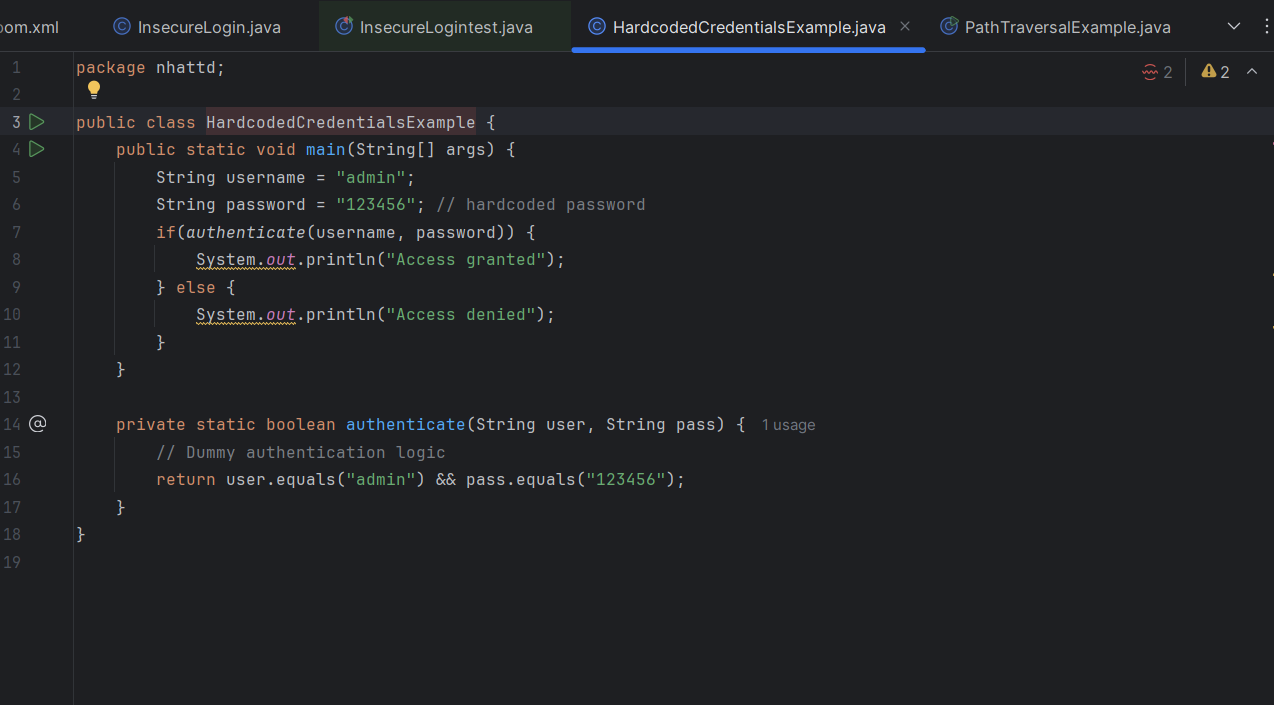
SWT301 Trương Đình Nhật DE180221

Lab 3 final

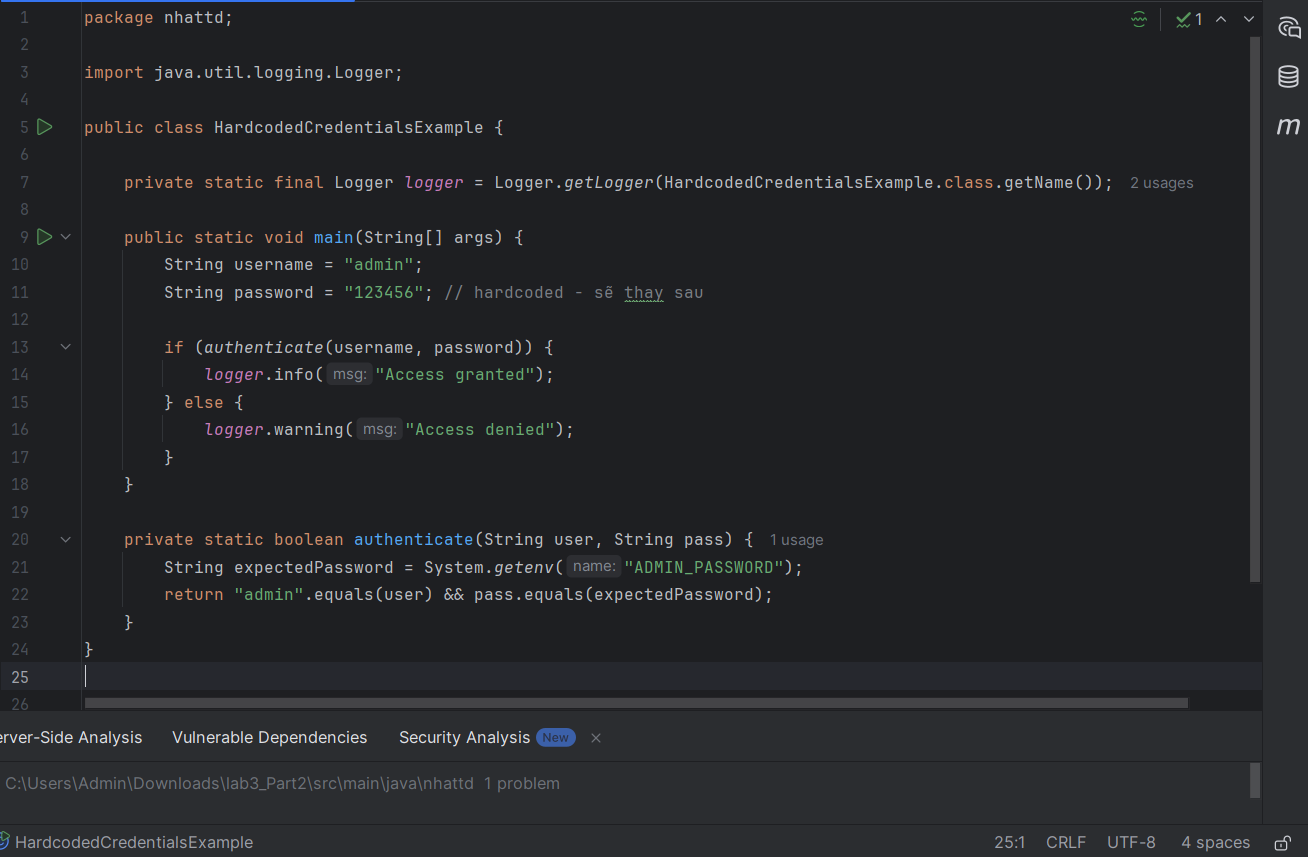
Sửa các lỗi code từng class

**Class:HardcodedCredentialsExample**

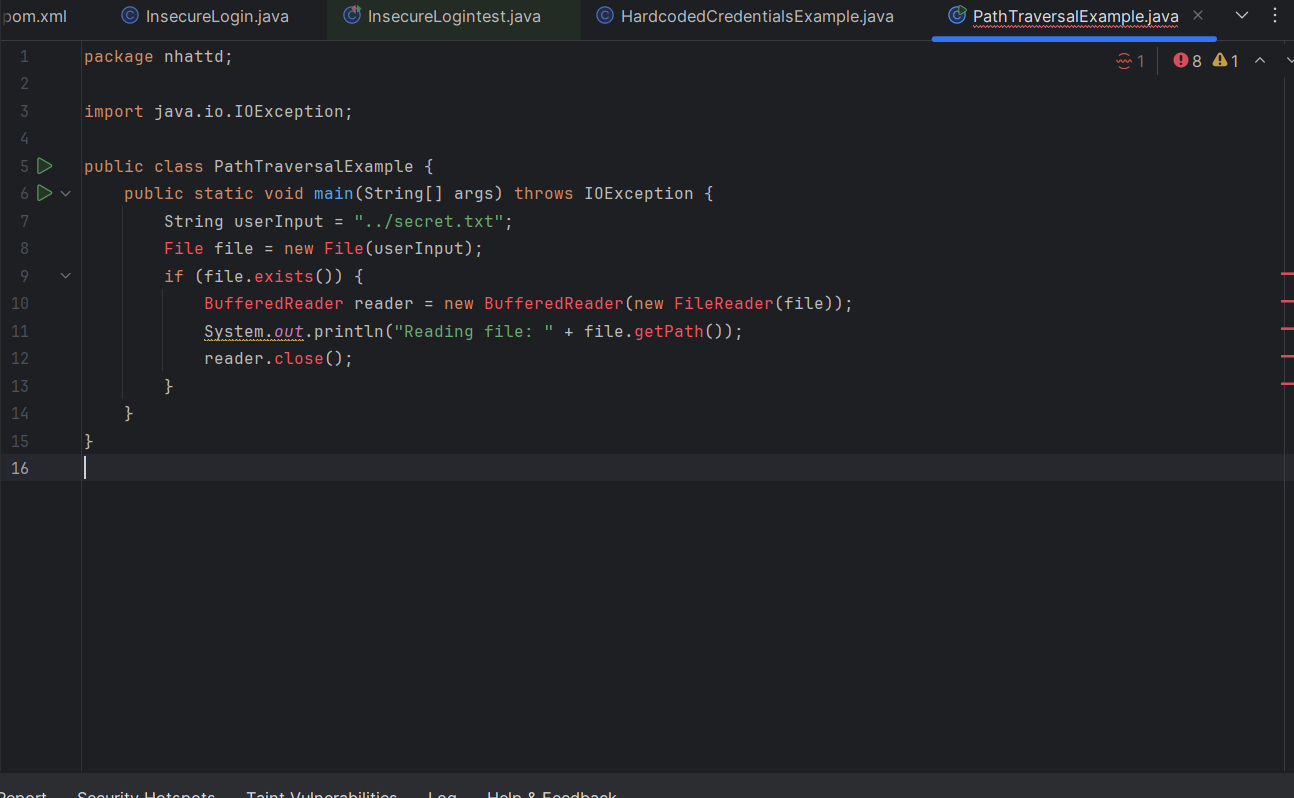


| **Line** | **Error Type** | **Description** |
| --- | --- | --- |
| 8 | Code Smell | Should not use System.out.println() → use logger instead |
| 10 | Code Smell | Should not use System.out.println() → use logger instead |
| 6 + 17 | Security | Hardcoded credentials → password "123456" is hardcoded |

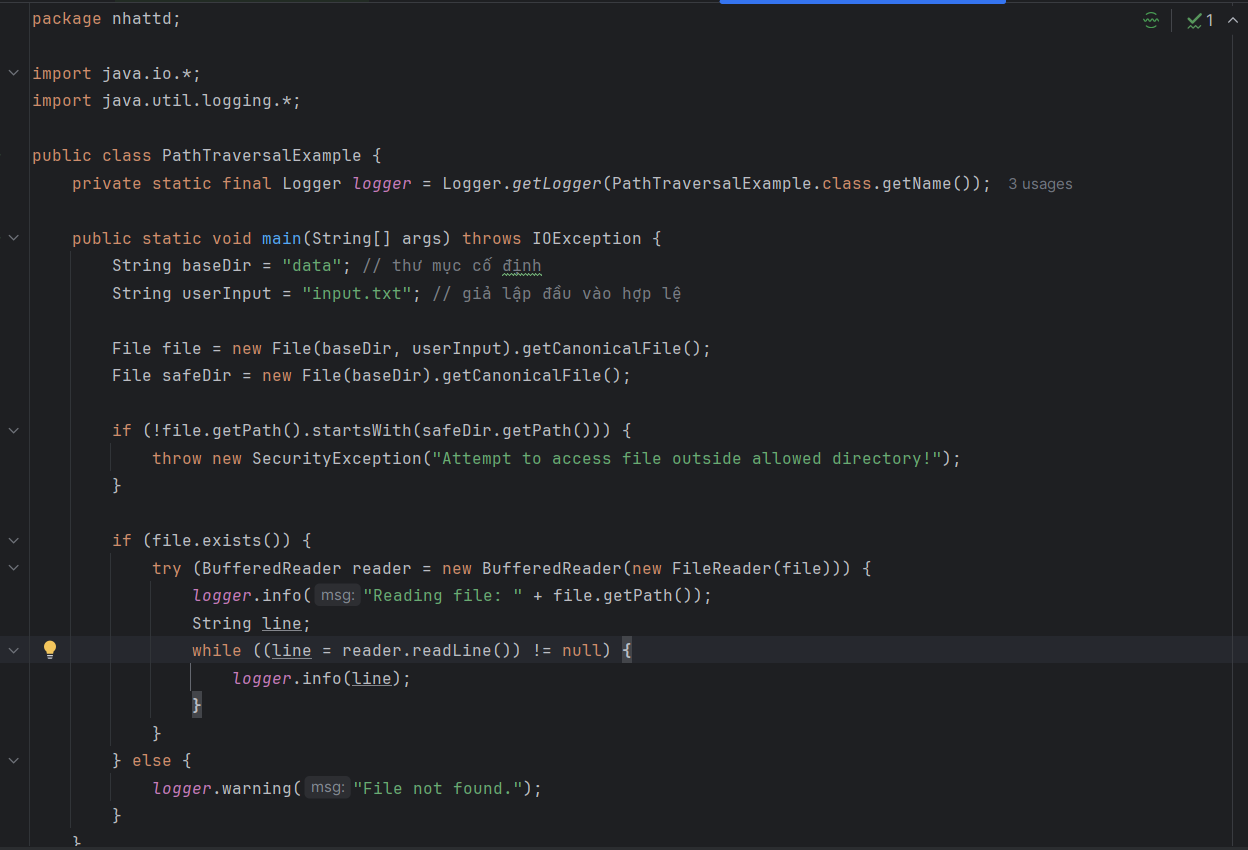
Code sạch hơn:



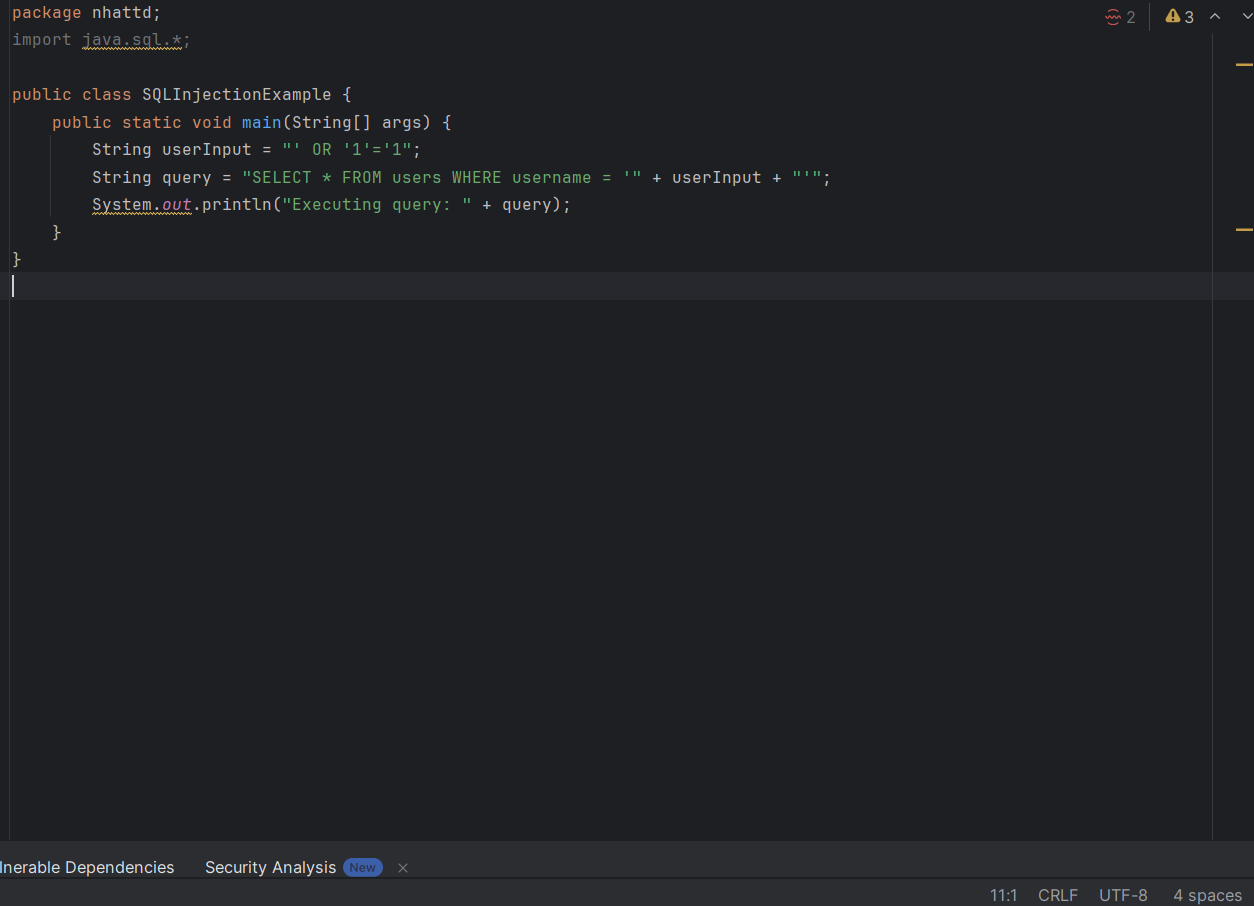
**Class: PathTraversalExample**



| **Line** | **Error Type** | **Error Description** |
| --- | --- | --- |
| 7 | Security | Using a user-supplied path without validation → Path Traversal vulnerability |
| 10 | Security | No authentication or access control on user-supplied file path |
| 11 | Code Smell | Using System.out.println() instead of a proper Logger |
| 12 | Resource Leak | BufferedReader not properly closed if an exception occurs |

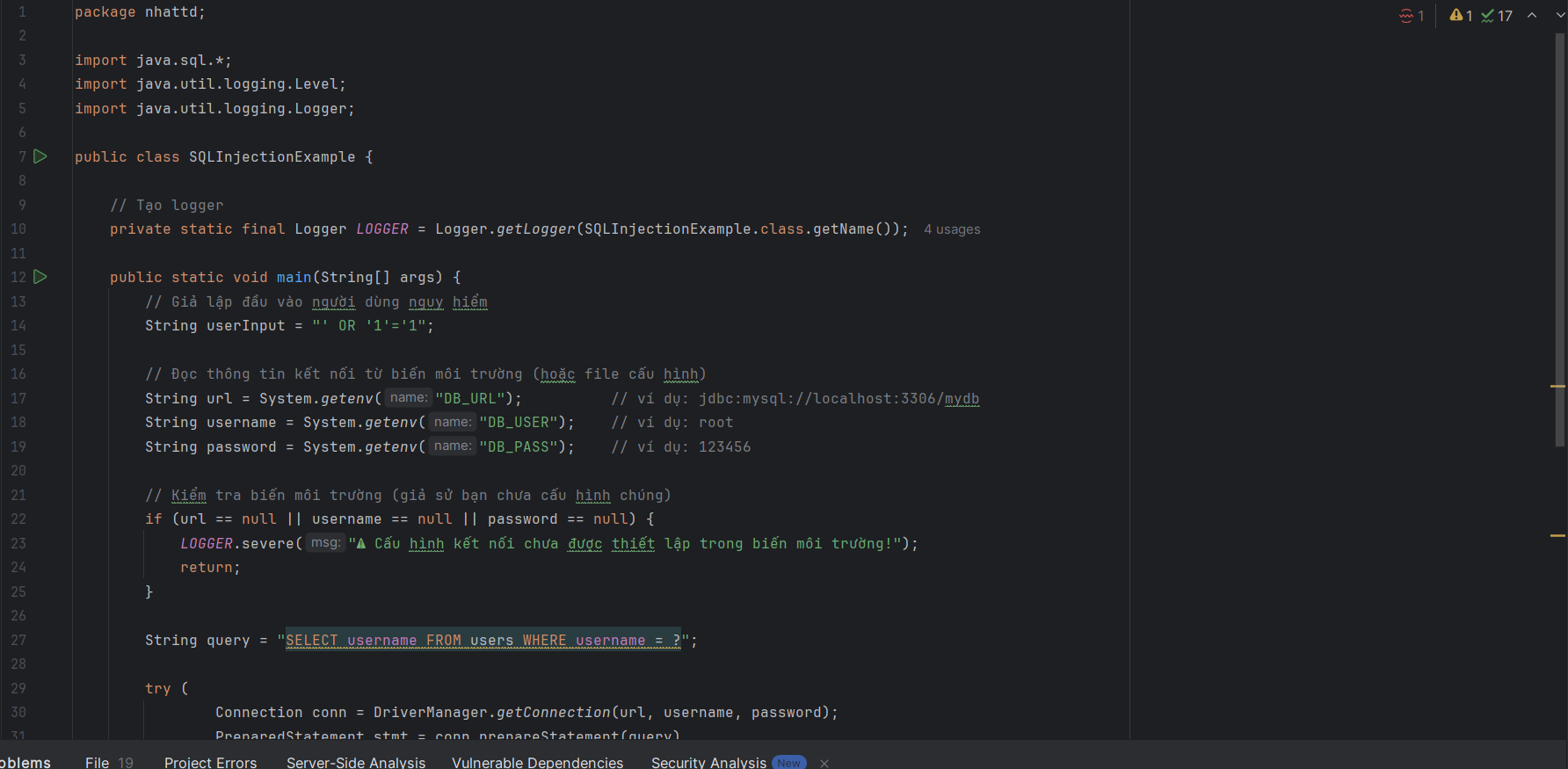


**Class: SQLInjectionExample**

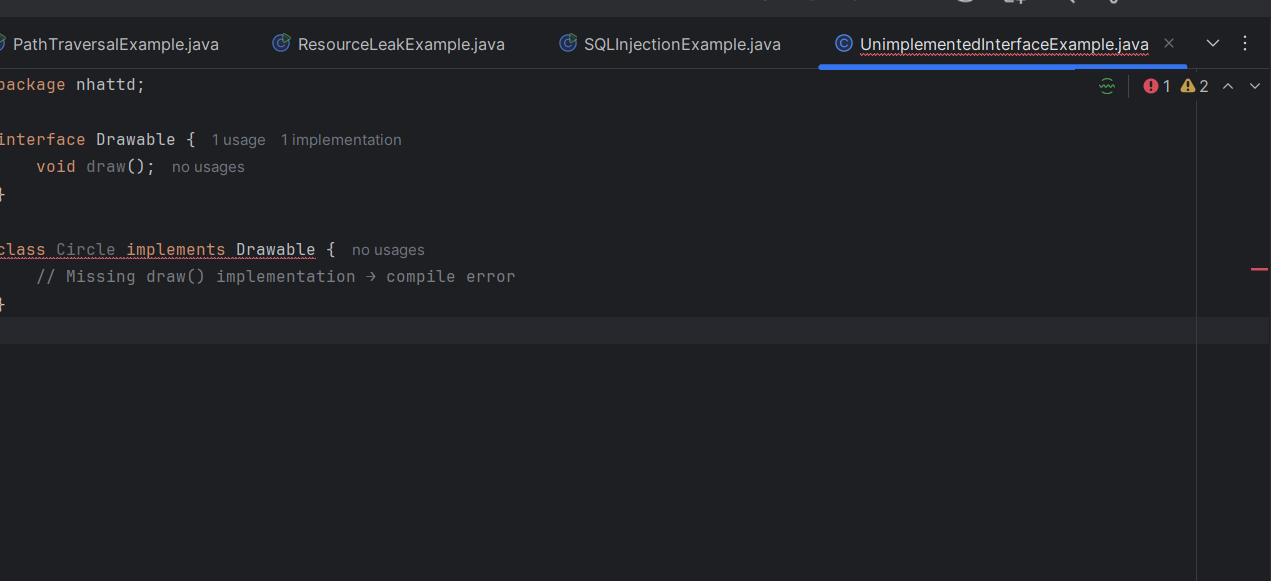


| **No.** | **Issue Type** | **Description** | **Example in Code** | **Consequence** | **Fix Recommendation** |
| --- | --- | --- | --- | --- | --- |
| 1 | **SQL Injection** | Concatenating SQL statements with user input leads to potential SQL injection vulnerability | "SELECT \* FROM users WHERE username = '" + userInput + "'" | Attackers can bypass login or extract data using input like ' OR '1'='1 | Use PreparedStatement with parameterized queries |
| 2 | **Unsafe Logging** | Using System.out.println() instead of a proper logging framework | System.out.println("Executing query: " + query); | Cannot control log levels, may expose sensitive data to the console | Use Logger (e.g., java.util.logging.Logger or SLF4J) |
| 3 | **Missing Exception Handling** | No try-catch block to handle SQL or connection errors | No try { ... } catch (SQLException e) { ... } | Program may crash on failure, no graceful error handling | Add try-catch and log exceptions properly |
| 4 | **Sensitive Data Exposure** | Logging raw SQL queries may leak sensitive data | System.out.println("Executing query: " + query); | Attackers can see logs and reverse-engineer database structure or input | Mask or avoid logging sensitive input |
| 5 | **Lack of Secure Structure** | Code is a simple mock, lacks real DB connection and resource management | No DB connection, no resource cleanup (Connection, Statement, ...) | Not secure or practical in production; missing closing of DB resources | Use real JDBC connection with try-with-resources and input validation |

Code sạch hơn:

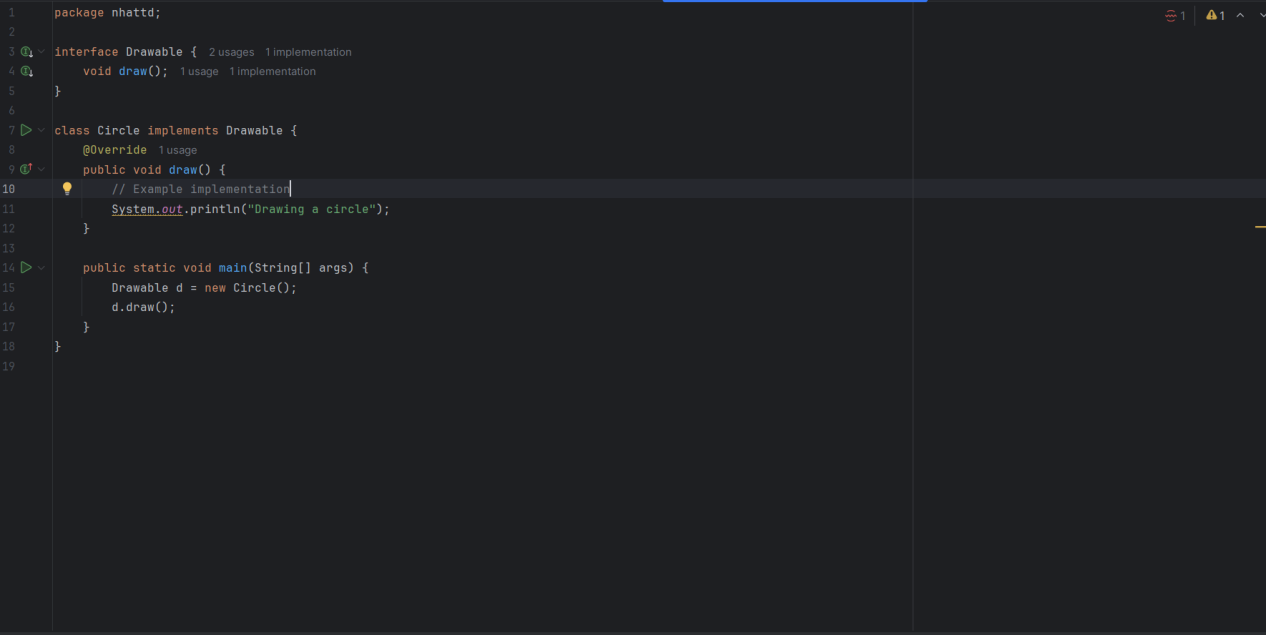


Class : UnimplementedInterfaceExample

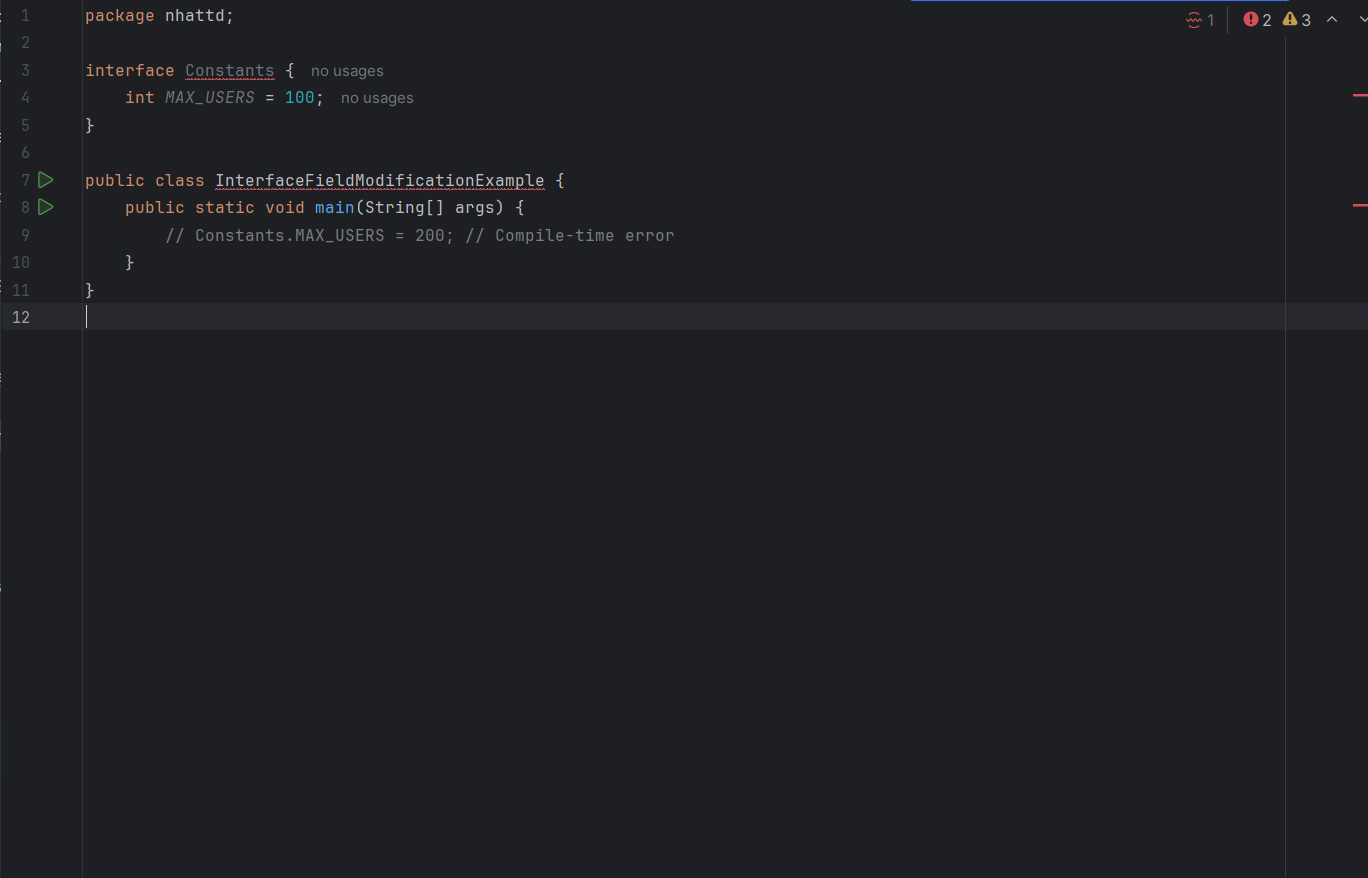


| **No.** | **Error Type** | **Description** | **Code Snippet** | **Consequence** | **Fix Recommendation** |
| --- | --- | --- | --- | --- | --- |
| 1 | **Unimplemented Interface Method** | The class implements an interface but does not provide implementation for all required methods | class Circle implements Drawable | Compile-time error: "Circle is not abstract and does not override abstract method draw()" | Implement the missing draw() method in the Circle class |

Code sạch hơn:

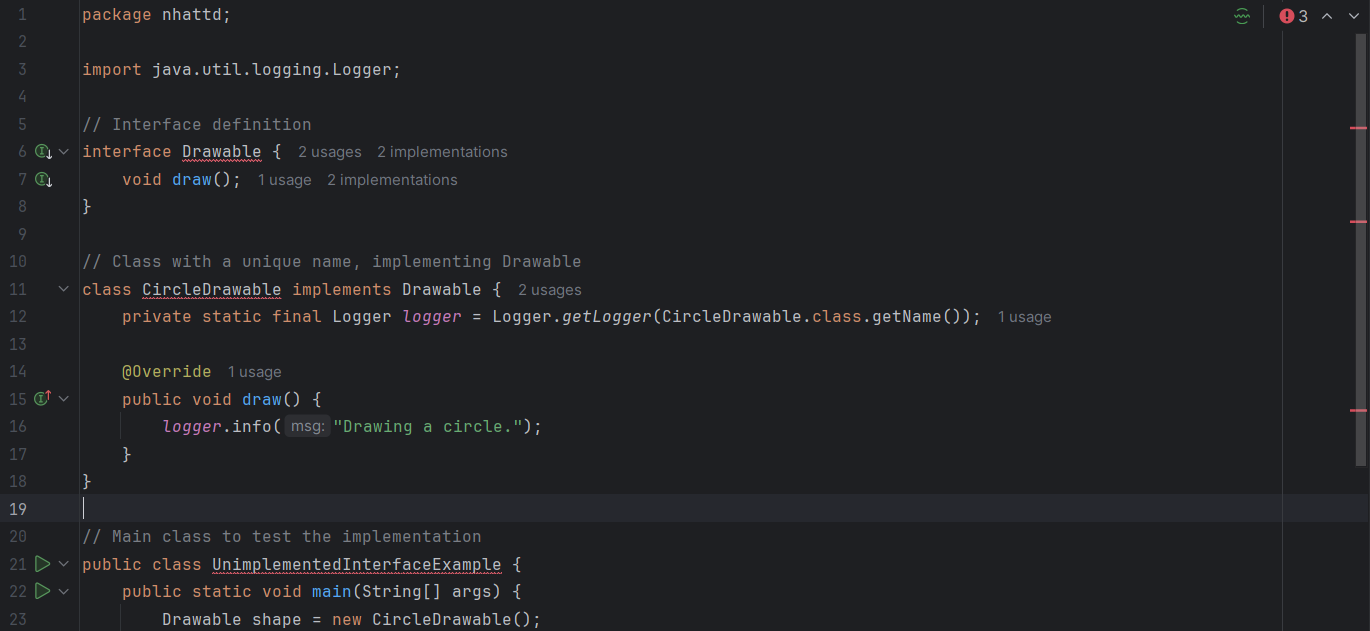


**Class InterfaceFieldModificationExample**

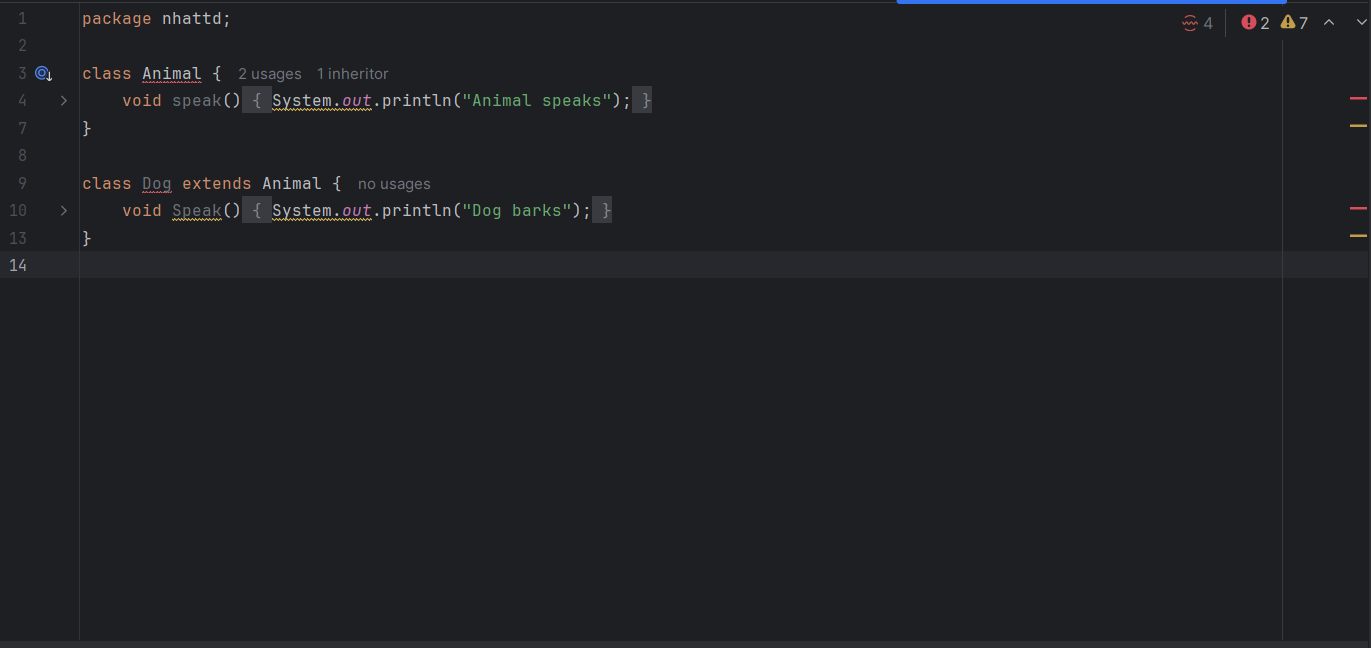


| **No.** | **Type of Warning** | **Description** | **Cause** | **Suggested Fix** |
| --- | --- | --- | --- | --- |
| 1 | Unused Method | Method draw() is defined but never called anywhere. | Method is implemented in Circle class but not invoked. | Add a main() method or test case to call draw(). |
| 2 | Unused Class | Class Circle is never used or instantiated in the code. | No object of the class is created or referenced. | Instantiate Circle class and use it in code (e.g., new Circle().draw()). |
| 3 | Empty Method Implementation | The draw() method has no implementation, and no comment explaining why it’s empty. | The method body is empty and lacks explanation. | Either implement the method, throw UnsupportedOperationException, or add a comment like // TODO. |

Code sạch hơn :

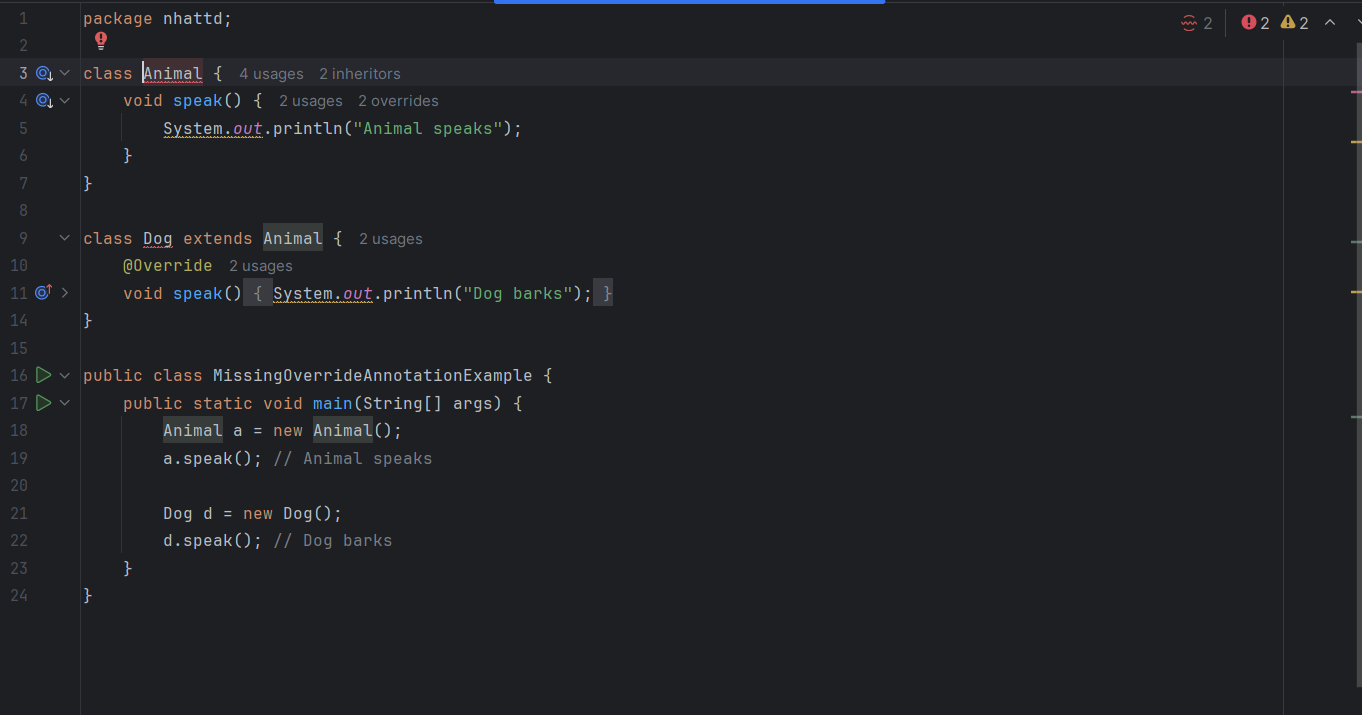


**Class: MissingOverrideAnnotationExample**

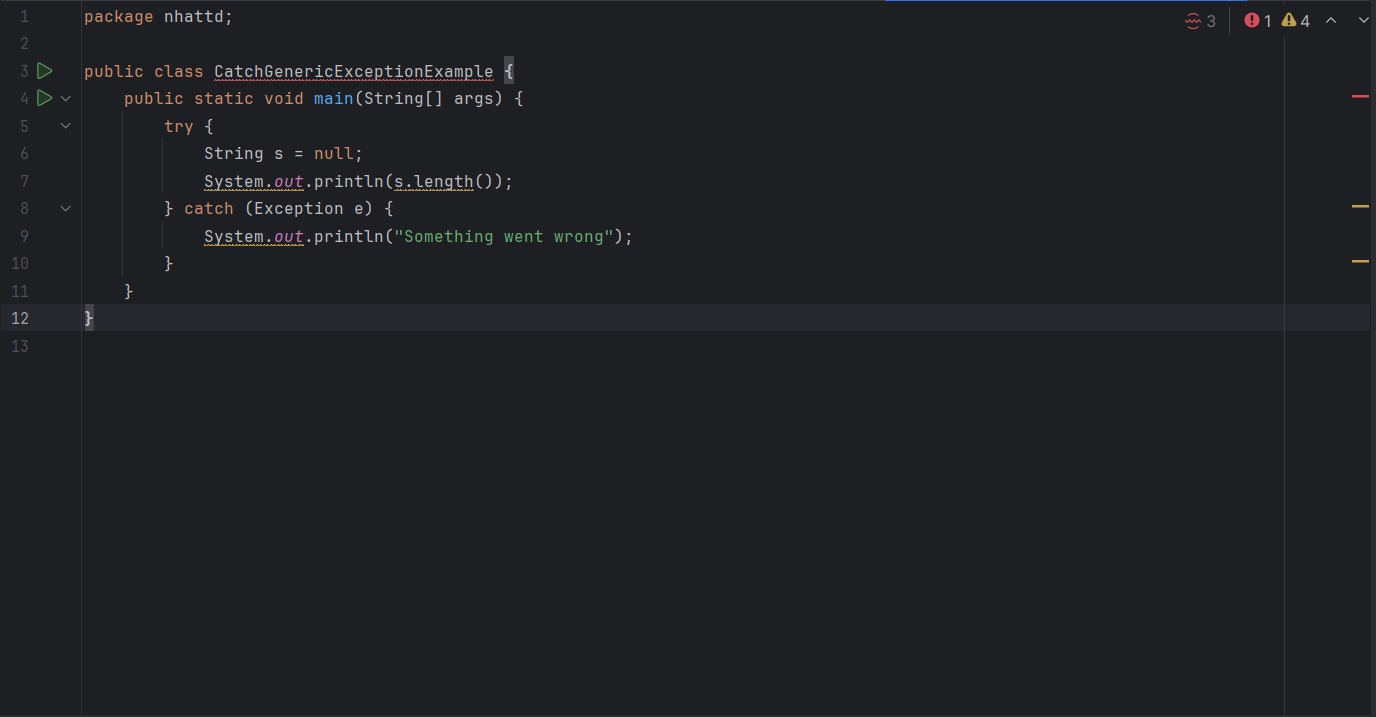


| **🔢 No.** | **⚠️ Error Type** | **🧠 Description** | **💡 Suggestion** |
| --- | --- | --- | --- |
| 1 | Duplicate Class | The class UnimplementedInterfaceExample appears in **multiple files**. | Ensure that each public class is declared in its **own file** named exactly like the class (e.g., UnimplementedInterfaceExample.java). Delete any duplicate file. |
| 2 | Public Class File Mismatch | The class UnimplementedInterfaceExample is public but not in a matching file. | Rename the file to UnimplementedInterfaceExample.java or remove the public modifier if not needed. |
| 3 | Method Never Used | The method draw() is defined but never used in the code. | Call the draw() method somewhere in your application or remove it if unnecessary. |
| 4 | Class Never Used | The class Circle is never instantiated or referenced anywhere. | Create an object or usage reference to the Circle class. |
| 5 | Empty Method Warning | An empty method body without comment or exception. | Either add a comment explaining it’s intentionally empty, throw UnsupportedOperationException, or implement the method. |
| 6 | System.out Usage | Use of System.out.println is discouraged in larger applications. | Use a logger (e.g., java.util.logging.Logger) instead of System.out.println for better practice. |
| 7 | Incorrect Method Override | In Dog, Speak() should be speak() to override the parent method. | Rename method to match the parent: speak(), and add @Override annotation. |
| 8 | Class Never Used | Class Dog is never used or instantiated. | Create an object of Dog and call its method in a main() method. |

Code sạch hơn:

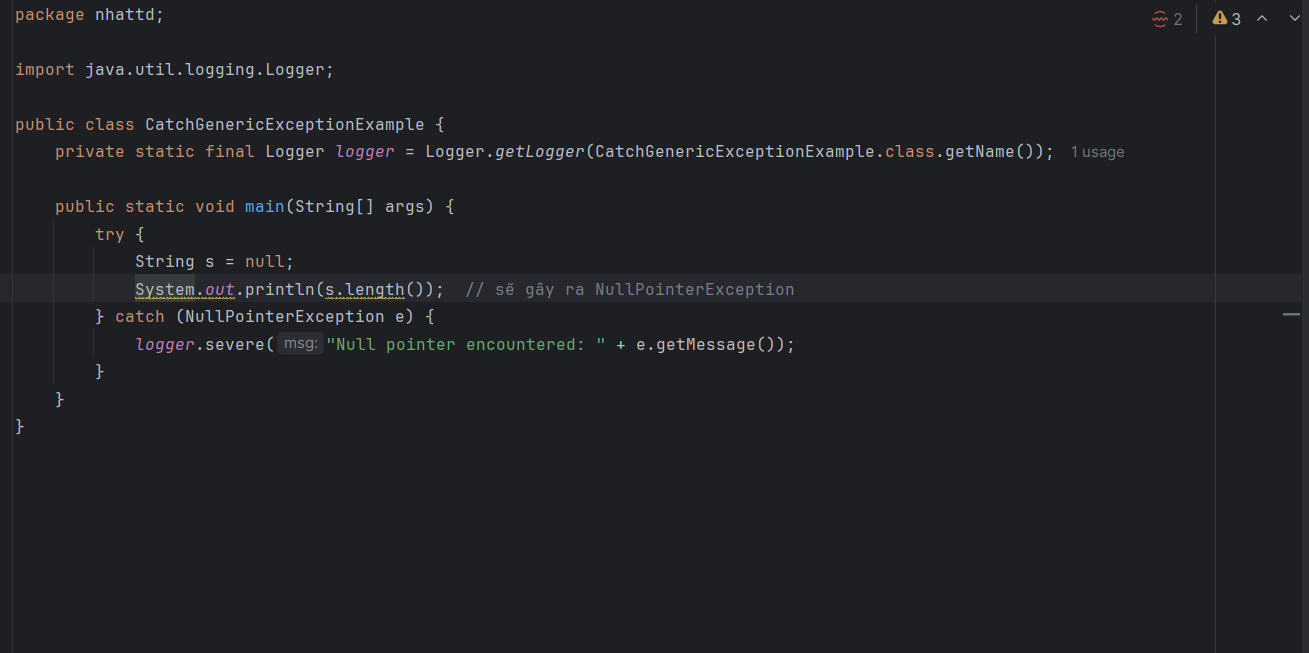


**Class: CatchGenericExceptionExample**

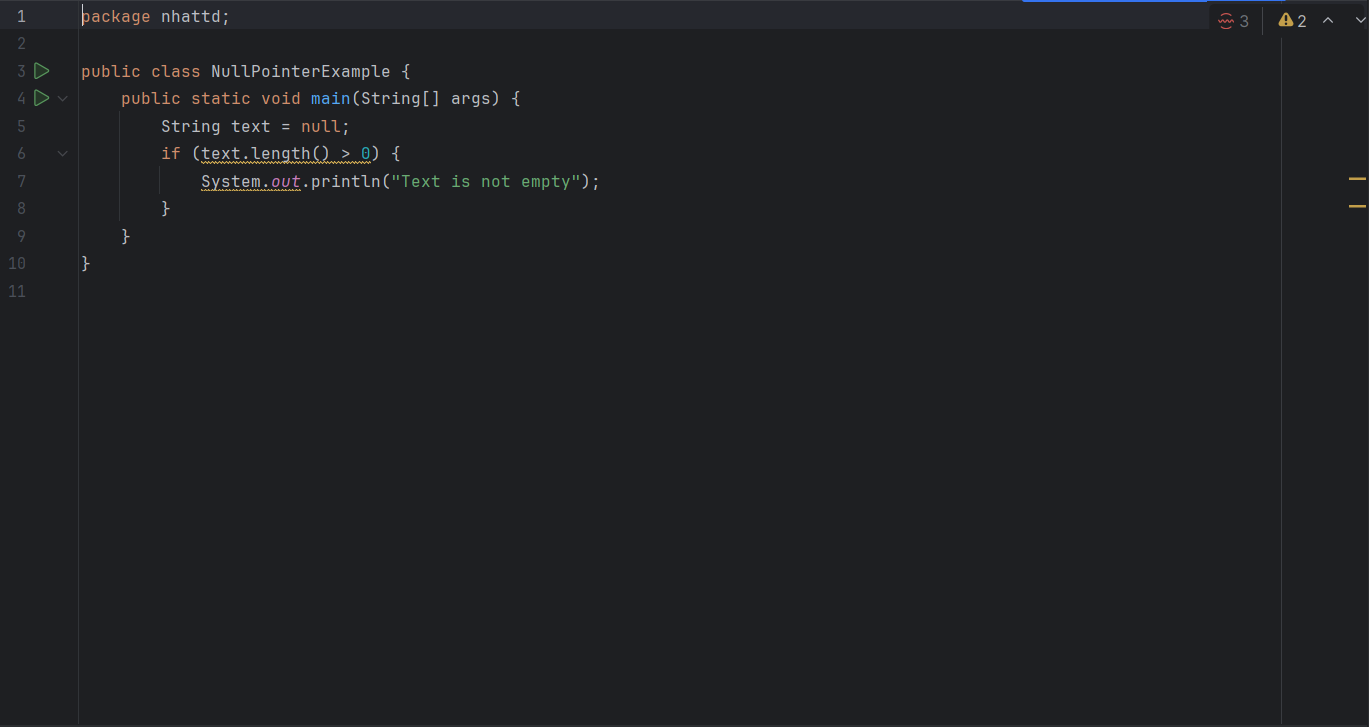


| **No.** | **Original Error** | **Corrected To** |
| --- | --- | --- |
| 1 | catch (Exception e) | catch (NullPointerException e) |
| 2 | System.out.println(...) | logger.severe(...) using java.util.logging |

Code sạch hơn :

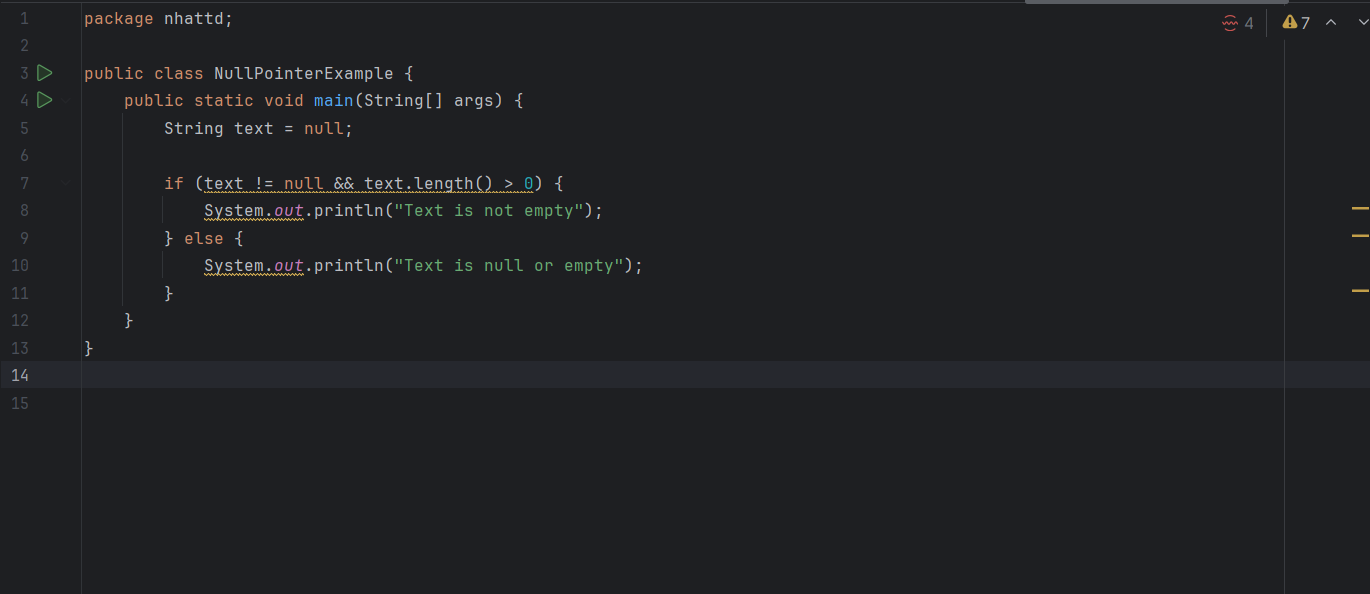


**Class: NullPointerExample**

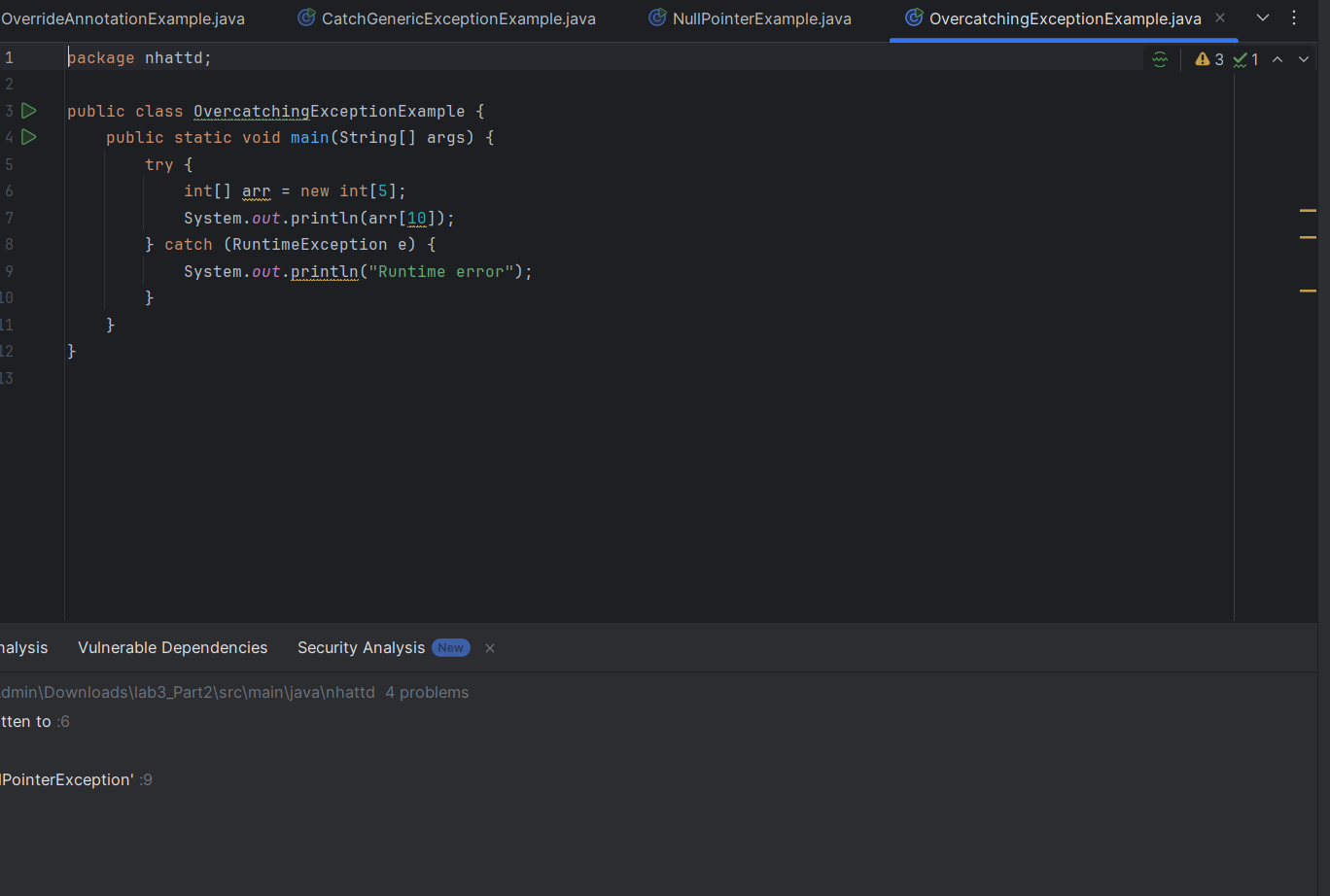


| **Line** | **Type** | **Issue** | **Description** |
| --- | --- | --- | --- |
| 5 | Error ⚠️ | NullPointerException | Calling text.length() when text is null causes a runtime exception. |
| 5 | Warning ⚠️ | Missing null check before calling .length() | Should check if text is not null before accessing its length. |
| 7 | Warning ⚠️ | Use of System.out.println in production code | Recommended to use a logger instead of System.out.println for better control. |

Code sạch hơn



**Class: OvercatchingExceptionExample**

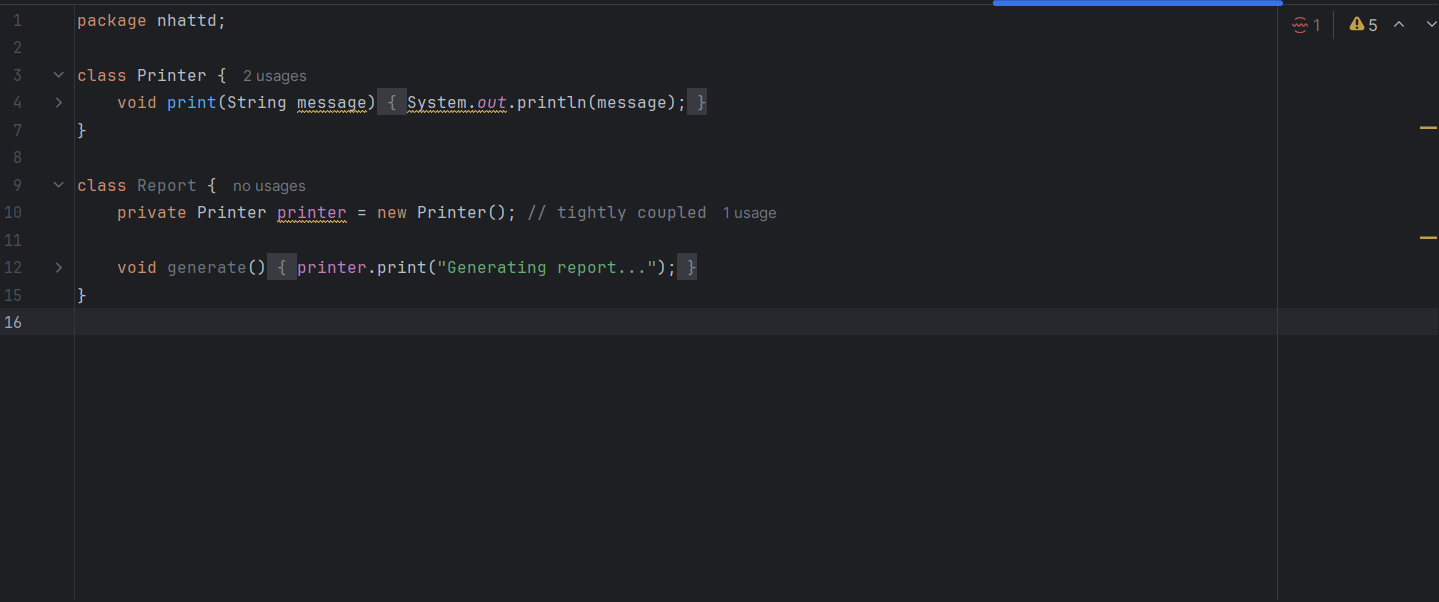


| **Line** | **Type** | **Issue** | **Description (English)** |
| --- | --- | --- | --- |
| 6 | Error ⚠️ | ArrayIndexOutOfBoundsException | Trying to access arr[10] when array only has 5 elements (indexes 0–4). |
| 8 | Warning ⚠️ | Over-catching RuntimeException | Catching too broad an exception type; may hide bugs or make debugging harder. Consider catching only what you need. |
| 9 | Warning ⚠️ | System.out.println used | Use a logger instead of System.out.println in production environments. |

Code sạch hơn:

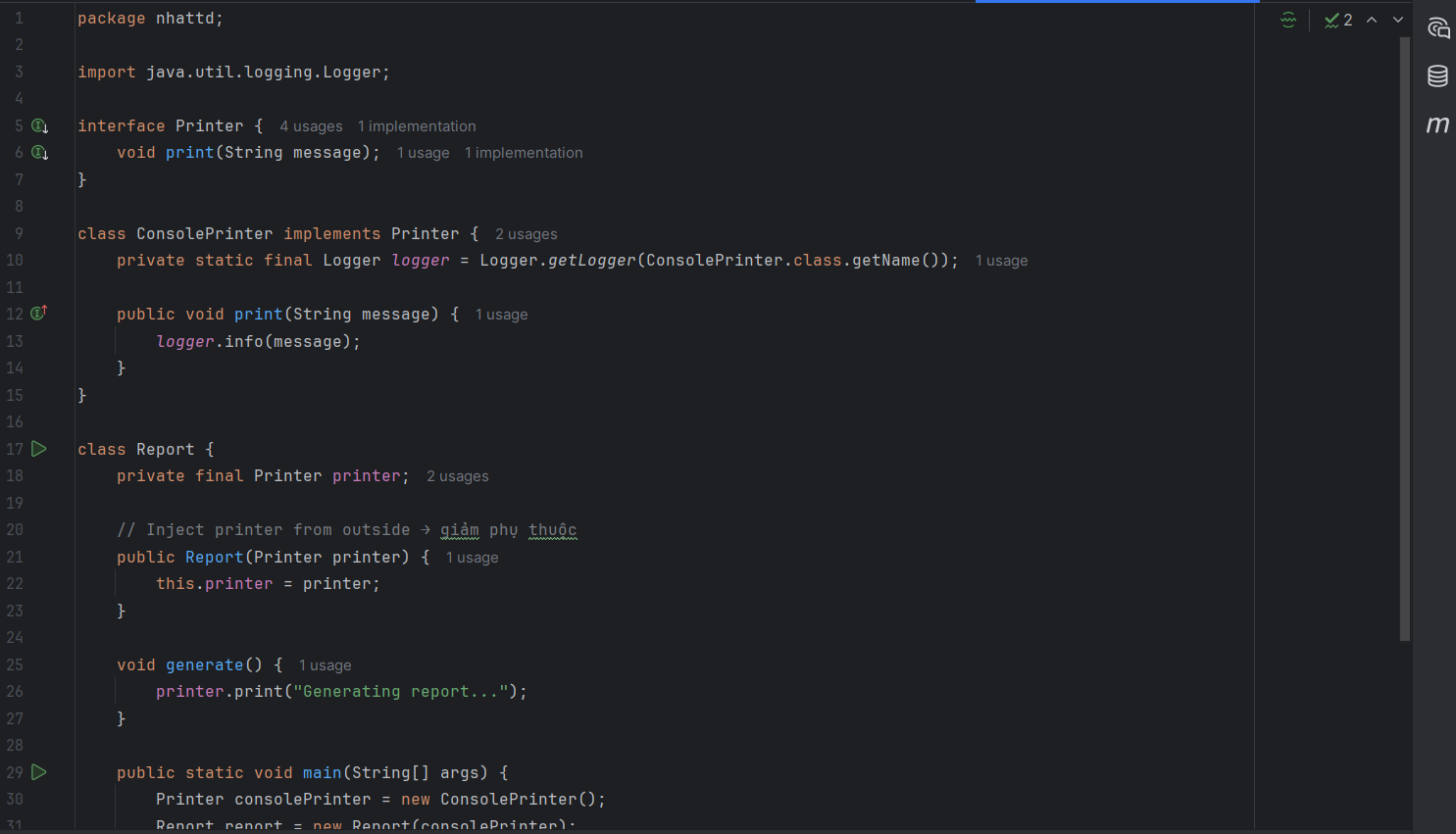


**Class: TightCouplingExample**

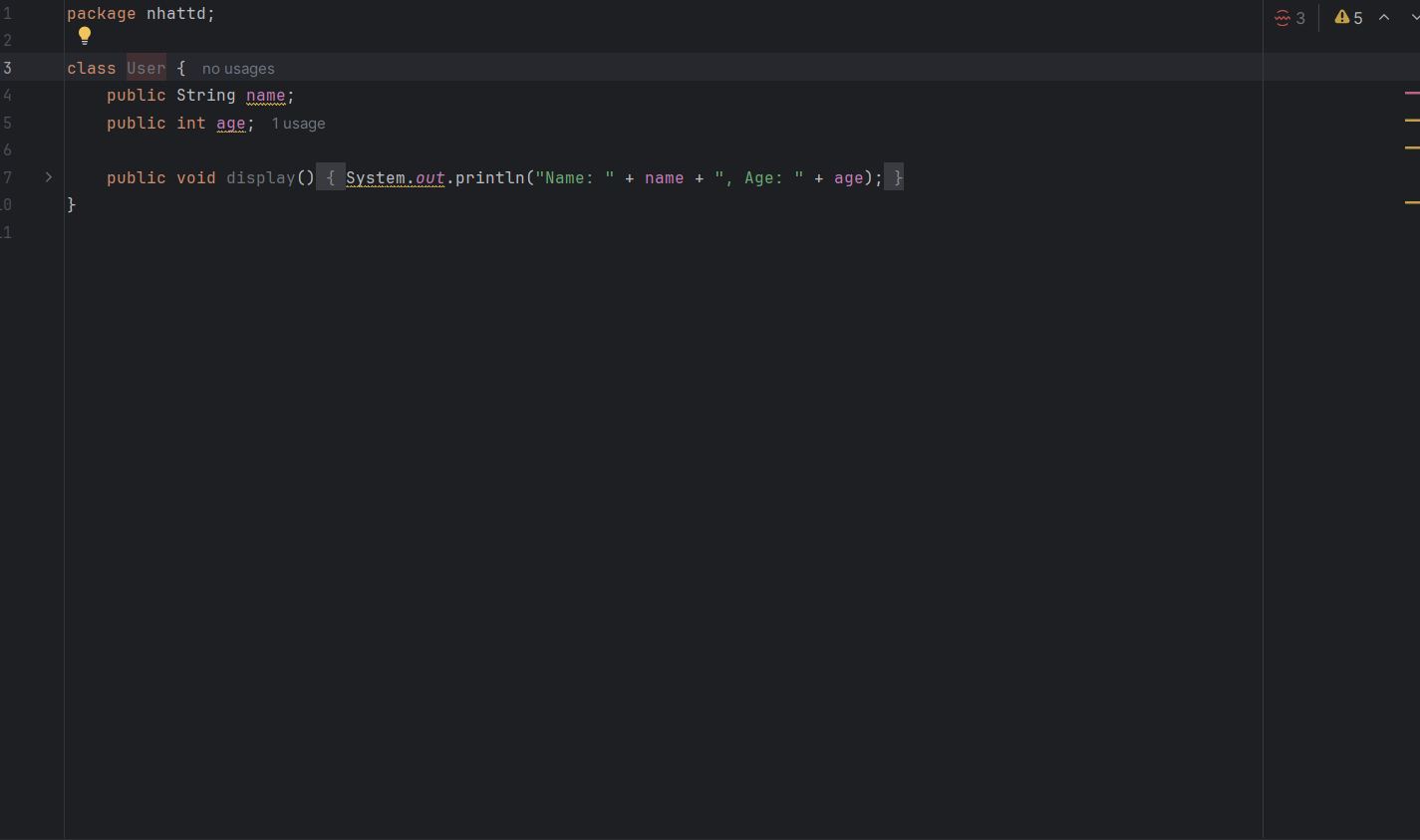


| **Line** | **Type** | **Description** | **Recommendation** |
| --- | --- | --- | --- |
| 6 | Code Smell | Direct use of System.out.println(...) for printing output. | Use a logging framework (e.g., Logger) for better logging control. |
| 10 | Tight Coupling | The Report class directly instantiates the Printer class. | Apply **Dependency Injection** to decouple the classes. |
| 10 | Maintainability | Hardcoded dependency makes the code harder to test and extend. | Pass the dependency via constructor (interface-based preferred). |
| — | Design Violation | Violates **Dependency Inversion Principle (DIP)** from SOLID principles. | Rely on abstraction (Printer interface), not concrete classes. |

Code sạch hơn:



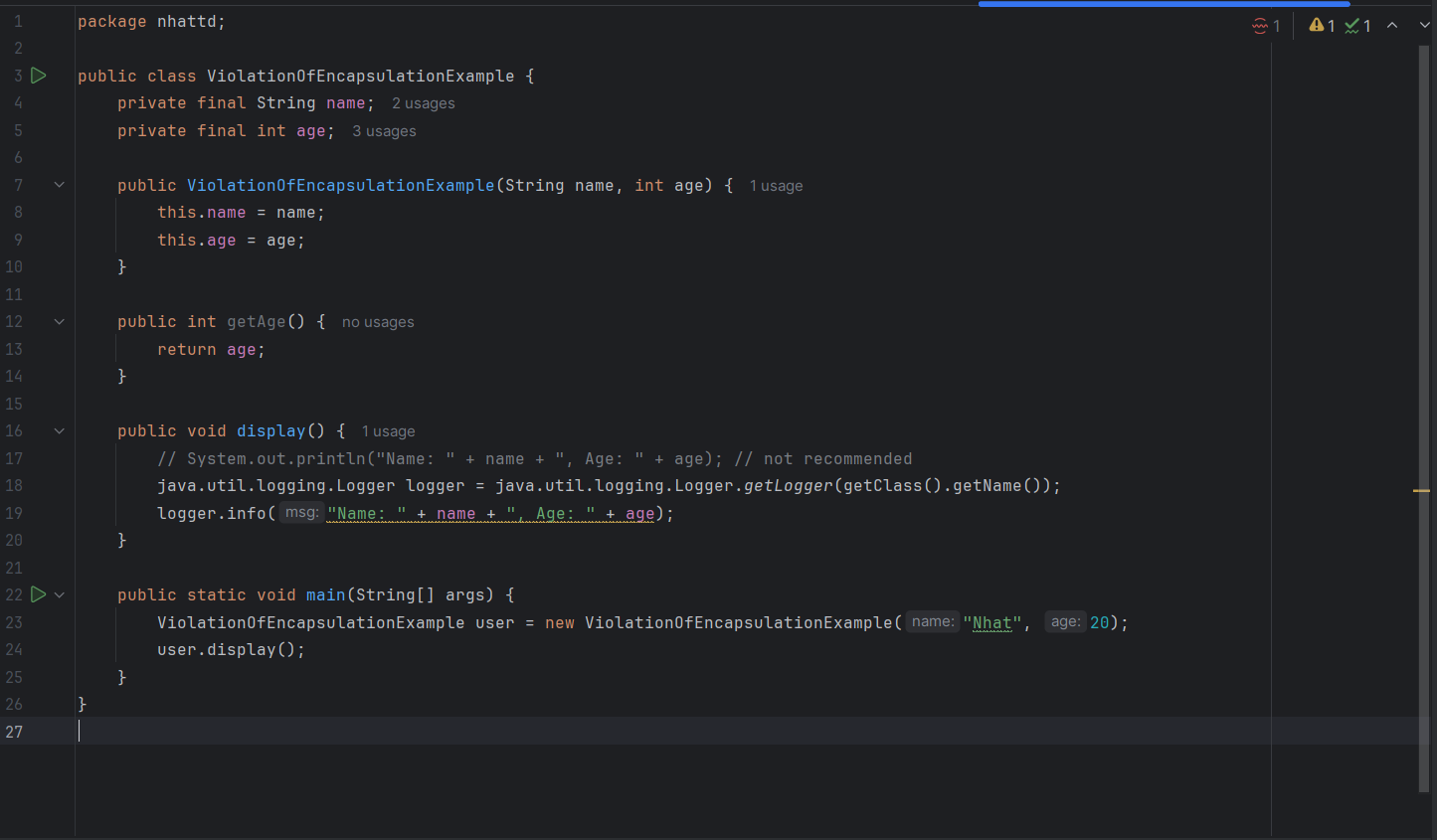
**Class : ViolationOfEncapsulationExample**



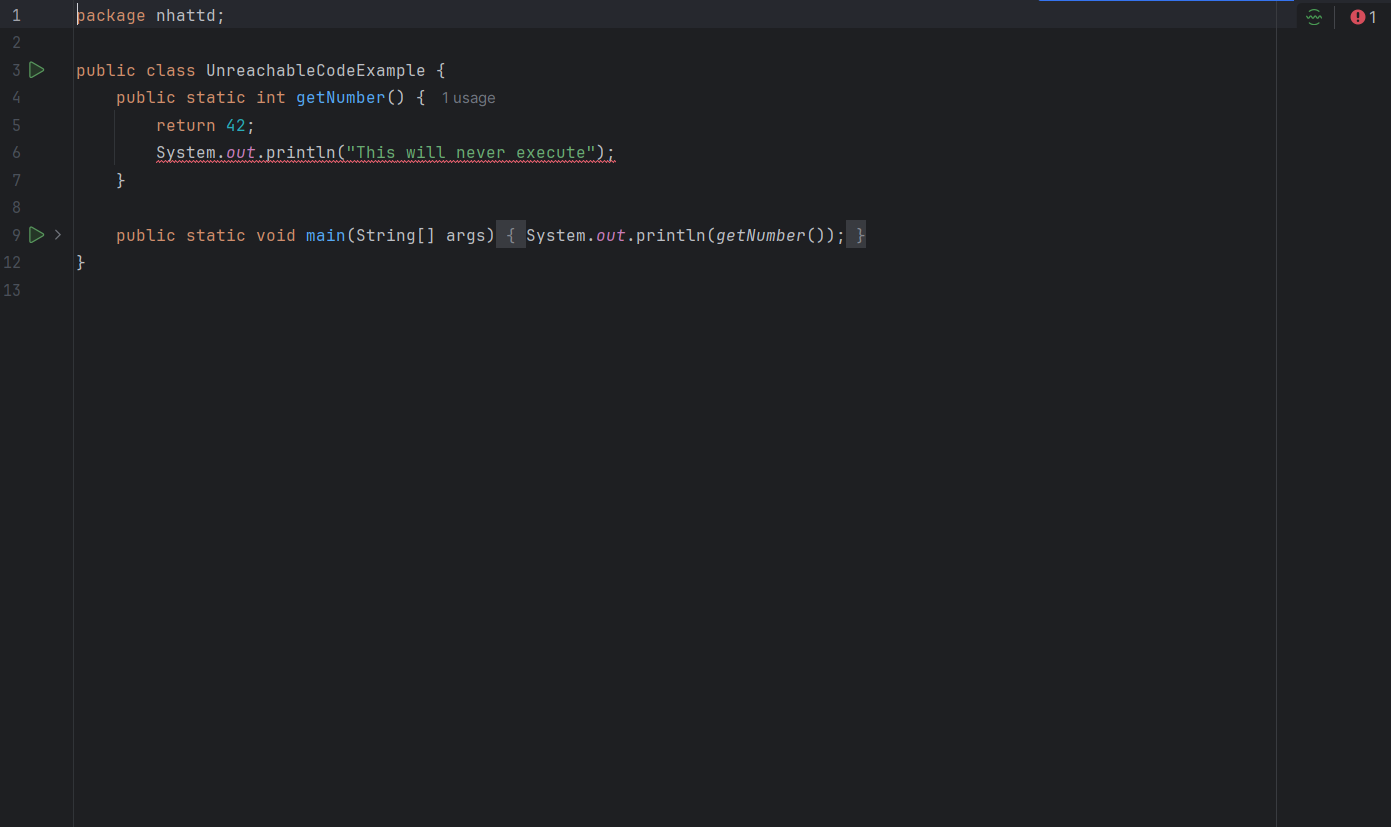
### ****Error Summary Table –**** ViolationOfEncapsulationExample.java

| **Line** | **Type** | **Description** | **Recommendation** |
| --- | --- | --- | --- |
| 4 | Poor Encapsulation | Public field name exposes internal state of the class. | Use private access modifier and provide getter/setter methods. |
| 5 | Poor Encapsulation | Public field age breaks encapsulation. | Same as above: make age private, and add getter/setter. |
| 7 | Code Smell | Direct use of System.out.println(...) inside business logic. | Use a logger (java.util.logging or other frameworks like Log4j, SLF4J). |
| 3 | Missing Constructor | Class User has no constructor; data must be set manually after object created. | Add a constructor to initialize name and age properly. |

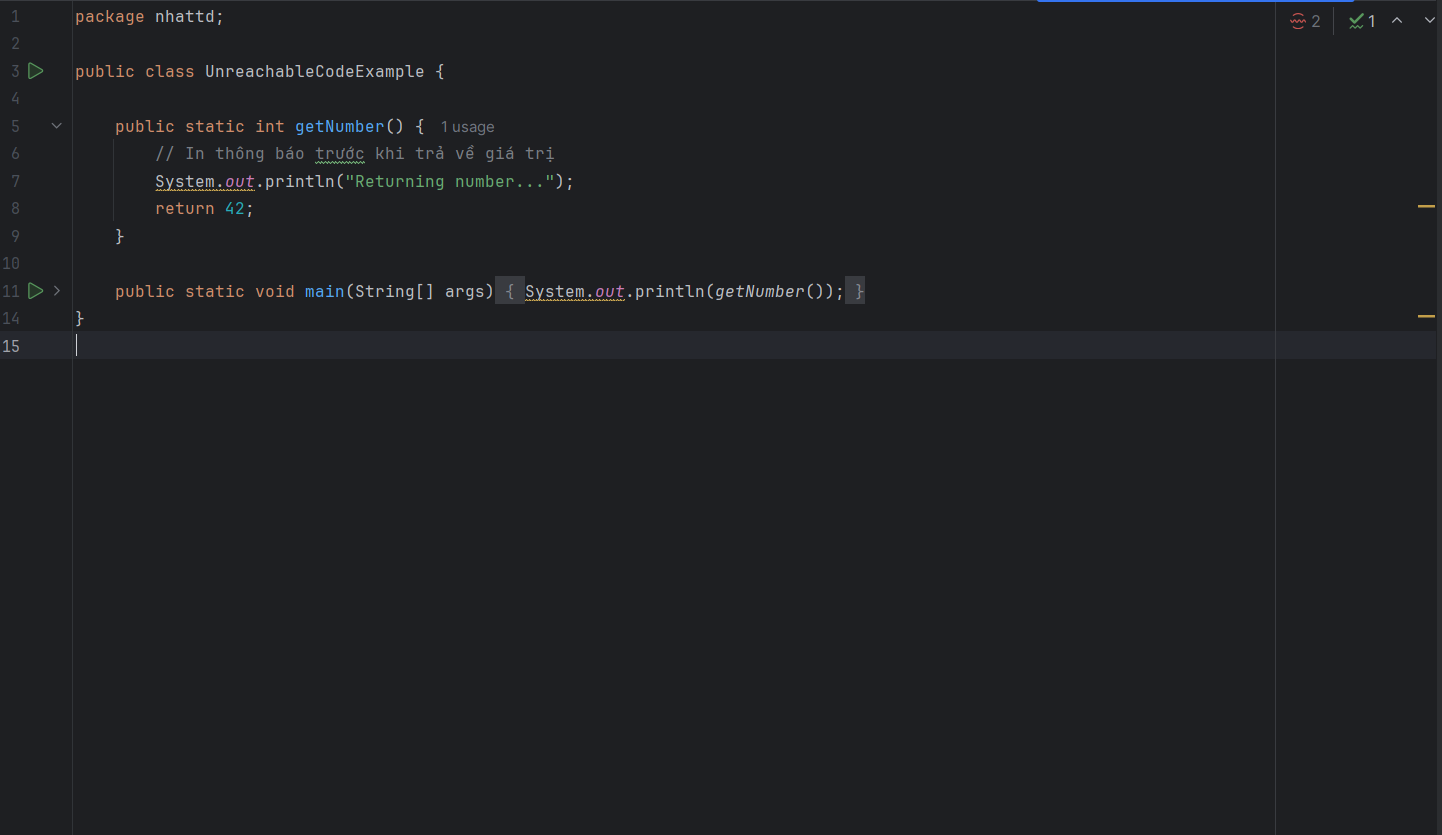
Code sạch hơn:



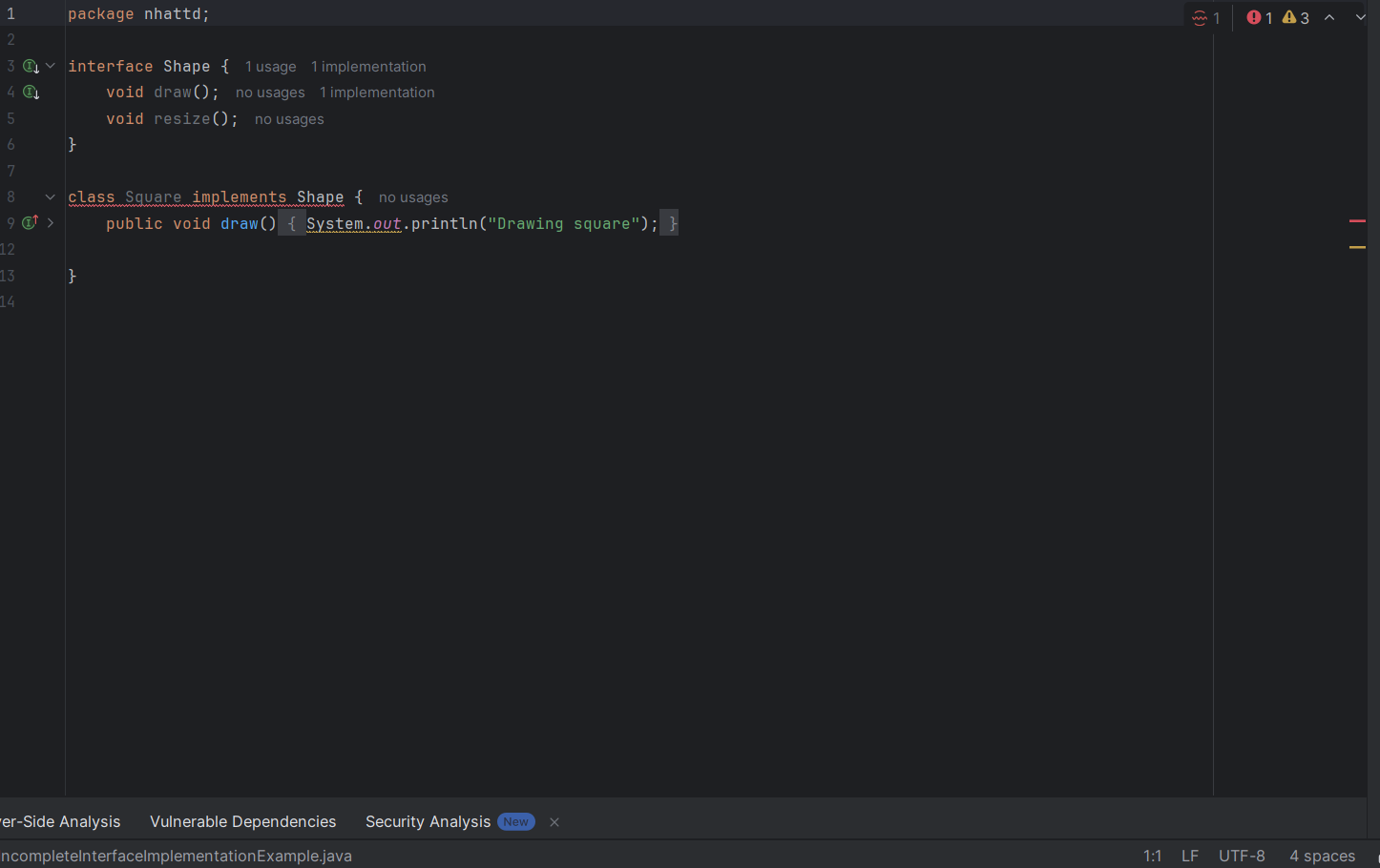
**Class: UnreachableCodeExample**



| **Line** | **Issue** | **Type** | **Explanation** | **Fix** |
| --- | --- | --- | --- | --- |
| 6 | Unreachable statement | Compilation Error | Code after return cannot be executed | Remove or move the System.out.println() line above the return |



**Class: IncompleteInterfaceImplementationExample**

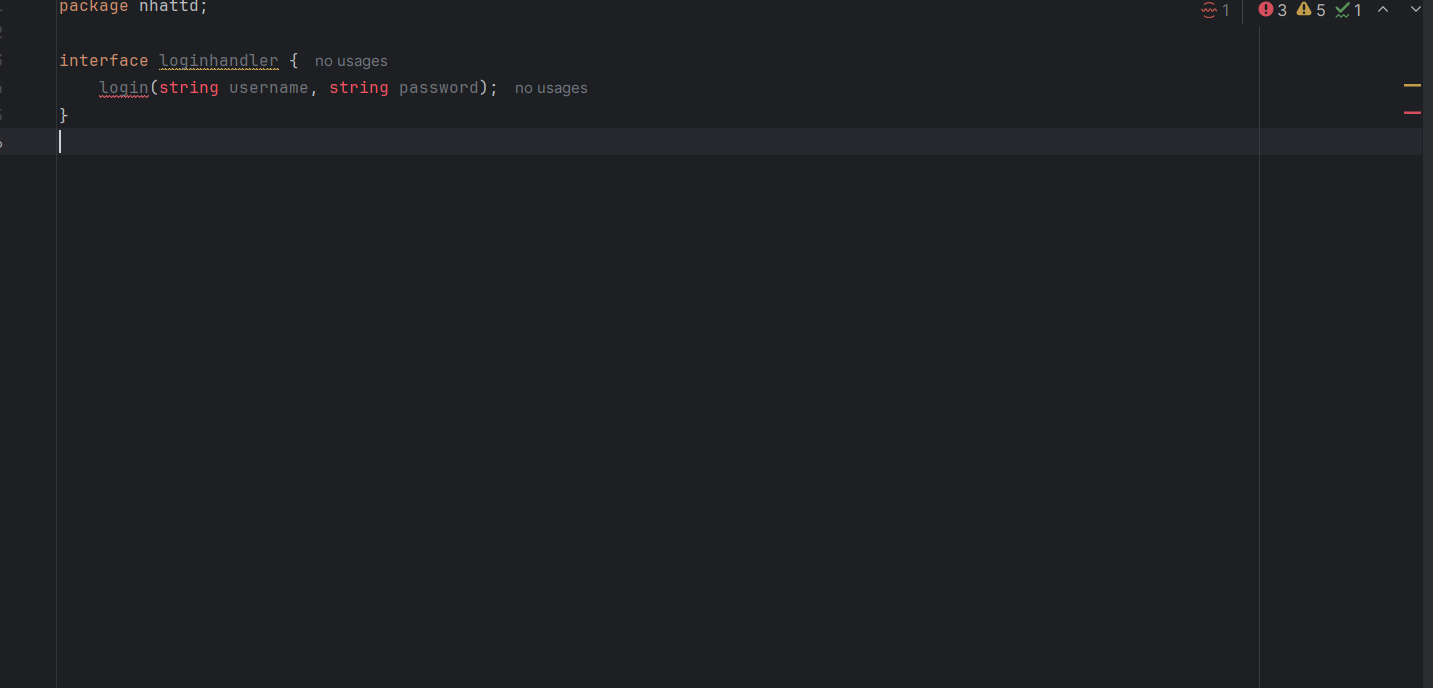


| **Line** | **Error Message** | **Cause** | **Solution** |
| --- | --- | --- | --- |
| 8 | Square is not abstract and does not override abstract method resize() in Shape | The Square class implements the Shape interface but **does not implement the** resize() **method**. | Add the resize() method implementation in the Square class. |
| 9 | draw() has no usages (warning) | The draw() method is defined but **not used anywhere** in the code. This is just a warning. | Use the draw() method somewhere or ignore the warning if it's not needed yet. |

Code sạch hơn:



**Class : InterfaceNamingInconsistencyExample**



| **No.** | **Error Line** | **Error Message** | **Cause** | **Solution** |
| --- | --- | --- | --- | --- |
| 1 | string username | Cannot resolve symbol 'string' | Java does not recognize lowercase string as a valid type | Change string to String |
| 2 | login(...) | Missing method return type | Interface methods must declare a return type in Java | Add a return type like boolean, void, etc. |
| 3 | interface loginhandler | (Naming convention warning) | Interface name does not follow Java naming conventions | Rename to LoginHandler (PascalCase recommended) |

Code sạch hơn :

