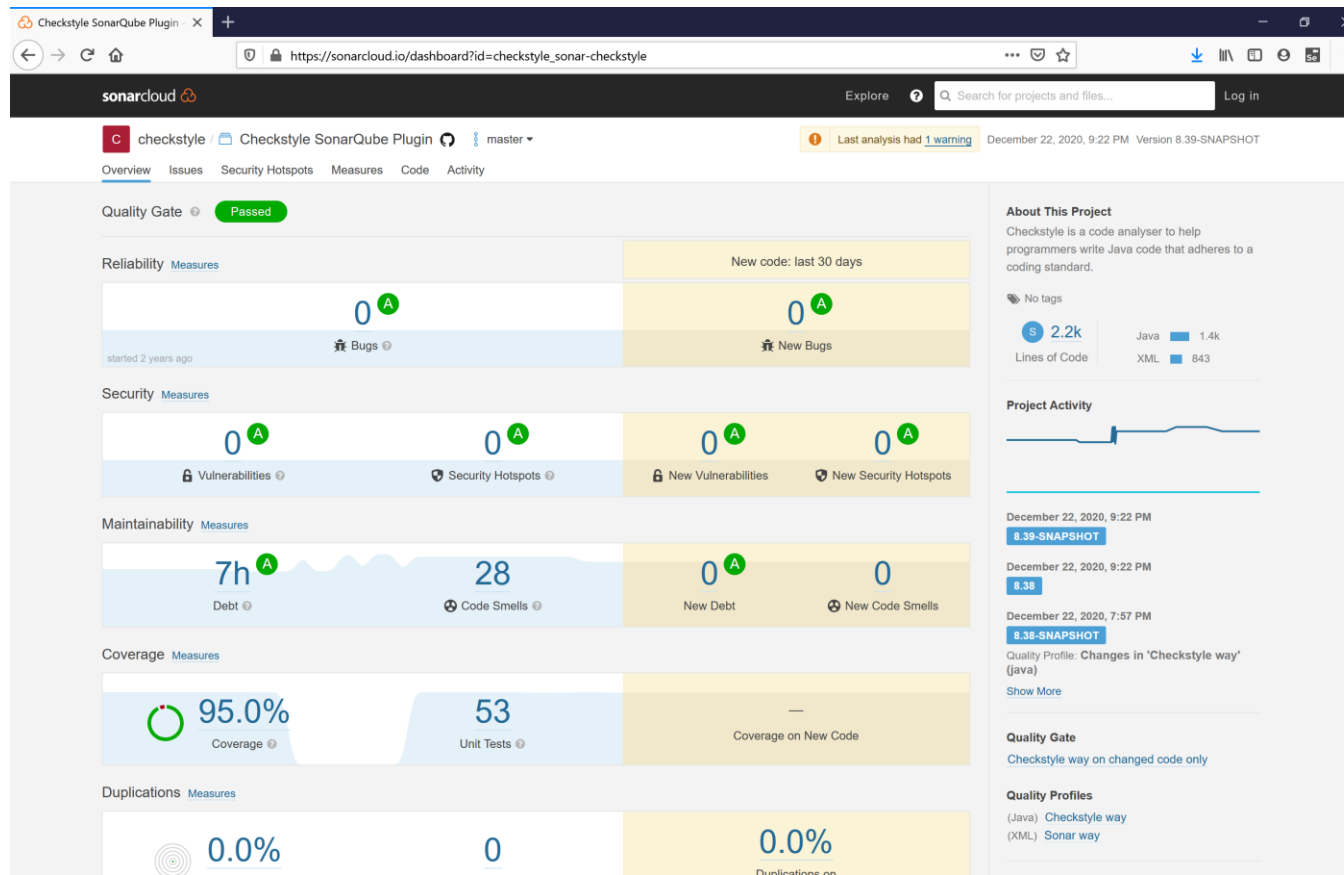
Reporting Tools

Warum?

Ergebnisse wie Tests, Code Coverage und Linting Results bilden immer nur den aktuellen Stand ab. Für ein erfolgreiches Projekt ist aber entscheidend

- Wieviele Unit Tests gibt es pro Anzahl Code? Nimmt die Zahl ab, zu oder ist sie stabil?
- Nimmt die Code Coverage ab, bleibt sie stabil oder nimmt sie zu?
- Wie beeinflusst ein einzelner Commit (oder sogar ein einzelner Entwickler) die Metriken?

Monitoring der Software-Qualität



<https://www.sonarqube.org>

<https://sonarcloud.io/dashboard?id=com.puppycrawl.tools%3Acheckstyle>

Tutorial

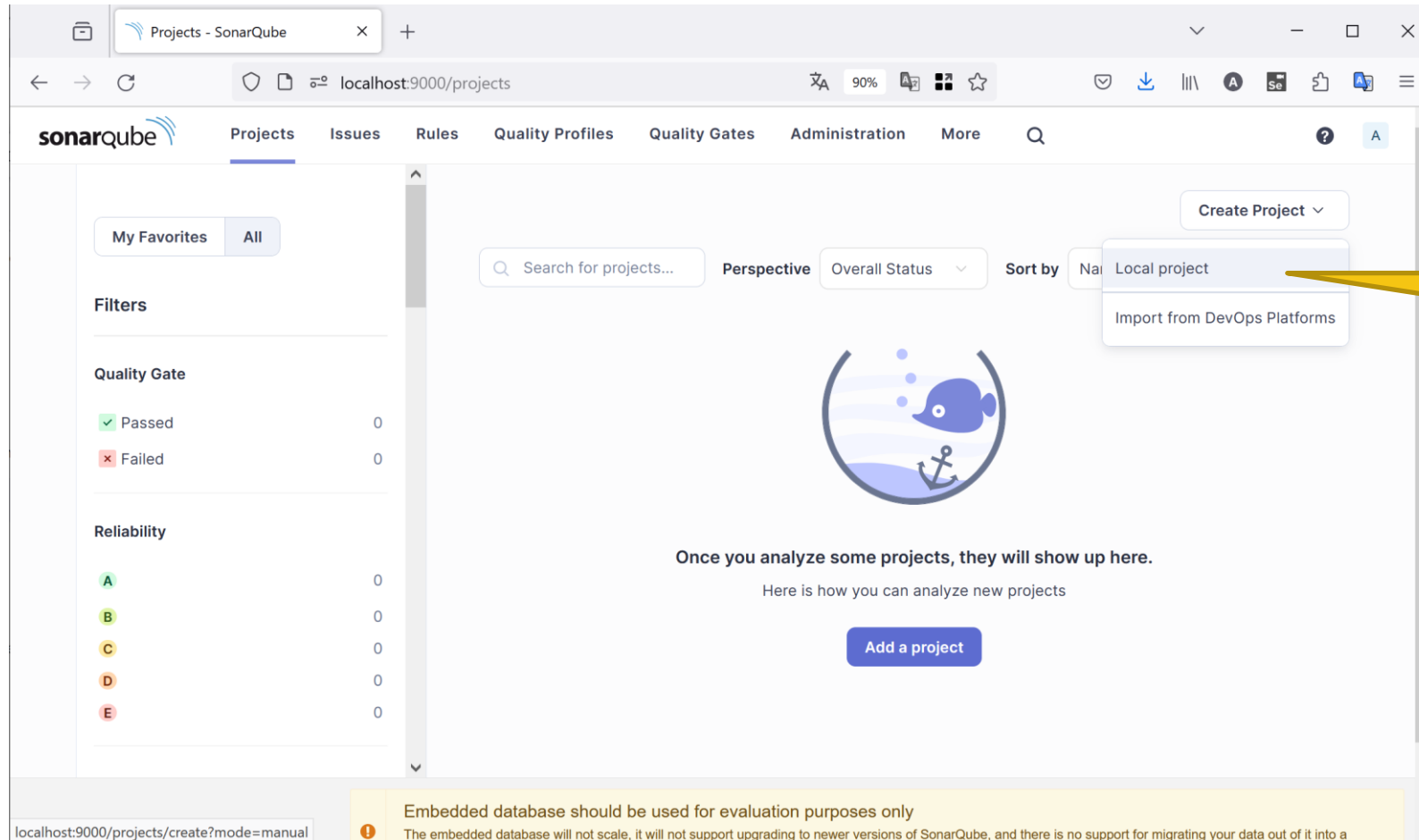
Mit SonarQube einen Report erstellen

- Backend
- Frontend

SonarQube Installation & Start

Tutorial

Projekt erstellen und der Anleitung folgen



SonarQube Backend-Projekt erstellen

Tutorial

Projekt «DevOpsDemo-Backend» (eigenen Namen wählen) erstellen
und der Anleitung folgen

The image shows a screenshot of the SonarQube web interface during the 'Create a local project' process. The browser address bar shows 'localhost:9000/projects/create?mode=manual'. The page title is 'Create a local project'. The form contains three main sections:

- Project display name ***: A text input field containing 'DevOpsDemo-Backend' with a green checkmark icon to its right. Below the field, it says 'Up to 255 characters. Some scanners might override the value you provide.'
- Project key ***: A text input field containing 'DevOpsDemo-Backend' with a green checkmark icon to its right. Below the field, it says 'The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.'
- Main branch name ***: A text input field containing 'main'. Below the field, it says 'The name of your project's default branch [Learn More](#)'.

A blue 'Next' button is at the bottom left of the form.

Two yellow callout boxes provide additional instructions:

- A large callout box points to the 'Project key' field with the text: **Project key wählen, brauchen wir später zur Identifikation**
- A smaller callout box points to the 'Choose the baseline for new code for this project' section with the text: **Global Setting**

The 'Set up project for Clean as You Code' section is visible on the right, showing two options: 'Use the global setting' (selected) and 'Define a specific setting for this project'.

SonarQube Backend-Projekt erstellen

Tutorial

SonarQube

localhost:9000/tutorials?id=DevOpsDemo-Backend

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration More

DevOpsDemo-Backend / main

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

Analysis Method

Use this page to manage and set-up the way your analyses are performed.

How do you want to analyze your repository?

- [With Jenkins](#)
- [With GitHub Actions](#)
- [With Bitbucket Pipelines](#)
- [With GitLab CI](#)
- [With Azure Pipelines](#)
- [Other CI](#)
SonarQube integrates with your workflow no matter which CI tool you're using.
- [Locally](#)
Use this for testing or advanced use-case. Other modes are recommended to help you set up your CI environment.

Locally

SonarQube Backend-Projekt erstellen

Tutorial

Projekt erstellen und der Anleitung folgen

localhost:9000/quality_gates

localhost:9000/tutorials?id=DevOpsDemo-Backend&selectedTut: 90%

Token für Zugriff auf SonarQube

SonarQube Backend-Projekt erstellen

Tutorial

2 Run analysis on your project

What option best describes your build?

Maven

Gradle

.NET

Other (for JS, TS, Go, Python, PHP, ...)

1. Gradle wählen

Execute the Scanner for Gradle

Running an analysis with Gradle is straightforward. You just need to declare the `org.sonarqube` plugin in your `build.gradle` or `build.gradle.kts` file:

build.gradle

build.gradle.kts

```
plugins {  
    id "org.sonarqube" version "4.4.1.3373"  
}
```

You can find the latest version of the Gradle plugin [here](#).

and run the following command:

```
./gradlew sonar \  
-Dsonar.projectKey=DevOpsDemo-Backend \  
-Dsonar.projectName='DevOpsDemo-Backend' \  
-Dsonar.host.url=http://localhost:9000 \  
-Dsonar.token=sqp_2832b6cc0677481253860d7152c2f637f967a78b
```

Please visit the [official documentation of the Scanner for Gradle](#) for more details.

2. Befehl in
Terminal/Konsole
(im Ordner
backend)
ausführen

Bei Schwierigkeiten...

- Befehl direkt auf Windows Konsole oder macOS Terminal ausführen, nicht in Visual Studio Code Console
- ./ am Anfang entfernen.
- Zeilenumbrüche und «\» entfernen
- gradlew durch gradle ersetzen

```
Eingabeaufforderung
C:\mosa\dev\edu\DevOps\04b DevOpsDemo\backend>gradlew sonar -Dsonar.projectKey=DevOpsDemo-Backend -Dsonar.projectName='DevOpsDemo-Backend' -Dsonar.host.url=http://localhost:9000 -Dsonar.token=sqp_2832b6cc0677481253860d7152c2f637f967a78b

> Configure project :
The 'sonarqube' task depends on compile tasks. This behavior is now deprecated and will be removed in version 5.x. To avoid implicit compilation, set property 'sonar.gradle.skipCompile' to 'true' and make sure your project is compiled, before analysis has started.
The 'sonar' task depends on compile tasks. This behavior is now deprecated and will be removed in version 5.x. To avoid implicit compilation, set property 'sonar.gradle.skipCompile' to 'true' and make sure your project is compiled, before analysis has started.

Deprecated Gradle features were used in this build, making it incompatible with Gradle 9.0.

You can use '--warning-mode all' to show the individual deprecation warnings and determine if they come from your own scripts or plugins.

For more on this, please refer to https://docs.gradle.org/8.5/userguide/command_line_interface.html#sec:command_line_warnings in the Gradle documentation.

BUILD SUCCESSFUL in 32s
4 actionable tasks: 1 executed, 3 up-to-date
C:\mosa\dev\edu\DevOps\04b DevOpsDemo\backend>
```

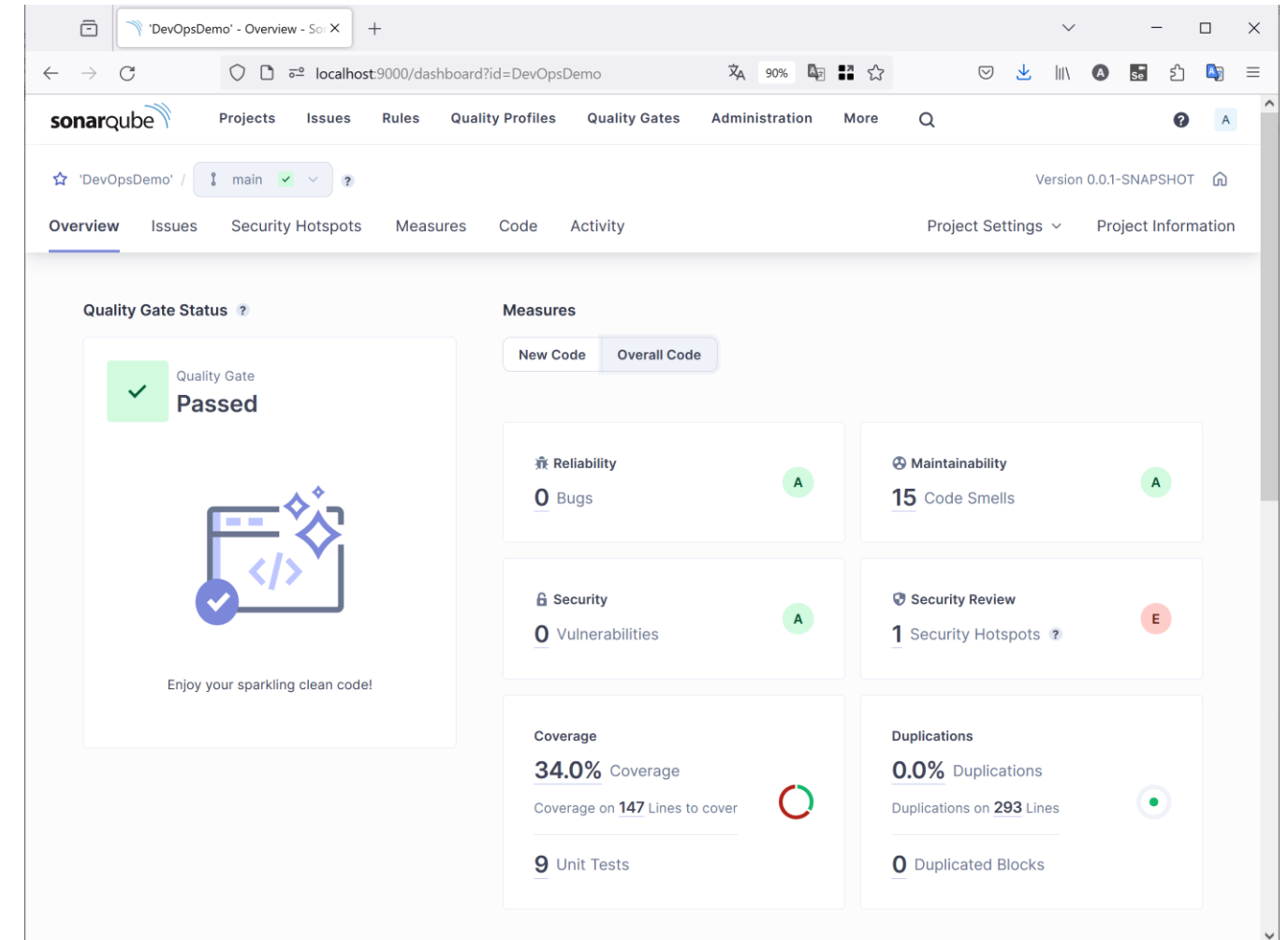
Backend: SonarQube Reports

Tutorial

Zusammenfassung

Über Gradle kann der Sonar Task gestartet werden. Folgendes passiert

- Gradle baut das Projekt
- Sonar analysiert den aktuellen Stand
- Der aktuelle Report wird in der Sonar Datenbank gespeichert



SonarQube: Frontend-Projekt erstellen

Tutorial

Analog zum Backend ein neues Projekt erstellen

Token generieren

Der Sonar Scanner ist mit NPM bereits vorinstalliert.

Ausführung mit dem folgenden Befehl im Verzeichnis «frontend»:

```
npx sonar-scanner -Dsonar.host.url=http://localhost:9000  
-Dsonar.projectKey=DevOpsDemo-Frontend -Dsonar.projectName='DevOpsDemo-Frontend'  
-Dsonar.token=sqp_2de58f399fa548fe6c44646c421b7363fa823518
```


Lernjournal

Ziele

- Code/Line Coverage erstellen und verstehen können
- Sonar Reports erstellen und verstehen können

Checkliste

- ✓ Der Code und alle Änderungen sind auf GitHub dokumentiert (Commits)
- ✓ Eigenen Code und Test schreiben
- ✓ Testabdeckung (Code Coverage) analysieren und dokumentieren
- ✓ Sonar Report für eigenes DevOpsDemo-Projekt erstellen (Backend und Frontend)
- ✓ Code-Anpassungen vornehmen und Sonar Report erneut erstellen (nur Backend)
- ✓ Dies mehrmals wiederholen und Veränderungen dokumentieren (nur Backend)
- ✓ Code Smell erstellen oder finden, dokumentieren und versuchen zu lösen, Veränderung in Sonar dokumentieren (nur Backend)