



SonarQube : It is automatically analyse code quality and security by detecting bugs and security issue.

The screenshot shows the SonarQube IDE interface. On the left, a tree view of the project structure under 'main/java/com.example.library.Controller' shows several controller classes: AuthorController, BookController, BorrowRecordController, LibraryController, MemberController, MemberProfileController, and Dto. The AuthorController class is selected. On the right, the code editor displays the following Java code:

```

public class AuthorController {
    @Autowired
    private AuthorRepository authorRepository;

    private static final String API_KEY = "1234"; 1 usage

    @PostMapping
    public Author createAuthor(@RequestBody Author authors) {
        if(authors.getAuthor_name() == null && API_KEY=="1234"){
            throw new RuntimeException();
        }
        return authorRepository.save(authors);
    }
}

```

Below the code editor, the SonarQube interface shows the following findings:

- 5 Issues**
- Security Hotspots**
- Taint Vulnerabilities**
- Dependency Risks**
- Rule**
- Locations**

The 'Found 5 issues' section lists the following problems:

- (22, 54) Strings and Boxed types should be compared using "equals()". few seconds ago
- (23, 22) Replace generic exceptions with specific library exceptions or a custom exception type. 1 minute ago
- (4, 7) Remove this unused import 'com.example.library.Entity.Book'. 8 minutes ago
- (15, 4) Remove this field injection and use constructor injection instead. 8 minutes ago
- (1, 8) Rename this package name to match the regular expression "[a-z_]+([a-z_][a-z_]*[a-z_])". 1 minute ago

Select a finding to display the rule description

⇒ In this image you can see 5 issues are there so for solve them all first I solve Dependency injection issues so I remove field injection because it break immutability and make code harder to test so I use constructor injection to solve this issue.

The screenshot shows the SonarQube IDE interface after refactoring. The project structure and code editor remain the same, but the findings have been reduced to 4 issues:

The 'Found 4 issues' section lists the following problems:

- (26, 54) Strings and Boxed types should be compared using "equals()". 3 minutes ago
- (27, 22) Replace generic exceptions with specific library exceptions or a custom exception type. 12 minutes ago
- (4, 7) Remove this unused import 'com.example.library.Entity.Book'. 12 minutes ago
- (1, 8) Rename this package name to match the regular expression "[a-z_]+([a-z_][a-z_]*[a-z_])". 12 minutes ago

Select a finding to display the rule description

⇒ Then I solve string and box type comparison issue so using equals() I solve that and then I remove unused import and then replace generic exception run time

exception to response status exception to make meaningful and improve API quality and solve issue.

The screenshot shows a static code analysis interface. On the left, a tree view of the project structure under 'main/java/com.example.library'. The 'AuthorController' class is selected. On the right, the code editor shows the following Java code:

```
public class AuthorController {  
    private static final String API_KEY = "1234"; 1 usage  
  
    @PostMapping  
    public Author createAuthor(@RequestBody Author authors) {  
        if(authors.getAuthor_name() == null && API_KEY.equals("1234")){  
            throw new ResponseStatusException(HttpStatus.BAD_REQUEST);  
        }  
        return authorRepository.save(authors);  
    }  
}
```

Below the code editor, there are tabs for 'for IDE', 'Findings', 'Log', and 'Help & Feedback'. The 'Findings' tab is selected. It displays one finding:

found 1 issue

⚠ (1, 8) Rename this package name to match the regular expression "[a-zA-Z_]+([a-zA-Z_])"

Select a finding to display the rule description

⇒ Then at last I rename package to controller to solve last regular expression issue.

The screenshot shows the same static code analysis interface after renaming the package. The project structure now shows 'controller' instead of 'library'. The code editor shows the same Java code as before:

```
package com.example.library.controller;  
  
import com.example.library.Entity.Author;  
import com.example.library.Repository.AuthorRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.web.bind.annotation.*;  
import org.springframework.web.server.ResponseStatusException;  
  
import java.util.List;
```

Below the code editor, there are tabs for 'for IDE', 'Findings', 'Log', and 'Help & Feedback'. The 'Findings' tab is selected. It displays the message:

No findings to display

Select a finding to display the rule description

Snyk : To find and solve vulnerable issues.

The screenshot shows the Snyk interface. On the left, there's a tree view of the project structure:

- Repository
 - AuthorRepository
 - BookRepository
 - BorrowRecordRepository
 - LibraryRepository
 - MemberProfileRepository
 - MemberRepository
- Service
 - impl
 - BookService
 - MemberService
 - LibraryApplication

On the right, the code editor shows a portion of the pom.xml file with a dependency for log4j-core at version 2.14.1 highlighted. A tooltip indicates a fix is available to upgrade to version 2.25.3 and fix 5 issues.

Snyk found these issues:
All scans are completed.
Severity: C H M L

- Open Source - 2 unique vulnerabilities: 2 critical, 0 high, 0 medium, 0 low
 - 2 issues
 - 2 issues are fixable automatically.
- library/pom.xml - 2 vulnerabilities
 - C 🔍 org.apache.logging.log4j:log4j-core@2.14.1: Remote Code Execution (CVE-2021-44228)
 - C 🔍 org.apache.logging.log4j:log4j-core@2.14.1: Remote Code Execution (CVE-2021-44228)
- Code Security - No issues found

Remote Code Execution (RCE)
Issue | CVE-2021-44228 | CWE-94 | CVSS 10.0 | SNYK-JAVA-ORGAPACHELOGGINGLOG4J-2021-44228
Learn about this issue type

Vulnerable module	Introduced through
org.apache.logging.log4j:log4j-core	org.apache.logging.log4j:log4j-core@2.14.1
Fixed in	org.apache.logging.log4j:log4j-core@2.3.1, 2.12.2, 2.15.0
Exploit maturity	High

⇒ In this image you can see like one vulnerable issue is come because I use very older versions so it not safe because from this dependency any other outsider hacker hack our project so I update that version to solve this problem.

The screenshot shows the Snyk interface after the dependency update. The code editor now shows the updated pom.xml file with the log4j-core dependency set to version 2.25.3.

Snyk found these issues:
All scans are completed.
Severity: C H M L

- Open Source - No issues found
 - Congrats! No issues found!
- Code Security - No issues found
 - Congrats! No open issues found!

Select an issue and start improving your project

```

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  </properties>
  <dependencies> <a href="#">Add Starters...</a>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
      <version>3.3.2</version>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-jpa</artifactId>
      <version>3.3.2</version>
    </dependency>
  </dependencies>

```

Snyk found these issues: 40 total 0 new

All scans are completed.

Severity: C H M L

- M emp-demo\pom.xml - 3 vulnerabilities**
 - C 🔍 org.apache.tomcat.embed:tomcat-embed-core@10.1.26:3
 - C 🔍 org.apache.tomcat.embed:tomcat-embed-core@10.1.26:3
 - C 🔍 org.apache.tomcat.embed:tomcat-embed-core@10.1.26:3
- Code Security - No issues found**
 - Congrats! No issues found!
- Configuration - No issues found**
 - Configuration - No issues found!

C Time-of-check Time-of-use (TOCTOU) Race Condition

Issue | CVE-2024-50379 | CWE-367 | CVSS 9.2 | SNYK-JAVA-ORGAPACHEJETTY-8523186

Learn about this issue type

Vulnerable module	org.apache.tomcat.embed:tomcat-embed-core
Introduced through	org.springframework.boot:spring-boot-starter-web@3.3.2
Fixed in	org.apache.tomcat.embed:tomcat-embed-core@9.0.98, 10.1.34, 11.0.2
Exploit maturity	Proof of Concept

⇒ In this the version issue I solve by giving proper version to dependency.

```

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  </properties>
  <dependencies> <a href="#">Add Starters...</a>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
      <version>3.3.2</version>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-jpa</artifactId>
      <version>3.3.2</version>
    </dependency>
  </dependencies>

```

Snyk found these issues: 40 total 0 new

All scans are completed.

Severity: C H M L

- Open Source - No issues found**
 - Congrats! No issues found!
- Code Security - No issues found**
 - Congrats! No issues found!
- Configuration - No issues found**
 - Congrats! No issues found!

Select an issue and start improving your project.