



**SOFTWARE  
ARCHITECTURE**

**2024-25**

Jose Emilio Labra Gayo

Pablo González

Irene Cid

Diego Martín



Escuela de  
Ingeniería  
Informática



Universidad de Oviedo

## Lab 6

TDD: Test-driven development

Code coverage (SonarCloud)

Continuous integration (GitHub Actions)

Static analysis tools (SonarCloud)

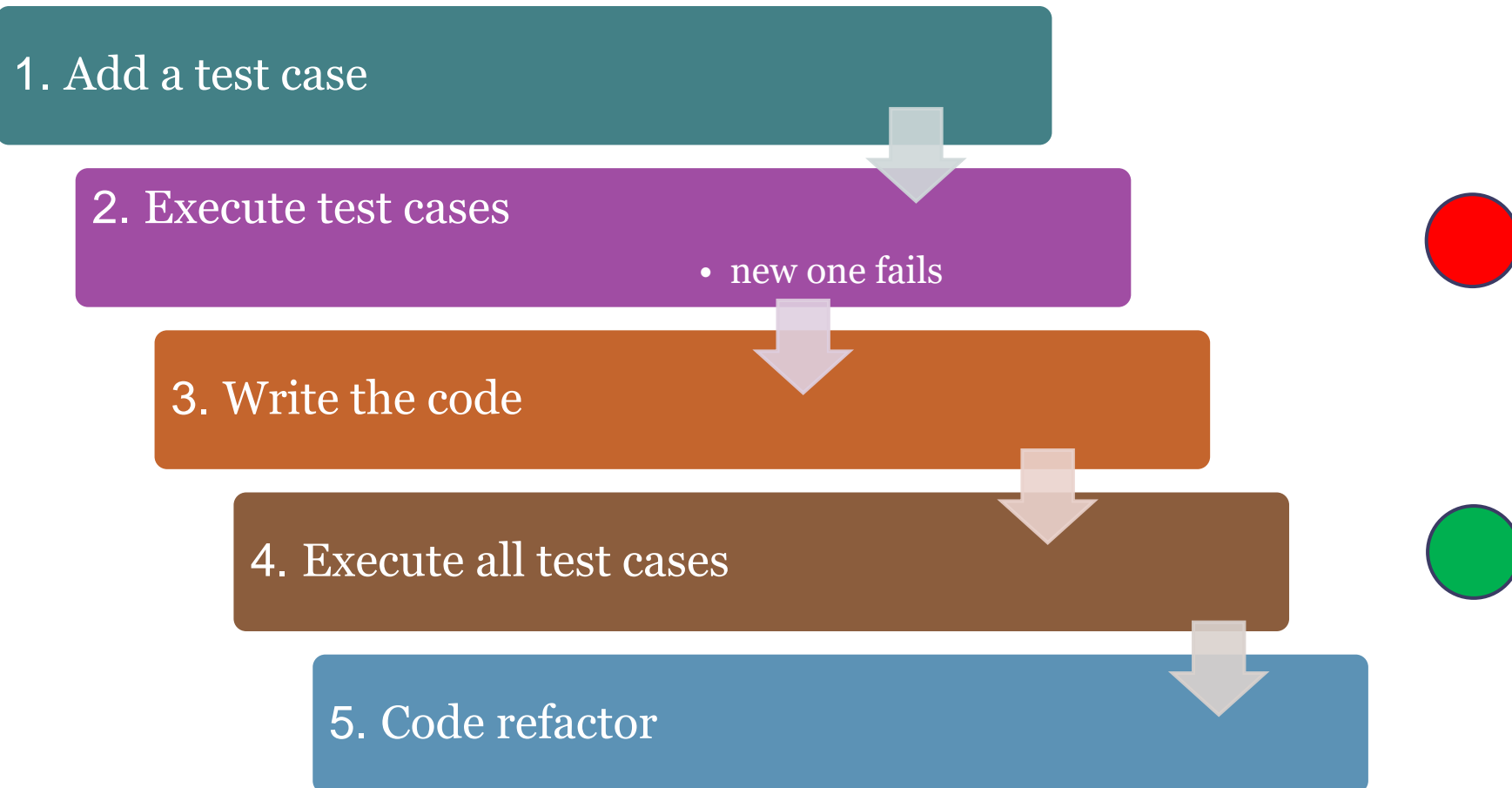
# TDD

Software development process where requirements are converted to specific test cases

The opposite to software development that allows not tested software to be deployed

Technique proposed by Kent Beck

# TDD - Phases



# TDD - Features

Simple code created to satisfy the test case

We get clean code as a result

And a test-suite

Helps focus to know what we want to implement

# Code Coverage (SonarCloud)

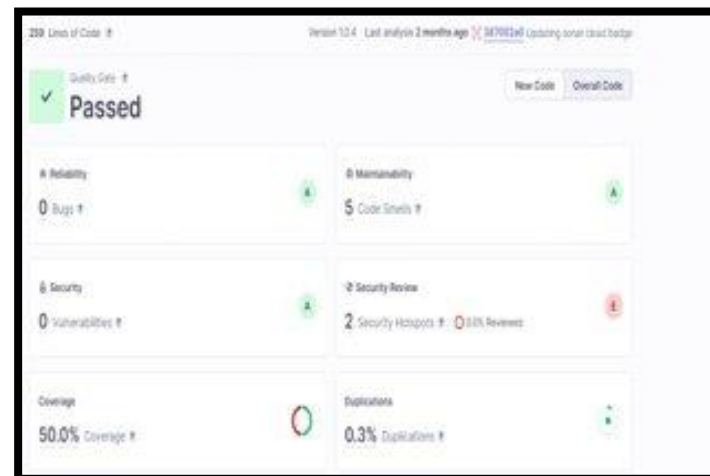
- Code coverage: Measure to show what code lines has been executed by a test suite
- Tool that includes code coverage as a metric in the code evaluation process
- Some terminology about SonarCloud:
  - LC: `lines_to_cover - uncovered_lines`
  - EL: `lines_to_cover`

# Code Coverage in SonarCloud

- Coverage ratio is calculated with the formula:

$$LC/EL$$

- After the tests, it generates a file that allows to do the analysis
  - [https://sonarcloud.io/summary/overall?id=Arquisoft\\_wichat\\_???](https://sonarcloud.io/summary/overall?id=Arquisoft_wichat_???)



# TDD - Example test

- Testing a basic component in React.js (App.test.js)

```
import { render, screen } from '@testing-library/react';
import App from './App';

test('renders welcome message', () => {
  render(<App />);
  const welcomeMessage = screen.getByText(/Welcome to the 2025 edition of the Software Architecture course/i);
  expect(welcomeMessage).toBeInTheDocument();
});
```

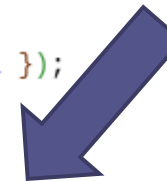
# TDD - Example test

- Checking that the AddUser component works well:
  - Sometimes we need to mock some part of the application
  - If we didn't mock the api, our test would depend on the *userservice*
  - As these are unitary tests, we simulate that part of the app

```

14  it('should add user successfully', async () => {
15      render(<AddUser />);
16
17      const usernameInput = screen.getByLabelText(/Username/i);
18      const passwordInput = screen.getByLabelText(/Password/i);
19      const addUserButton = screen.getByRole('button', { name: /Add User/i });
20
21      // Mock the axios.post request to simulate a successful response
22      mockAxios.onPost('http://localhost:8000/adduser').reply(200);
23
24      // Simulate user input
25      fireEvent.change(usernameInput, { target: { value: 'testUser' } });
26      fireEvent.change(passwordInput, { target: { value: 'testPassword' } });
27
28      // Trigger the add user button click
29      fireEvent.click(addUserButton);
30
31      // Wait for the Snackbar to be open
32      await waitFor(() => {
33          expect(screen.getByText(/User added successfully/i)).toBeInTheDocument();
34      });
35  });

```





# Continuous Integration (CI)

- Development practice that promotes developers to **integrate** code into a shared repository several times a day
- Every task to build the software is executed when some condition is met
  - For instance, a push a pull request, or the creation of a new release

# Continuous Integration (CI)

- Detect and solve problems continuously
- Always available
- Immediate execution of unit test cases and E2E tests.
- Automatic deployment
- Project quality monitorization.

# Continuous Integration (CI)

- Examples:
  - Jenkins
  - Pipeline
  - Hudson
  - Apache Continuum
  - Travis
  - **GitHub Actions**

# Continuous Integration (CI) -Uses

- Common usages:
  - Maintenance of the code in a repository
  - Building automation
  - Quick building
  - Execute test cases in a cloned production environment
  - Show results of last build.

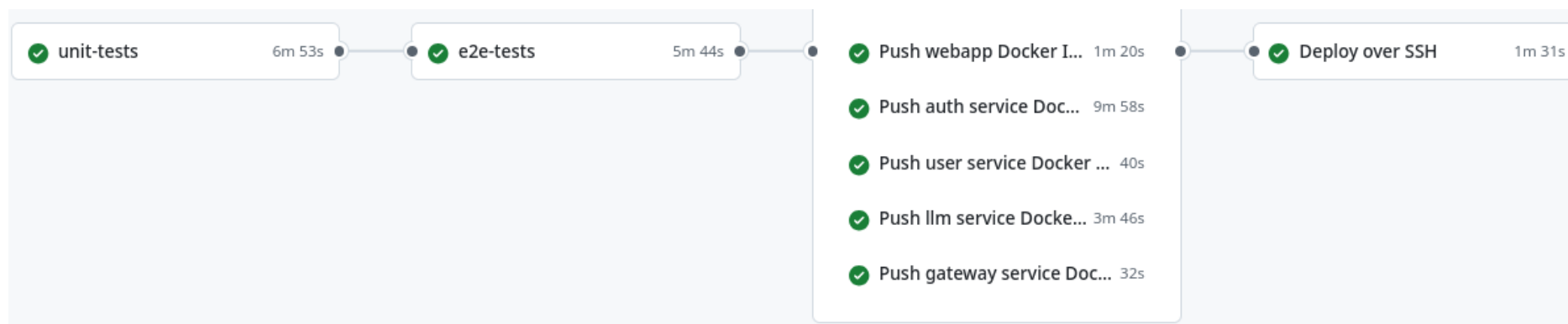
# GitHub Actions

- Continuous integration service for projects stored in GitHub
- Free for free software projects
- Configuration is in one or multiple YAML files inside the `.github/workflows` directory that is localized in the root directory of the project

# GitHub Actions

- .yml specifies:
  - Conditions for firing the process
  - List of jobs
    - Each executed in a specific environment
  - Steps to carry out the job (checkout, install dependencies, build and test)

```
jobs:
  unit-tests:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - uses: actions/setup-node@v4
        with:
          node-version: 22
      - run: npm --prefix users/authservice ci
      - run: npm --prefix users/userservice ci
      - run: npm --prefix llmservice ci
      - run: npm --prefix gatewayservice ci
      - run: npm --prefix webapp ci
      - run: npm --prefix users/authservice test -- --coverage
      - run: npm --prefix users/userservice test -- --coverage
      - run: npm --prefix llmservice test -- --coverage
      - run: npm --prefix gatewayservice test -- --coverage
      - run: npm --prefix webapp test -- --coverage
      - name: Analyze with SonarQube
        uses: SonarSource/sonarqube-scan-action@master
        env:
          SONAR_TOKEN: ${ secrets.SONAR_TOKEN }
```



# GitHub Actions

- Each job can have a specific purpose
  - Test a part of the app, deploy, etc.
- GitHub actions can be used to automate other parts of the repository.
  - Example: autoreply to new issues created in the repository

# GitHub Actions

- We have jobs also to build the docker images and publish them to github
- Check the full [documentation](#) for the CI configuration

```
docker-push-webapp:
  name: Push webapp Docker Image to GitHub Packages
  runs-on: ubuntu-latest
  permissions:
    contents: read
    packages: write
  needs: [e2e-tests]
  steps:
    - uses: actions/checkout@v4
    - name: Publish to Registry
      uses: elgohr/Publish-Docker-Github-Action@v5
      env:
        API_URI: http://${{ secrets.DEPLOY_HOST }}:8000
      with:
        name: arquisoft/wichat_0/webapp
        username: ${{{ github.actor }}}
        password: ${{{ secrets.GITHUB_TOKEN }}}
        registry: ghcr.io
        workdir: webapp
        buildargs: API_URI
```



# Static analysis of the code

Analyze the code without compiling it based in rules

Detects bugs, code smells, system vulnerabilities, etc.

Useful to control the code quality.

If the code does not meet the quality requirements, then the commit can be blocked

# Static Analysis - SonarCloud



Static code analysis tool

It needs:

- Git server like GitHub

- Repository access

- An accepted language

Two types of analysis configuration:

- Automated Analysis** (Default). Code coverage not available. Scanner running in SonarCloud servers

- CI-based analysis**. Sonar scanner running at the project server and sending reports to SonarCloud.

# Sonarlint



SonarLint detects and highlights issues that can lead to bugs, vulnerabilities, and code smells in your IDE (available in the popular ones e.g. IntelliJ, Visual Code, Visual Studio, Eclipse...)

The análisis is performed locally (before the changes are submitted to the repository), can be executed:

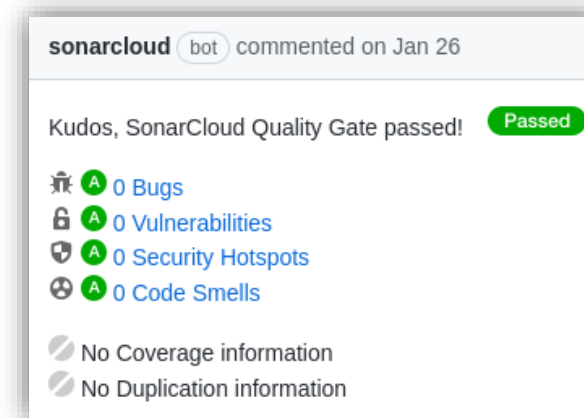
Manually

Automatically over the changed files before the commit-push.

For further details regarding supported IDEs, languages and installation instructions, please visit the [official webpage](#)

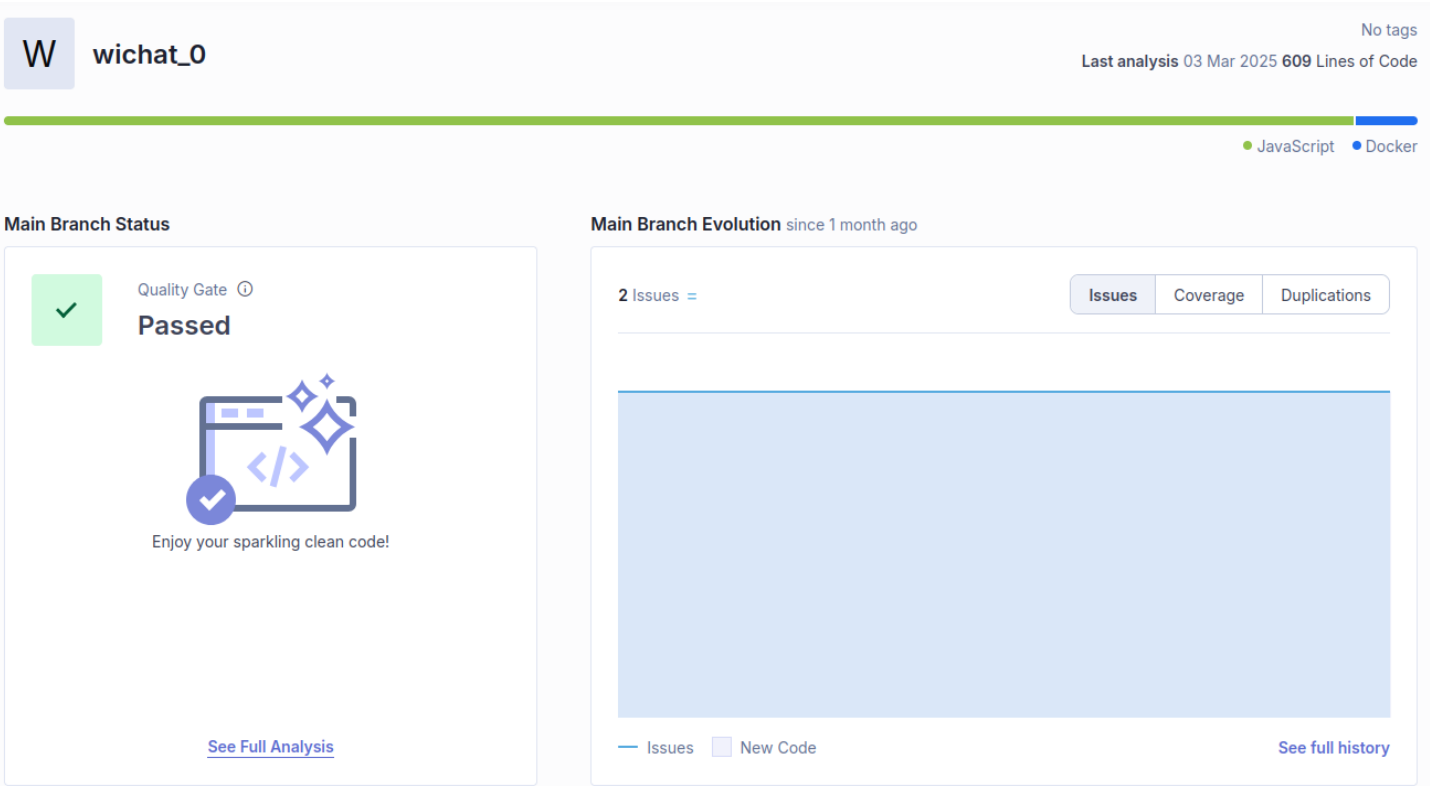
# SonarCloud - wiq\_0 configuration

After changes are pushed to the repository (example, a new pull request)  
We have information about the code quality of the pull request that we are merging to our project

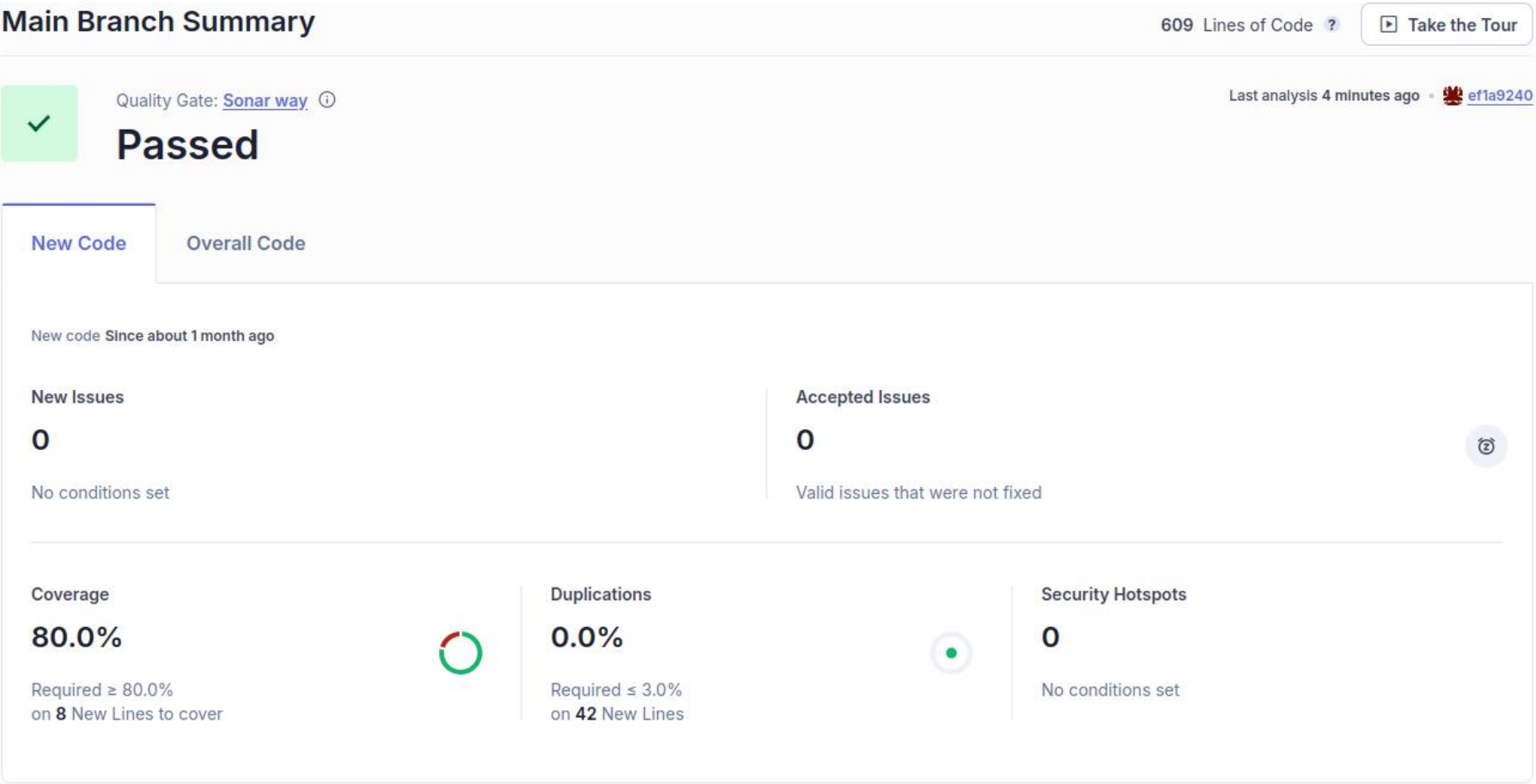


# SonarCloud

In the Project Dashboard we can check project last analysis in the main branch, pull request and specific branches



# SonarCloud: Project certification and Quality evolution



# SonarCloud: Quality Gates

At organization level, we can define the Quality Gates that our project must pass.

sonarcloud

My Projects

My Issues

+

Q

Arquitectura del Software

http://campusvirtual.uniovi.es

Key: arquisoft

Projects

Quality Profiles

Rules

Quality Gates

Members

Administration

Quality Gates

Create

aws-quality-gates

default

Sonar way

DEFAULT

BUILT-IN

aws-quality-gates

Rename

Copy

Set as Default

Delete

Conditions

Add Condition

Conditions on New Code

Conditions on New Code apply to all branches and to Pull Requests.

Metric	Operator	Value	Edit	Delete
Coverage	is less than	80.0%		
Duplicated Lines (%)	is greater than	15.0%		
Maintainability Rating	is worse than	A		
Reliability Rating	is worse than	A		
Security Hotspots Reviewed	is less than	100%		
Security Rating	is worse than	A		

Projects

Add Condition

☒ On New Code ☐ On Overall Code

Quality Gate fails when

Search for metrics...

Coverage

Condition Coverage

Conditions to Cover

Line Coverage

Lines to Cover

Uncovered Conditions

Uncovered Lines

Duplications

Duplicated Blocks

Duplicated Lines

Example AWS-Quality-Gates , we increase the procentage of duplicate lines that can be found before launch exception

# SonarCloud: Quality gates

A **Quality Gate** is a set of conditions that our project should meet.

That conditions include different aspect: code coverage, static code analyse based in rules, code duplicated, ..  
**wichat\_o** default project uses code coverage with SonarCloud



# SonarCloud: Profiles and Rules

Rules are defined at profile level

We can add, desactivate, update rules creating a new profile :

Copy a parent profile - change it - associate it to the project

The image consists of three screenshots from the SonarCloud web interface, illustrating the process of creating a new profile, setting rules, and associating it with a project.

**Left Screenshot: Profile List**  
 This screenshot shows the 'Quality Profiles' page. A list of profiles is displayed, including 'Sonar way', 'Text, 1 profile(s)', 'TypeScript, 2 profile(s)', and 'VB.NET, 1 profile(s)'. The 'TypeScript, 2 profile(s)' profile is selected. A red box highlights the 'Settings' gear icon, and a dropdown menu is shown with options: 'Compare', 'Copy', 'Extend', and 'Set as Default'. An arrow points from this menu to the 'Create a new profile' label below.

**Middle Screenshot: Profile Rules**  
 This screenshot shows the 'Rules' configuration page for the 'Sonar new Way' profile. A table lists rules with columns for 'Rules', 'Active', and 'Inactive'. The table shows counts for various rule categories like Bugs, Vulnerabilities, Code Smells, and Security Hotspots. A red box highlights the table, and an arrow points from it to the 'Set the profile rules' label below.

**Right Screenshot: Profile Projects**  
 This screenshot shows the 'Projects' configuration page for the 'Sonar new Way' profile. It indicates that no projects are explicitly associated with the profile. A red box highlights the 'Change Projects' button, and an arrow points from it to the 'Associate the profile to the project' label below.

Create a new profile

Set the profile rules

Associate the profile to the project

# Rules configuration

← → ↺

sonarcloud.io/organizations/arquisoft/rules?qprofile=AX-mgR2YnzNFv0H6nzDH&activation=true

🔍 📄 ☆ ⚙️ 👤 Paused ⋮

sonarcloud

My Projects My Issues + 🔍

type 1/1 ^ v x 🔔 ? 👁 🗺

Arquitectura del Software

http://campusvirtual.uniovi.es Key: arquisoft

Projects Quality Profiles Rules Quality Gates Members Administration

Filters

Clear All Filters

Search for rules...

Language

Type

- Bug 36
- Vulnerability 24
- Code Smell 108
- Security Hotspot 32

Tag

Repository

Default Severity

Status

Security Category

Available Since

Quality Profile SONAR N... Clear

Inheritance

Bulk Change

↑ ↓ to select rules ← → to navigate ↺ 1 / 200 rules

⬆	"===" and "!==" should be used instead of "==" and "!="	TypeScript Code Smell suspicious	⌵	Deactivate
⬆	"arguments.caller" and "arguments.callee" should not be used	TypeScript Code Smell obsolete	⌵	Deactivate
⬆	"await" should not be used redundantly	TypeScript Code Smell redundant	⌵	Deactivate
⬆	"await" should only be used with promises	TypeScript Code Smell confusing	⌵	Deactivate
⬆	"catch" clauses should do more than rethrow	TypeScript Code Smell clumsy, error-ha...	⌵	Deactivate
⬆	"default" clauses should be last	TypeScript Code Smell	⌵	Deactivate
⬆	"delete" should be used only with object properties	TypeScript Bug	⌵	Deactivate
⬆	"delete" should not be used on arrays	TypeScript Code Smell	⌵	Deactivate
⬆	"for in" should not be used with iterables	TypeScript Code Smell	⌵	Deactivate
⬆	"for of" should be used with Iterables	TypeScript Code Smell clumsy	⌵	Deactivate
⬆	"for" loop increment clauses should modify the loops' counters	TypeScript Code Smell confusing	⌵	Deactivate

# View alerts when coding

- <https://marketplace.visualstudio.com/items?itemName=SonarSource.sonarlint-vscode>

