1.HardcodedCredentialsExample.java:

Original code:  
A screen shot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
public class HardcodedCredentialsExample {  
 public static void main(String[] args) {  
 String username = "admin";  
 String password = "123456"; // hardcoded password  
 if(*authenticate*(username, password)) {  
 System.*out*.println("Access granted");  
 } else {  
 System.*out*.println("Access denied");  
 }  
 }  
  
 private static boolean authenticate(String user, String pass) {  
 // Dummy authentication logic  
 return user.equals("admin") && pass.equals("123456");  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 5,6  String username = "admin";  String password = "123456"; | Vulnerability | High | Hardcoded credentials: tên người dùng và mật khẩu được ghi cứng trong code |
| 8, 10  System.out.println("Access granted");  System.out.println("Access denied"); | Code Smell | Minor | Sử dụng System.out.println() thay vì Logger |
| 13 | Code Smell | Minor | So sánh mật khẩu dạng chuỗi rõ ràng (plaintext comparison) |

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Explaination:**

**\* Hardcoded Credentials**:

* Là lỗi bảo mật nghiêm trọng vì thông tin xác thực được viết cố định, dễ bị lộ khi mã nguồn bị public hoặc truy cập trái phép.
* Vi phạm quy tắc bảo mật (ví dụ: OWASP Top 10 – *Sensitive Data Exposure*).

\* **System.out.println()**:

* Không nên dùng trong ứng dụng thực tế để ghi log, vì khó quản lý mức độ log, định dạng log hoặc chuyển hướng log.
* SonarQube đánh giá đây là “Code Smell” và đề xuất dùng java.util.logging.Logger.

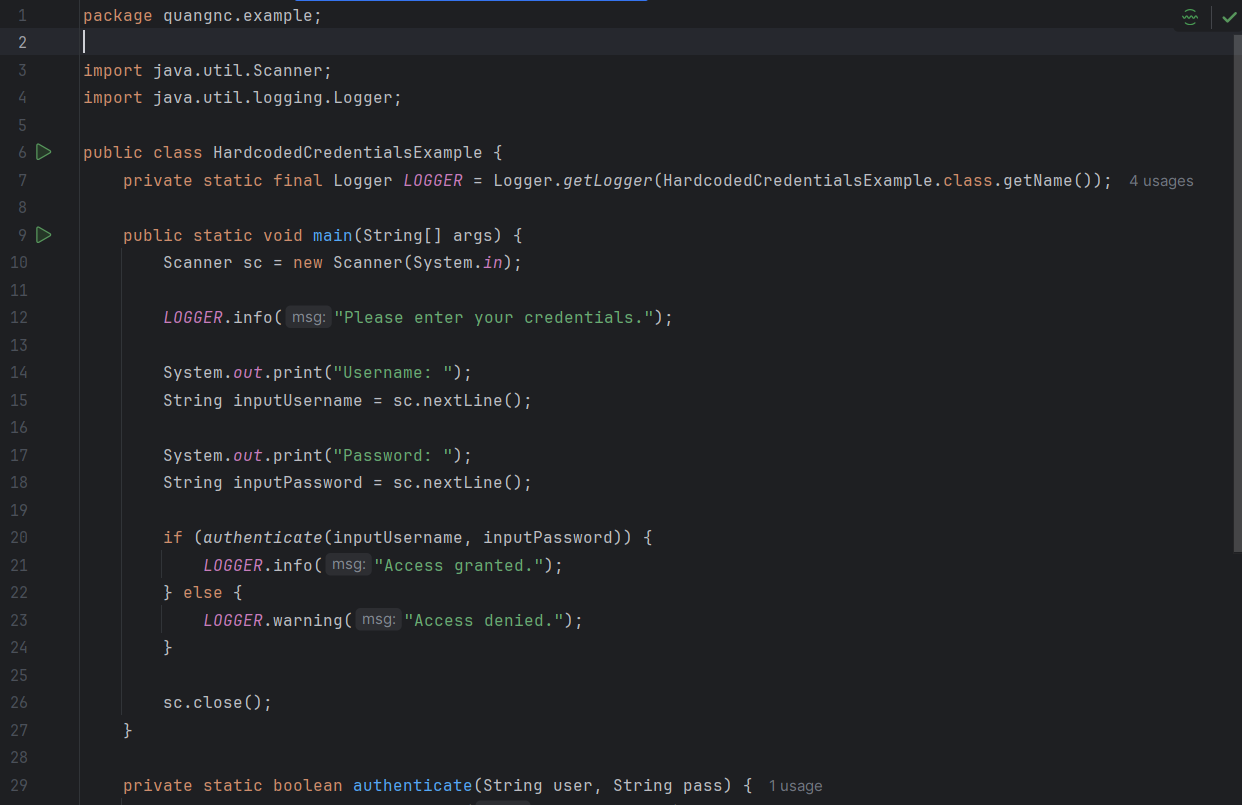
\* **So sánh chuỗi mật khẩu trực tiếp**:

* Không an toàn nếu có ai đó kiểm tra file log hoặc debug.
* Cần hash mật khẩu khi so sánh hoặc sử dụng cơ chế xác thực bảo mật hơn.

**Fix Suggestion**

* Không lưu thông tin xác thực trong code → sử dụng **biến môi trường**.
* Thay System.out.println() bằng Logger để tuân thủ chuẩn logging.
* Xác thực nên được thực hiện qua cơ chế mã hóa hoặc ít nhất không nên so sánh chuỗi trực tiếp.
* Thêm kiểm tra null nếu biến môi trường không tồn tại để tránh lỗi thời gian chạy.

Rewite code:



A computer screen with text on it

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.Scanner;  
import java.util.logging.Logger;  
  
public class HardcodedCredentialsExample {  
 private static final Logger *LOGGER* = Logger.*getLogger*(HardcodedCredentialsExample.class.getName());  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
  
 *LOGGER*.info("Please enter your credentials.");  
  
 System.*out*.print("Username: ");  
 String inputUsername = sc.nextLine();  
  
 System.*out*.print("Password: ");  
 String inputPassword = sc.nextLine();  
  
 if (*authenticate*(inputUsername, inputPassword)) {  
 *LOGGER*.info("Access granted.");  
 } else {  
 *LOGGER*.warning("Access denied.");  
 }  
  
 sc.close();  
 }  
  
 private static boolean authenticate(String user, String pass) {  
 String envUser = System.*getenv*("APP\_USERNAME");  
 String envPass = System.*getenv*("APP\_PASSWORD");  
  
 if (envUser == null || envPass == null) {  
 *LOGGER*.severe("Environment variables APP\_USERNAME or APP\_PASSWORD not set.");  
 return false;  
 }  
  
 return user.equals(envUser) && pass.equals(envPass);  
 }  
}

**Ghi chú cấu hình để chạy trong IntelliJ**

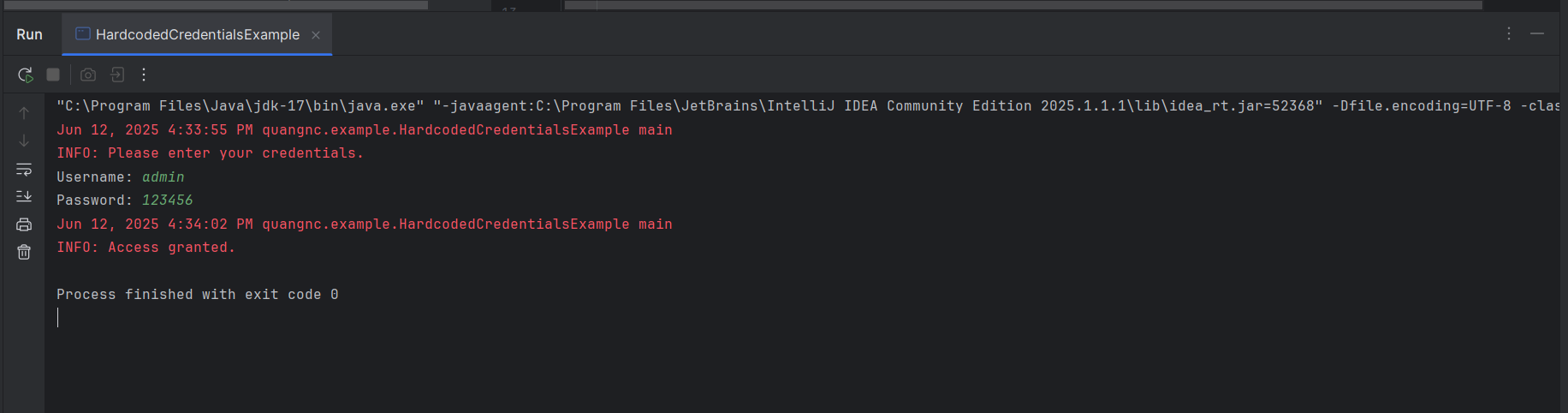
* Vào: Run > Edit Configurations...
* Trong phần **Environment Variables**, thêm:

APP\_USERNAME=admin;APP\_PASSWORD=123456

A screenshot of a computer

AI-generated content may be incorrect.

Result:



2.PathTraversalExample:  
Original code:  
A screen shot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.io.\*;  
  
public class PathTraversalExample {  
 public static void main(String[] args) throws IOException {  
 String userInput = "../secret.txt";  
 File file = new File(userInput);  
 if (file.exists()) {  
 BufferedReader reader = new BufferedReader(new FileReader(file));  
 System.*out*.println("Reading file: " + file.getPath());  
 reader.close();  
 }  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 7 String userInput = "../secret.txt"; | Vulnerability | High | Path Traversal – người dùng có thể truy cập file ngoài thư mục dự kiến |
| 10 File file = new File(userInput); | Code Smell | Minor | Không xử lý ngoại lệ (ví dụ file không đọc được |
| 11 System.*out*.println("Reading file: " + file.getPath()); | Code Smell | Minor | Sử dụng System.out.println() thay vì Logger |

Explaination:

**🔸 Path Traversal (Lỗi bảo mật nghiêm trọng):**

* Khi người dùng cung cấp đường dẫn như "../../../../etc/passwd" hoặc "../secret.txt" → họ có thể truy cập **file nhạy cảm** ngoài thư mục an toàn.
* Lỗi này nằm trong **OWASP Top 10** (Insecure Direct Object References / IDOR / Path Traversal).
* Hacker có thể trích xuất cấu hình hệ thống, mật khẩu, hoặc mã nguồn nội bộ.

**🔸 Không kiểm tra hợp lệ đường dẫn:**

* Không xác thực input trước khi dùng.
* Không giới hạn truy cập vào "safe directory".

**🔸 System.out:**

* Không phù hợp với ứng dụng thực tế hoặc khi dùng logging framework.

**Fix Suggestion:**

* Chỉ cho phép truy cập file nằm **trong thư mục hợp lệ** (whitelist).
* **Normalize** và kiểm tra đường dẫn để tránh ../ lách ra ngoài thư mục gốc.
* Dùng Logger thay cho System.out.

Rewite code:  
  
A screen shot of a computer program

AI-generated content may be incorrect.

A computer screen shot of text

AI-generated content may be incorrect.  
package quangnc.example;  
  
import java.io.\*;  
import java.util.logging.Logger;  
  
public class PathTraversalExample {  
  
 // Tạo Logger thay vì dùng System.out để đúng theo cảnh báo của SonarQube  
 private static final Logger *LOGGER* = Logger.*getLogger*(PathTraversalExample.class.getName());  
  
 public static void main(String[] args) {  
 // → Giúp tránh lỗi path traversal (cố thoát khỏi thư mục an toàn)  
 String userInput = "secret.txt";  
  
 // Cập nhật đường dẫn thư mục an toàn thành src/main/resources  
 // → Nơi chứa file hợp lệ để đọc  
 File safeDirectory = new File("src/main/resources");  
  
 try {  
 // Ghép safeDirectory + userInput rồi chuẩn hóa (loại bỏ ../, vv)  
 File targetFile = new File(safeDirectory, userInput).getCanonicalFile();  
  
 // Kiểm tra nếu file nằm ngoài thư mục an toàn → log lỗi và thoát  
 if (!targetFile.getPath().startsWith(safeDirectory.getCanonicalPath())) {  
 *LOGGER*.severe("Invalid file access attempt: " + targetFile.getPath());  
 return;  
 }  
  
 // Nếu file tồn tại → đọc từng dòng và in ra bằng LOGGER  
 if (targetFile.exists()) {  
 BufferedReader reader = new BufferedReader(new FileReader(targetFile));  
 *LOGGER*.info("Reading file: " + targetFile.getPath());  
  
 // Dòng mới: in từng dòng file ra log (tránh dùng System.out)  
 reader.lines().forEach(*LOGGER*::info);  
 reader.close();  
 } else {  
 // File không tồn tại → cảnh báo  
 *LOGGER*.warning("File not found: " + targetFile.getPath());  
 }  
  
 } catch (IOException e) {  
 // Bắt và ghi log lỗi khi đọc file  
 *LOGGER*.severe("Error reading file: " + e.getMessage());  
 }  
 }  
}

Result:  
A screenshot of a computer

AI-generated content may be incorrect.

3. ResourceLeakExample.java:  
Original code:  
A screen shot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.io.\*;  
  
public class ResourceLeakExample {  
 public static void main(String[] args) {  
 try {  
 BufferedReader reader = new BufferedReader(new FileReader("data.txt"));  
 String line;  
 while ((line = reader.readLine()) != null) {  
 System.*out*.println(line);  
 }  
   
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 6 BufferedReader reader = ... | Resource Leak | High | BufferedReader được mở nhưng không được đóng → có thể gây rò rỉ tài nguyên |
| 11 System.out.println(line); | Coding Style | Medium | Dùng System.out.println thay vì Logger → không kiểm soát được log output |
| 15 e.printStackTrace(); | Code Smell | Medium | Không nên in trực tiếp stack trace trong production |

A screenshot of a computer

AI-generated content may be incorrect.

Explain Error:

* BufferedReader mở file nhưng **không đóng lại** sau khi dùng.  
  → Nếu chương trình chạy nhiều lần hoặc trong hệ thống lớn, điều này gây:
  + **Rò rỉ tài nguyên** (file handle)
  + **Chặn truy cập file khác**
  + **Lỗi bảo mật hoặc treo ứng dụng**
* System.out.println và e.printStackTrace() nên thay bằng Logger để dễ kiểm soát log.

**Fix Suggestion:**

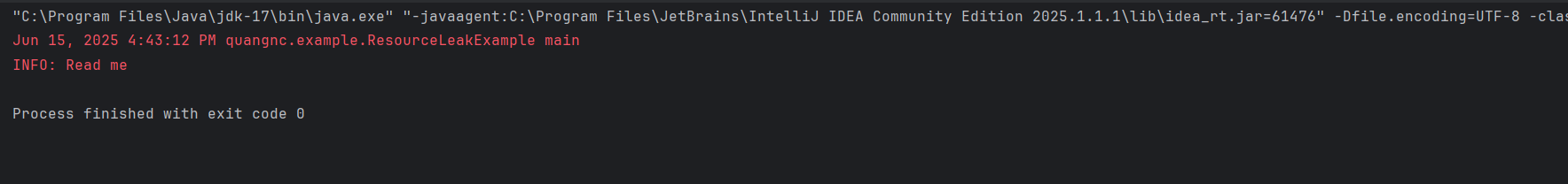
* **Dùng try-with-resources (Java 7+) để tự động đóng BufferedReader.**
* Thay System.out.println bằng Logger.
* Ghi log lỗi thay vì in stack trace.

Rewrite code:

A screen shot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.io.\*;  
import java.util.logging.Level;  
import java.util.logging.Logger;  
  
public class ResourceLeakExample {  
 private static final Logger *LOGGER* = Logger.*getLogger*(ResourceLeakExample.class.getName());  
  
 public static void main(String[] args) {  
 File dataFile = new File("data.txt");  
  
 // Dùng try-with-resources để đảm bảo file được đóng sau khi dùng  
 try (BufferedReader reader = new BufferedReader(new FileReader(dataFile))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 *LOGGER*.info(line); // Ghi log thay vì System.out  
 }  
 } catch (IOException e) {  
 *LOGGER*.log(Level.*SEVERE*, String.*format*("Error reading file: %s", e.getMessage()), e);// Ghi log lỗi  
 }  
 }  
}

Result:  


4. SQLInjectionExample.java:

Original code:

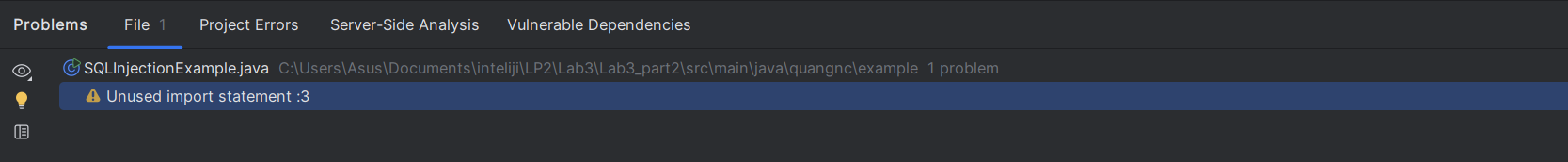
A screen shot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.sql.\*;  
  
 public class SQLInjectionExample {  
 public static void main(String[] args) {  
 String userInput = "' OR '1'='1";  
 String query = "SELECT \* FROM users WHERE username = '" + userInput + "'";  
 System.*out*.println("Executing query: " + query);  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 3 import java.sql.\*; | Code Smell | Minor | **Unused import statement** – không có lớp nào trong java.sql được dùng trong mã |
| 7 String userInput = "' OR '1'='1"; | Vulnerability | High | Input giả lập tấn công SQL Injection – có thể khai thác để truy cập trái phép dữ liệu |
| 8 String query = "SELECT \* FROM users WHERE username = '" + userInput + "'"; | Vulnerability | High | Ghép chuỗi SQL trực tiếp với đầu vào – dễ bị chèn lệnh SQL độc hại |
| 9 System.out.println("Executing query: " + query); | Code Smell | Minor | Sử dụng System.out thay vì Logger |



**Explaination:**

**🔸 Unused import:**

* **import java.sql.\*; được khai báo nhưng không có lớp nào trong java.sql được sử dụng → cảnh báo thừa.**

🔸 **SQL Injection (dòng 7 & 8):**

* userInput = "' OR '1'='1" là một ví dụ điển hình cho tấn công SQL Injection.
* Khi chèn vào câu lệnh:

SELECT \* FROM users WHERE username = '' OR '1'='1'

→ Luôn đúng → truy xuất toàn bộ dữ liệu người dùng hoặc qua mặt đăng nhập.

🔸 **Ghép chuỗi SQL (dòng 8):**

* Việc xây dựng câu SQL bằng cách nối chuỗi với input **là cực kỳ nguy hiểm**, dễ bị khai thác nếu input không được kiểm tra.

🔸 **System.out:**

* Không dùng được trong môi trường sản xuất (production).
* Không kiểm soát mức log (info, warning, error...).
* Khó chuyển hướng ra file hoặc log server.

**Fix Suggestion:**

* Loại bỏ import không dùng.
* Dùng PreparedStatement để chống SQL Injection.
* Dùng Logger thay vì System.out.

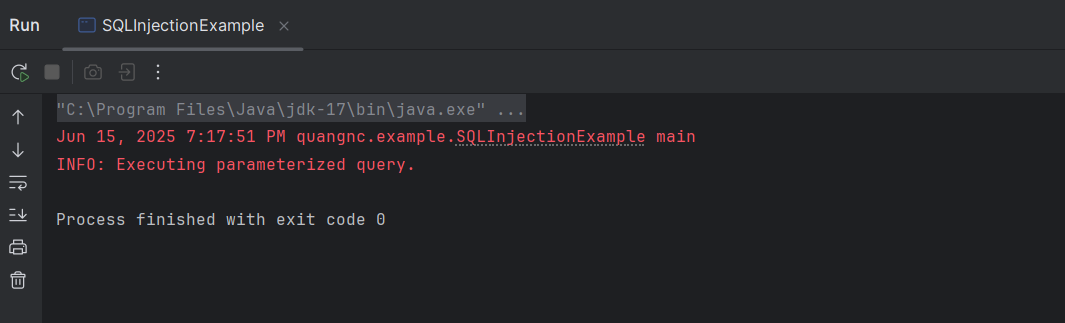
Rewrite code:  
A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer screen

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.sql.\*;  
import java.util.logging.Logger;  
  
public class SQLInjectionExample {  
 private static final Logger *LOGGER* = Logger.*getLogger*(SQLInjectionExample.class.getName());  
  
 public static void main(String[] args) {  
 String userInput = "' OR '1'='1"; // Dữ liệu đầu vào nguy hiểm mô phỏng  
  
 // Đọc thông tin kết nối từ biến môi trường  
 String dbUser = System.*getenv*("DB\_USER");  
 String dbPass = System.*getenv*("DB\_PASS");  
  
 // Thêm dòng này để tắt SSL  
 String url = "jdbc:sqlserver://localhost:1433;databaseName=mydb;encrypt=false";  
  
 try (Connection conn = DriverManager.*getConnection*(url, dbUser, dbPass)) {  
  
 String sql = "SELECT username FROM users WHERE username = ?";  
 try (PreparedStatement stmt = conn.prepareStatement(sql)) {  
 stmt.setString(1, userInput);  
  
 *LOGGER*.info("Executing parameterized query.");  
  
 try (ResultSet rs = stmt.executeQuery()) {  
 while (rs.next()) {  
 *LOGGER*.info("Found user: " + rs.getString("username"));  
 }  
 }  
 }  
  
 } catch (SQLException e) {  
 *LOGGER*.severe("Database error: " + e.getMessage());  
 }  
 }  
}

Result:  


5.UnimplementedInterfaceExample.java

A screenshot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
interface Drawable {  
 void draw();  
}  
  
class Circle implements Drawable {  
 // Missing draw() implementation → compile error  
}

:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 7 class Circle implements Drawable | Compile Error | High | Class'Circle' phải được khai báo là abstract hoặc phải triển khai phương thức 'draw()'. |
| 4 void draw(); | Code Smell | Minor | Method 'draw()' chưa được dùng. |
| 7 class Circle implements Drawable | Code Smell | Minor | Class 'Circle' chưa được dung. |

**Explanation:**

* 🔸 **Interface not fully implemented (Compile Error)**  
  → Interface Drawable yêu cầu mọi class implement nó phải định nghĩa phương thức draw().  
  → Circle chưa định nghĩa draw() → lỗi biên dịch.
* 🔸 **Code Smell – Không sử dụng**  
  → Không có chỗ nào gọi draw() hay sử dụng class Circle, dẫn đến cảnh báo từ IntelliJ hoặc SonarQube.

**Fix Suggestion:**

* Cần override draw() trong class Circle.
* Nếu muốn thử chạy thì nên thêm main() để dùng thử đối tượng Circle.

Rewrite code:  
A screen shot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.logging.Logger;  
  
interface Drawable {  
 void draw();  
}  
  
class Circle implements Drawable {  
 // Tạo logger cho lớp Circle  
 private static final Logger *LOGGER* = Logger.*getLogger*(Circle.class.getName());  
  
 @Override  
 public void draw() {  
 *LOGGER*.info("Drawing a circle.");  
 }  
}  
  
public class UnimplementedInterfaceExample {  
 public static void main(String[] args) {  
 Drawable d = new Circle();  
 d.draw();  
 }  
}

Result:

A screenshot of a computer

AI-generated content may be incorrect.

6.InterfaceFieldModificationExample.java:

Original code:  
A screen shot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
interface Constants {  
 int *MAX\_USERS* = 100;  
}  
  
public class InterfaceFieldModificationExample {  
 public static void main(String[] args) {  
 // Constants.MAX\_USERS = 200; // Compile-time error  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 9 // Constants.MAX\_USERS = 200; | Compile-time | High | Không thể gán giá trị cho biến hằng MAX\_USERS vì nó là final trong interface |
| 4 int MAX\_USERS = 100; | Code Smell | Minor | Trường MAX\_USERS không được sử dụng trong chương trình |
| 7 class Circle implements Drawable | Code Smell | Minor | Class 'Circle' chưa được dung. |

Explain:

* Trong Java, mọi biến trong interface đều **ngầm định là public static final**: public static final int MAX\_USERS = 100;
* Do đó, dòng gán: Constants.MAX\_USERS = 200;

sẽ gây ra lỗi biên dịch: cannot assign a value to final variable 'MAX\_USERS'

* Ngoài ra, vì interface Constants **chưa được sử dụng** trong main() hoặc ở nơi khác nên IntelliJ sẽ cảnh báo **“Interface is never used”** và **“Field is never used”**.

Fix Suggestion:

* Nếu muốn thay đổi giá trị, hãy chuyển MAX\_USERS sang một lớp thông thường có setter hoặc sử dụng biến không final
* Nếu không cần interface này nữa thì nên xóa hoặc sử dụng nó đúng mục đích
* Hãy dùng MAX\_USERS trong một đoạn logic để tránh cảnh báo
* Nên đặt hằng số trong lớp Constants dạng class hoặc enum thay vì interface

Rewrite code:

A screenshot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.logging.Logger;  
  
// Dùng class thay vì interface  
class Constants {  
 public static final int *MAX\_USERS* = 100;  
}  
  
public class InterfaceFieldModificationExample {  
 private static final Logger *LOGGER* = Logger.*getLogger*(InterfaceFieldModificationExample.class.getName());  
  
 public static void main(String[] args) {  
  
 for (int i = 1; i <= Constants.*MAX\_USERS*; i += 50) {  
 *LOGGER*.info("Processing user batch: " + i);  
 }  
 }  
}

Result:  
A screenshot of a computer

AI-generated content may be incorrect.

7.MissingOverrideAnnotationExample.java:  
A screenshot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
class Animal {  
 void speak() {  
 System.*out*.println("Animal speaks");  
 }  
}  
  
class Dog extends Animal {  
 void Speak() {   
 System.*out*.println("Dog barks");  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 9 class Dog | Method never used | Minor | Class Dog chưa bao giờ được sử dụng |
| 10 void Speak() {   System.*out*.println("Dog barks"); } | Method not overriding | Major | Phương thức Speak() không override đúng speak() của lớp cha do sai chữ hoa/thường |
| 10 | Missing @Override annotation | Minor | Trường MAX\_USERS không được sử dụng trong chương trình |
| 4 void speak() | Method never used | Minor | Phương thức speak() chưa được gọi ở đâu cả |
| 10 | Method never used | Minor | Phương thức Speak() không được gọi ở đâu cả |

A screenshot of a computer program

AI-generated content may be incorrect.

**Fix Suggestion:**

* **Sửa tên phương thức** Speak() → speak() để đúng với phương thức trong Animal.
* **Thêm annotation** @Override để trình biên dịch kiểm tra đúng override.
* **Tạo đối tượng Dog và gọi phương thức speak()** trong hàm main() để sử dụng lớp này và kiểm tra kết quả override.

Rewrite code:  
A screenshot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.logging.Logger;  
  
class Animal {  
 void speak() {  
 Logger.*getLogger*(Animal.class.getName()).info("Animal speaks");  
 }  
}  
  
class Dog extends Animal {  
 @Override  
 void speak() {  
 Logger.*getLogger*(Dog.class.getName()).info("Dog barks");  
 }  
}  
  
public class MissingOverrideAnnotationExample {  
 public static void main(String[] args) {  
 Animal a = new Dog();  
 a.speak();  
 }  
}

Result:  
A screenshot of a computer

AI-generated content may be incorrect.

8.CatchGenericExceptionExample.java:

Original code:

A screenshot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
public class CatchGenericExceptionExample {  
 public static void main(String[] args) {  
 try {  
 String s = null;  
 System.*out*.println(s.length());  
 } catch (Exception e) {  
 System.*out*.println("Something went wrong");   
 }  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 8 } catch (Exception e) { | Catch generic exception | Medium | Catch Exception tổng quát có thể che giấu lỗi cụ thể như NullPointerException. |
| 9 System.*out*.println("Something went wrong"); | Message không cụ thể | Minor | Dòng System.out.println("Something went wrong"); không hiển thị thông tin lỗi cụ thể. |
| 9 | Code Smell | Minor | Sử dụng System.out thay vì Logger |

A screenshot of a computer

AI-generated content may be incorrect.

Fix Suggestion:

* **Bắt lỗi cụ thể (NullPointerException) thay vì Exception chung chung.**
* In rõ thông báo lỗi bằng Logger, tránh System.out.println.

Rewrite code:

A screen shot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.logging.Logger;  
  
public class CatchGenericExceptionExample {  
 private static final Logger *LOGGER* = Logger.*getLogger*(CatchGenericExceptionExample.class.getName());  
  
 public static void main(String[] args) {  
 try {  
 String s = null;  
 System.*out*.println(s.length()); // dòng này để cố tình gây lỗi NullPointerException  
 } catch (NullPointerException e) {  
 *LOGGER*.severe("Đã xảy ra lỗi NullPointerException: " + e.getMessage());  
 }  
 }  
}

Result:  
A screen shot of a computer

AI-generated content may be incorrect.

9.NullPointerExample.java:

Original code:

A screenshot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
public class NullPointerExample {  
 public static void main(String[] args) {  
 String text = null;  
 if (text.length() > 0) {   
 System.*out*.println("Text is not empty");  
 }  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 5 if (text.length() > 0) | Vulnerability | High | Gọi phương thức length() trên biến có thể là null, gây NullPointerException. |
| 5 if (text.length() > 0) | Vulnerability | High | Lỗi tiềm ẩn: đoạn gọi text.length() sẽ ném ra lỗi nếu text == null. |
| 5 if (text.length() > 0) | Code Smell | Minor | Có thể thay thế bằng !text.isEmpty() để rõ ràng và chuẩn hơn. |
| 5 if (text.length() > 0) | Code Smell | Minor | Nên kiểm tra null trước khi truy cập thuộc tính chuỗi. |
| 6 System.out.println(...) | Code Smell | Minor | Không nên dùng System.out.println() trong ứng dụng thực tế, dùng Logger thay thế. |

A screenshot of a computer

AI-generated content may be incorrect.

**Fix Suggestion:**

* Thêm kiểm tra text != null trước khi gọi .length() hoặc .isEmpty().
* Thay length() > 0 bằng !text.isEmpty().
* Dùng Logger thay vì System.out.

Rewrite code:

A screenshot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.logging.Logger;  
  
public class NullPointerExample {  
 private static final Logger *LOGGER* = Logger.*getLogger*(NullPointerExample.class.getName());  
  
 public static void main(String[] args) {  
 String text = *fetchText*();  
  
 if (text != null && !text.isEmpty()) {  
 *LOGGER*.info("Text is not empty");  
 } else {  
 *LOGGER*.warning("Chuỗi đang null hoặc rỗng");  
 }  
 }  
  
 // Mô phỏng một hàm trả về null hoặc chuỗi thực tế  
 private static String fetchText() {  
 return Math.*random*() > 0.5 ? "Hello" : null;  
 }  
}

Result:  
A screenshot of a computer program

AI-generated content may be incorrect.

10.OvercatchingExceptionExample.java:

Original code:

A screenshot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
public class OvercatchingExceptionExample {  
 public static void main(String[] args) {  
 try {  
 int[] arr = new int[5];  
 System.*out*.println(arr[10]);   
 } catch (RuntimeException e) {  
 System.*out*.println("Runtime error");   
 }  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 8 } catch (RuntimeException e) { | Overcatching Exception | Medium | Bắt RuntimeException quá rộng, có thể che giấu lỗi cụ thể như ArrayIndexOutOfBoundsException |
| 7 System.*out*.println(arr[10]); | Code Smell | Minor | Nên kiểm tra null trước khi truy cập thuộc tính chuỗi. |
| 9 System.*out*.println("Runtime error"); | Code Smell | Minor | Không nên dùng System.out.println() trong ứng dụng thực tế, dùng Logger thay thế. |

A screenshot of a computer program

AI-generated content may be incorrect.

**Fix Suggestion:**

* Bắt **ngoại lệ cụ thể hơn** thay vì RuntimeException → giúp code rõ ràng, dễ bảo trì.
* Dùng **Logger** để log lỗi thay vì System.out.

Rewrite code:

A screenshot of a computer screen

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.logging.Logger;  
  
public class OvercatchingExceptionExample {  
 private static final Logger *LOGGER* = Logger.*getLogger*(OvercatchingExceptionExample.class.getName());  
  
 public static void main(String[] args) {  
 int[] arr = {1, 2, 3, 4, 5};  
  
 // indexToAccess thay đổi theo tham số dòng lệnh để IDE không chắc chắn giá trị  
 int indexToAccess = args.length > 0 ? Integer.*parseInt*(args[0]) : 2;  
  
 if (indexToAccess >= 0 && indexToAccess < arr.length) {  
 *LOGGER*.info("Giá trị tại vị trí " + indexToAccess + ": " + arr[indexToAccess]);  
 } else {  
 *LOGGER*.warning("Chỉ số " + indexToAccess + " vượt quá giới hạn mảng.");  
 }  
 }  
}

Result:

A screenshot of a computer

AI-generated content may be incorrect.

11.TightCouplingExample.java:

Original code:

A screenshot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
class Printer {  
 void print(String message) {  
 System.*out*.println(message);  
 }  
}  
  
class Report {  
 private Printer printer = new Printer(); // tightly coupled  
  
 void generate() {  
 printer.print("Generating report...");  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 9 class Report | Design Flaw | Medium | Report phụ thuộc chặt vào Printer khiến việc mở rộng hoặc thay đổi khó khăn (tight coupling). |
| 4 {  System.*out*.println(message); } | Code Smell | Minor | Sử dụng System.out.println thay vì Logger. |
| 10 private Printer printer = new Printer(); | Code Smell | Minor | Trường printer được khởi tạo trực tiếp thay vì tiêm phụ thuộc (dependency injection). |

A screenshot of a computer

AI-generated content may be incorrect.

**Fix Suggestion:**

* Áp dụng **Dependency Injection** để giảm tight coupling (phụ thuộc cứng).
* Thay System.out.println bằng **Logger** để quản lý log hiệu quả hơn và tránh cảnh báo.
* Cải thiện khả năng kiểm thử và mở rộng bằng cách **tách interface**.

Rewrite code:  
A screen shot of a computer program

AI-generated content may be incorrect.  
A screen shot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.logging.Logger;  
  
interface Printer {  
 void print(String message);  
}  
  
class ConsolePrinter implements Printer {  
 private static final Logger *LOGGER* = Logger.*getLogger*(ConsolePrinter.class.getName());  
  
 @Override  
 public void print(String message) {  
 *LOGGER*.info(message);  
 }  
}  
  
class Report {  
 private final Printer printer;  
  
 // Constructor Injection  
 public Report(Printer printer) {  
 this.printer = printer;  
 }  
  
 void generate() {  
 printer.print("Generating report...");  
 }  
}  
  
public class TightCouplingExample {  
 public static void main(String[] args) {  
 Printer printer = new ConsolePrinter();  
 Report report = new Report(printer);  
 report.generate();  
 }  
}

Result:

A screenshot of a computer program

AI-generated content may be incorrect.

12.ViolationOfEncapsulationExample.java:

Original code:

A screenshot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
class User {  
 public String name;   
 public int age;  
  
 public void display() {  
 System.*out*.println("Name: " + name + ", Age: " + age);  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 3 class User | Code Smell | Minor | Class User chưa được dùng |
| 4 public String name; | Code Smell | High | Trường name bị khai báo public, vi phạm nguyên lý đóng gói (encapsulation). |
| 5 public int age; | Code Smell | High | Trường age bị khai báo public, vi phạm nguyên lý đóng gói. |
| 7 public void display() {  System.*out*.println("Name: " + name + ", Age: " + age); } | Code Smell | Minor | Sử dụng System.out.println thay vì Logger, không phù hợp với hệ thống lớn. |

A screenshot of a computer

AI-generated content may be incorrect.

**Fix Suggestion:**

* Sử dụng từ khóa private cho các biến thành viên.
* Cung cấp các phương thức getter và setter để truy cập và cập nhật dữ liệu theo cách có kiểm soát.
* Dùng Logger thay vì System.out.println để ghi log.

Rewrite code:

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.logging.Logger;  
  
class User {  
 private static final Logger *LOGGER* = Logger.*getLogger*(User.class.getName());  
  
 private String name;  
 private int age;  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public int getAge() {  
 return age;  
 }  
  
 public void setAge(int age) {  
 if (age >= 0) {  
 this.age = age;  
 } else {  
 *LOGGER*.warning("Tuổi không hợp lệ!");  
 }  
 }  
  
 public void display() {  
 *LOGGER*.info("Name: " + name + ", Age: " + age);  
 }  
}  
  
public class ViolationOfEncapsulationExample {  
 public static void main(String[] args) {  
 User user = new User();  
 user.setName("Alice");  
 user.setAge(25);  
 user.display();  
  
 int currentAge = user.getAge(); // sử dụng getAge để tránh cảnh báo  
 Logger.*getLogger*(ViolationOfEncapsulationExample.class.getName()).info("Tuổi hiện tại: " + currentAge);  
 }  
}

Result:

A screenshot of a computer program

AI-generated content may be incorrect.

13. UnreachableCodeExample.java:

Original code:  
A screenshot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
public class UnreachableCodeExample {  
 public static int getNumber() {  
 return 42;  
 System.*out*.println("This will never execute");  
 }  
  
 public static void main(String[] args) {  
 System.*out*.println(*getNumber*());  
 }  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 6 System.*out*.println("This will never execute"); | Unreachable Code | High | System.out.println("This will never execute"); sẽ không bao giờ được chạy vì nằm sau return. |

A screenshot of a computer

AI-generated content may be incorrect.

**Fix Suggestion:**

* Di chuyển hoặc xóa dòng không thể truy cập sau return.

Rewrite code:

A screenshot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.logging.Logger;  
  
public class UnreachableCodeExample {  
 private static final Logger *LOGGER* = Logger.*getLogger*(UnreachableCodeExample.class.getName());  
 private static final int *NUMBER* = 42;  
 // System.out.println("This will never execute"); // Đã bị xóa vì không bao giờ chạy  
  
  
 public static void main(String[] args) {  
 *LOGGER*.info("Số trả về là: " + *NUMBER*);  
 }  
}

Result:

A screenshot of a computer

AI-generated content may be incorrect.

14.IncompleteInterfaceImplementationExample.java:

Original code:

A screenshot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
interface Shape {  
 void draw();  
 void resize();  
}  
  
class Square implements Shape {  
 public void draw() {  
 System.*out*.println("Drawing square");  
 }  
   
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 8 class Square implements Shape | Code Smell | High | Class Square chưa triển khai đầy đủ phương thức từ interface Shape. Cụ thể, thiếu resize().   |  | | --- | |  |  |  | | --- | |  | |
| 8 class Square implements Shape | Code Smell | Low | Class Square chưa được sử dụng . |
| 5 void resize(); | Code Smell | Medium | Phương thức resize() trong interface Shape chưa được triển khai ở lớp Square. |
| 9 public void draw() {  System.*out*.println("Drawing square"); } | Code Smell | Minor | Sử dụng System.out.println thay vì Logger, không phù hợp với hệ thống lớn. |

A screenshot of a computer

AI-generated content may be incorrect.

**Fix Suggestion:**

* Triển khai đầy đủ tất cả các phương thức của interface Shape, bao gồm resize().
* Sử dụng Logger thay vì System.out.println() để tuân thủ logging practice.
* Tạo lớp IncompleteInterfaceImplementationExample để sử dụng và kiểm thử Square.

Rewrite code:

A screenshot of a computer program

AI-generated content may be incorrect.

package quangnc.example;  
  
import java.util.logging.Logger;  
  
interface Shape {  
 void draw();  
 void resize();  
}  
  
class Square implements Shape {  
 private static final Logger *LOGGER* = Logger.*getLogger*(Square.class.getName());  
  
 @Override  
 public void draw() {  
 *LOGGER*.info("Drawing square");  
 }  
  
 @Override  
 public void resize() {  
 *LOGGER*.info("Resizing square");  
 }  
}  
  
public class IncompleteInterfaceImplementationExample {  
 public static void main(String[] args) {  
 Shape square = new Square();  
 square.draw();  
 square.resize();  
 }  
}

Result:

A screenshot of a computer program

AI-generated content may be incorrect.

15.InterfaceNamingInconsistencyExample.java:

Original code:

A screenshot of a computer

AI-generated content may be incorrect.

package quangnc.example;  
  
interface loginhandler {  
 login(string username, string password);  
}

Error:

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Error Type | Level | Description |
| 3 interface loginhandler | Code Smell | Medium | |  | | --- | | Tên interface loginhandler không tuân theo quy tắc PascalCase. Định danh nên là LoginHandler. |  |  | | --- | |  | |
| 4 login(string username,...) | Compile Error | High | Sử dụng sai kiểu dữ liệu string. Trong Java phải là String (viết hoa). |
| 4 login(string username,...) | Compile Error | High | Thiếu kiểu trả về cho phương thức login(...). Interface không thể có phương thức không định nghĩa kiểu trả về. |

A screenshot of a computer program

AI-generated content may be incorrect.

**Fix Suggestion:**

* Đổi tên interface từ loginhandler → LoginHandler (đúng chuẩn Java naming convention).
* Sửa string thành String.
* Thêm kiểu trả về (ví dụ: boolean) cho phương thức login.

Rewrite code:

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

Result:

A screenshot of a computer

AI-generated content may be incorrect.