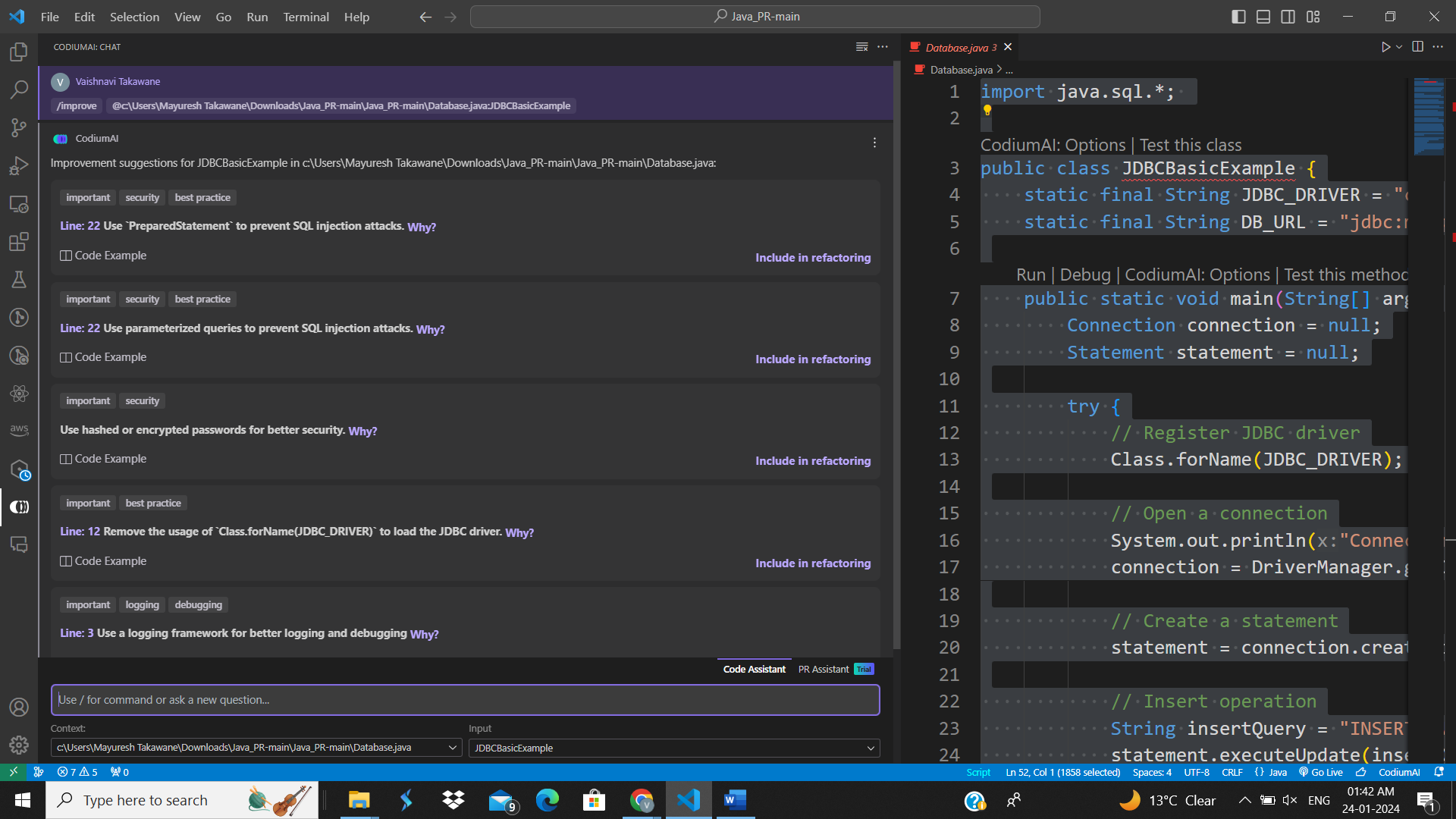
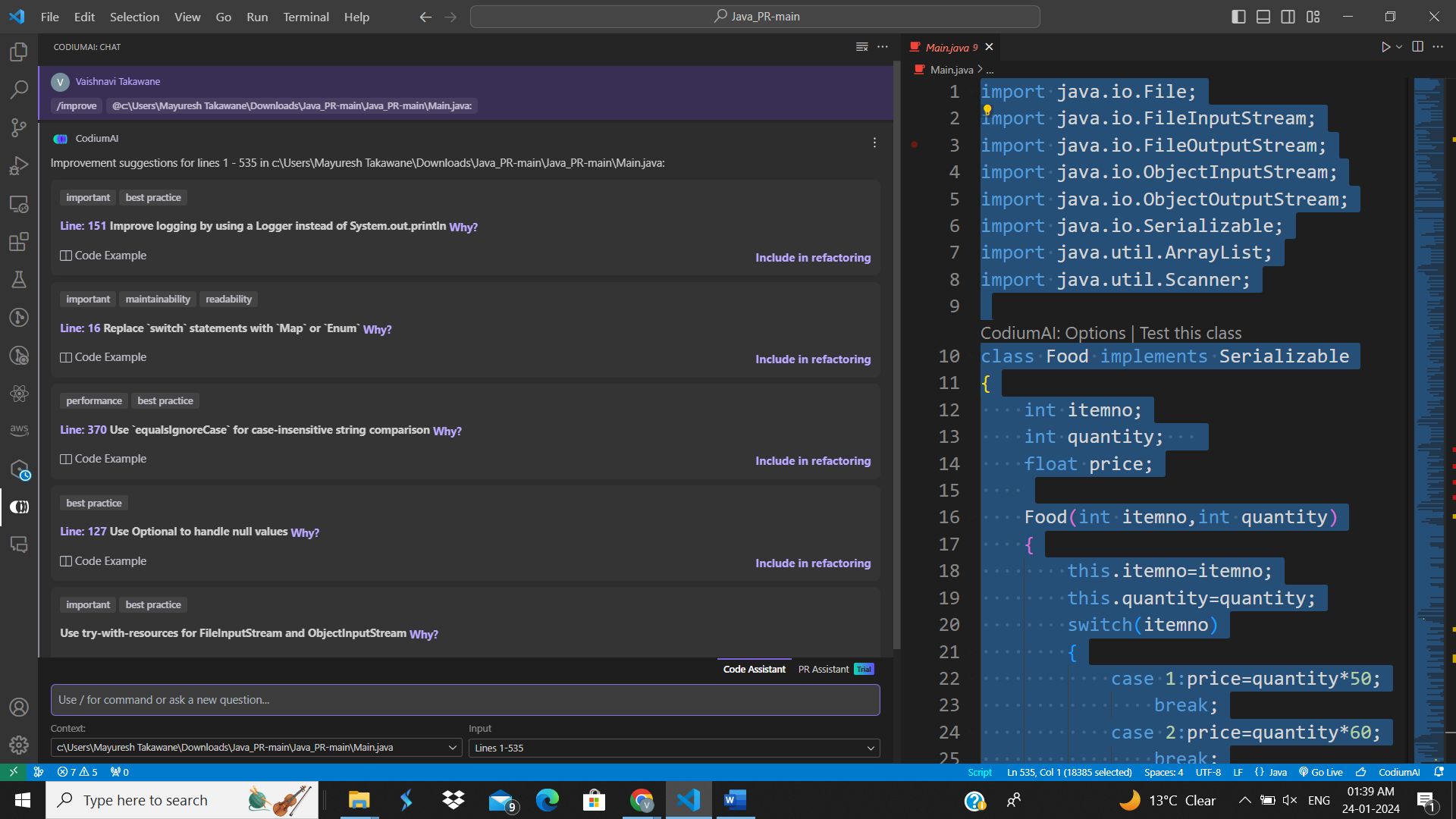
**Codium AI –**

**Code Analysis on Database.java –**

 **Code Analysis on Main.java –**

**Database.java-**

Line: 22Use `PreparedStatement` to prevent SQL injection attacks.

Line: 22Use parameterized queries to prevent SQL injection attacks.

Use hashed or encrypted passwords for better security.

Line: 12Remove the usage of `Class.forName(JDBC\_DRIVER)` to load the JDBC driver.

Line: 3Use a logging framework for better logging and debugging

**Main.java-**

Line: 151Improve logging by using a Logger instead of System.out.println

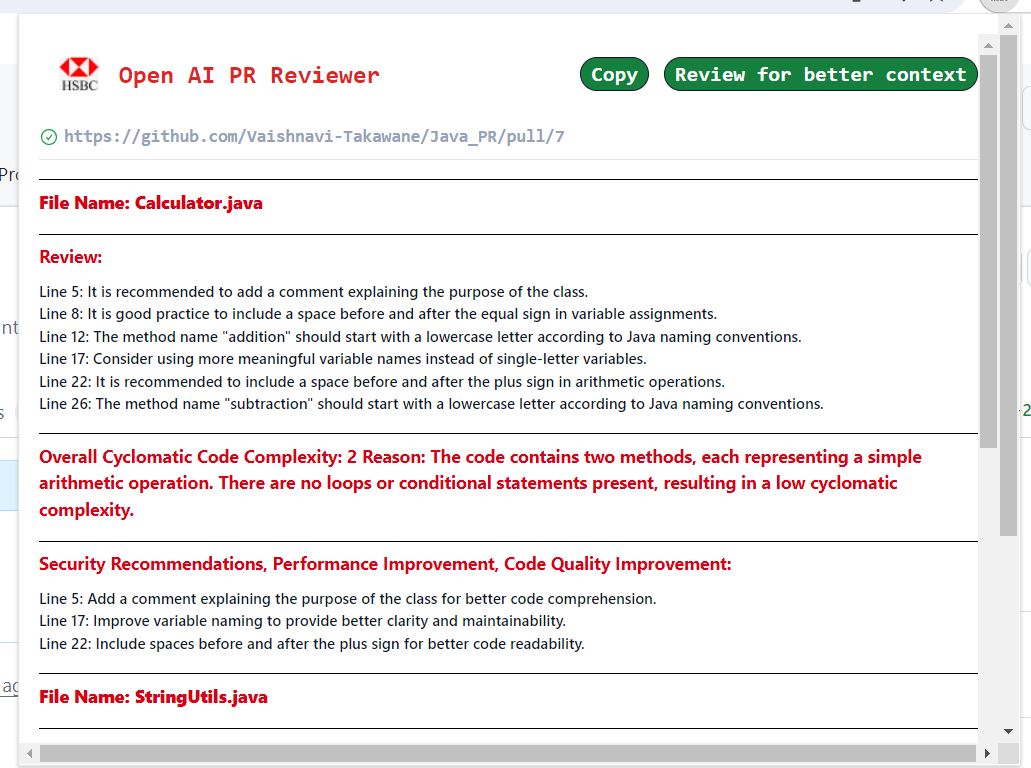
Line: 16Replace `switch` statements with `Map` or `Enum`

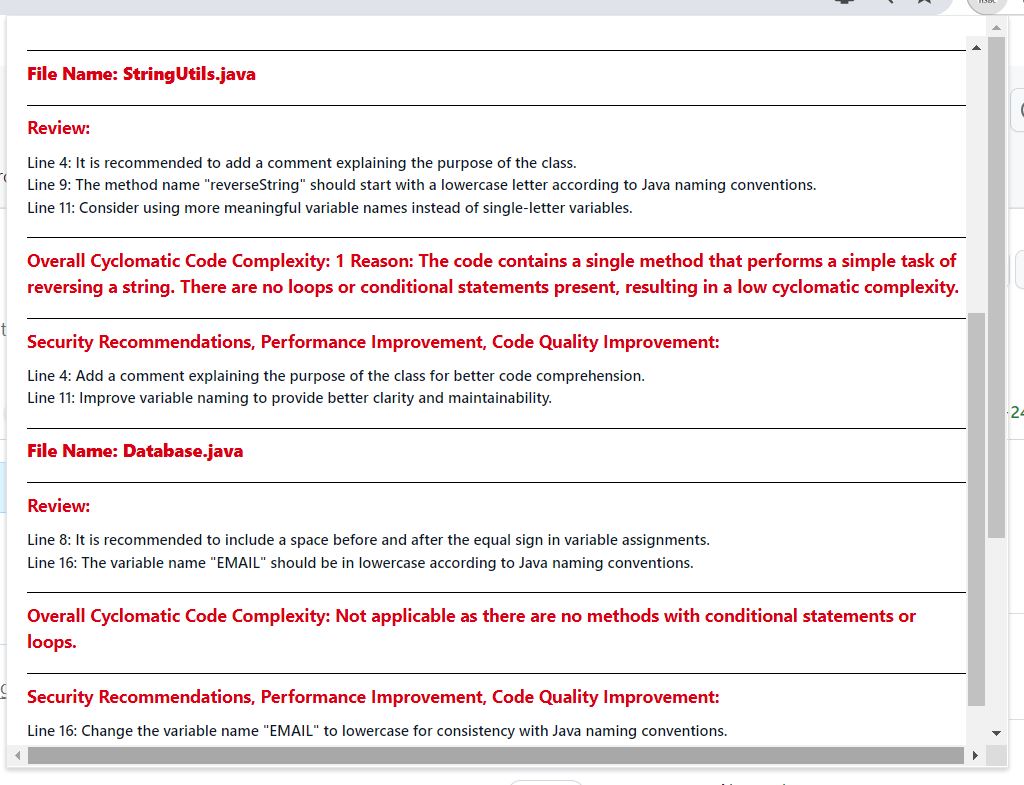
Line: 370Use `equalsIgnoreCase` for case-insensitive string comparison

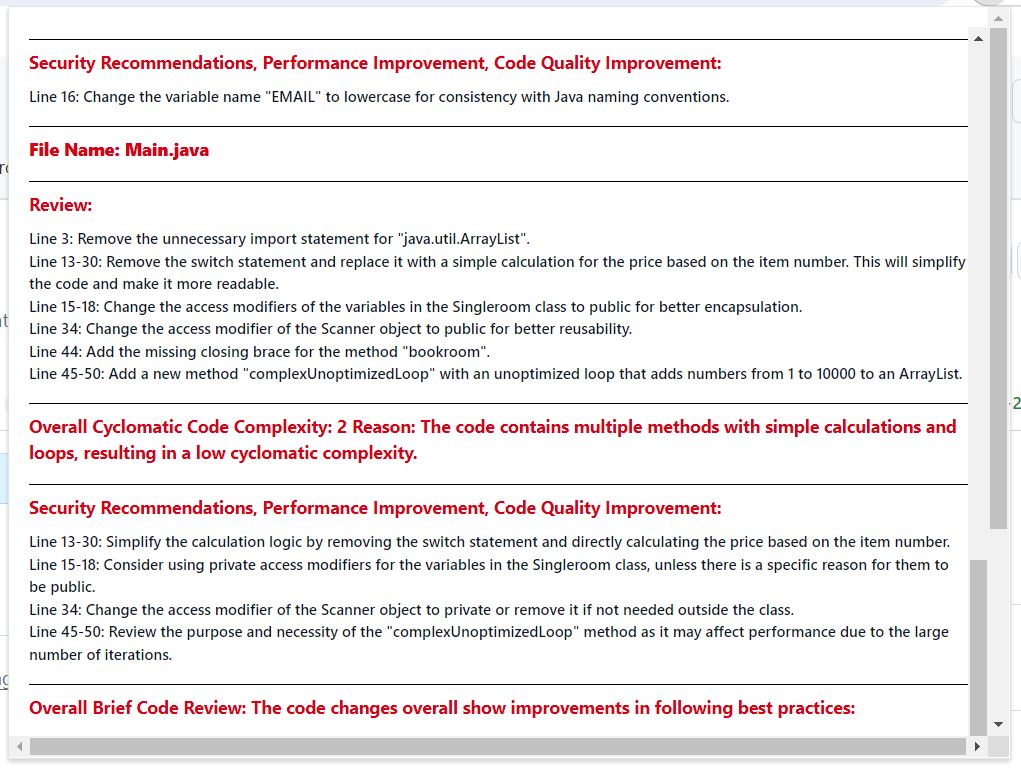
Line: 127Use Optional to handle null values

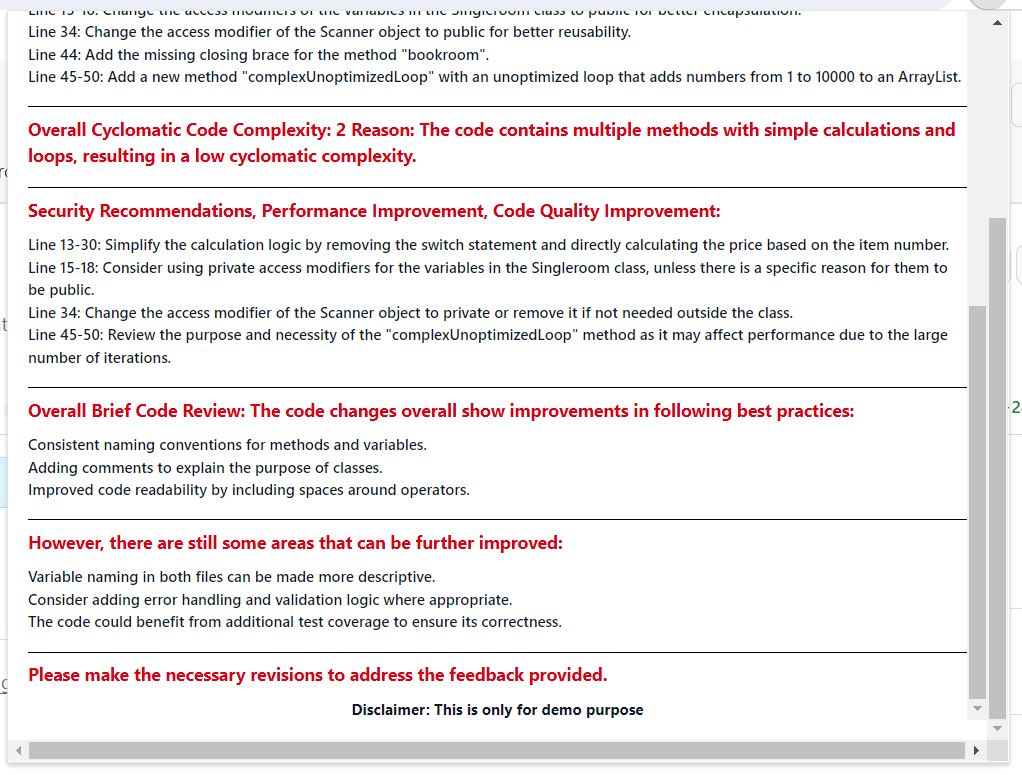
Use try-with-resources for FileInputStream and ObjectInputStream

**Open AI PR Reviewer- (Code Analysis)**









**File Name: Calculator.java**

**Review:**

Line 5: It is recommended to add a comment explaining the purpose of the class.

Line 8: It is good practice to include a space before and after the equal sign in variable assignments.

Line 12: The method name addition should start with a lowercase letter according to Java naming conventions.

Line 17: Consider using more meaningful variable names instead of single-letter variables.

Line 22: It is recommended to include a space before and after the plus sign in arithmetic operations.

Line 26: The method name subtraction should start with a lowercase letter according to Java naming conventions.

**Overall Cyclomatic Code Complexity:** 2 Reason: The code contains two methods, each representing a simple arithmetic operation. There are no loops or conditional statements present, resulting in a low cyclomatic complexity.

**Security Recommendations, Performance Improvement, Code Quality Improvement:**

Line 5: Add a comment explaining the purpose of the class for better code comprehension.

Line 17: Improve variable naming to provide better clarity and maintainability.

Line 22: Include spaces before and after the plus sign for better code readability.

**File Name: StringUtils.java**

**Review:**

Line 4: It is recommended to add a comment explaining the purpose of the class.

Line 9: The method name reverseString should start with a lowercase letter according to Java naming conventions.

Line 11: Consider using more meaningful variable names instead of single-letter variables.

**Overall Cyclomatic Code Complexity:** 1 Reason: The code contains a single method that performs a simple task of reversing a string. There are no loops or conditional statements present, resulting in a low cyclomatic complexity.

**Security Recommendations, Performance Improvement, Code Quality Improvement:**

Line 4: Add a comment explaining the purpose of the class for better code comprehension.

Line 11: Improve variable naming to provide better clarity and maintainability.

**File Name: Database.java**

**Review:**

Line 8: It is recommended to include a space before and after the equal sign in variable assignments. Line 16: The variable name EMAIL should be in lowercase according to Java naming conventions.

Overall Cyclomatic Code Complexity: Not applicable as there are no methods with conditional statements or loops.

**Security Recommendations, Performance Improvement, Code Quality Improvement:**

Line 16: Change the variable name EMAIL to lowercase for consistency with Java naming conventions.

**File Name: Main.java**

**Review:**

Line 3: Remove the unnecessary import statement for java.util.ArrayList.

Line 13-30: Remove the switch statement and replace it with a simple calculation for the price based on the item number. This will simplify the code and make it more readable.

Line 15-18: Change the access modifiers of the variables in the Singleroom class to public for better encapsulation.

Line 34: Change the access modifier of the Scanner object to public for better reusability.

Line 44: Add the missing closing brace for the method bookroom;.

Line 45-50: Add a new method complexUnoptimized Loop; with an unoptimized loop that adds numbers from 1 to 10000 to an ArrayList.

**Overall Cyclomatic Code Complexity**: 2 Reason: The code contains multiple methods with simple calculations and loops, resulting in a low cyclomatic complexity.

**Security Recommendations, Performance Improvement, Code Quality Improvement:**

Line 13-30: Simplify the calculation logic by removing the switch statement and directly calculating the price based on the item number. Line 15-18: Consider using private access modifiers for the variables in the Singleroom class, unless there is a specific reason for them to be public.

Line 34: Change the access modifier of the Scanner object to private or remove it if not needed outside the class.

Line 45-50: Review the purpose and necessity of the complexUnoptimized Loop; method as it may affect performance due to the large number of iterations.

**Overall Brief Code Review:** The code changes overall show improvements in following best practices:

Consistent naming conventions for methods and variables.

Adding comments to explain the purpose of classes.

Improved code readability by including spaces around operators.

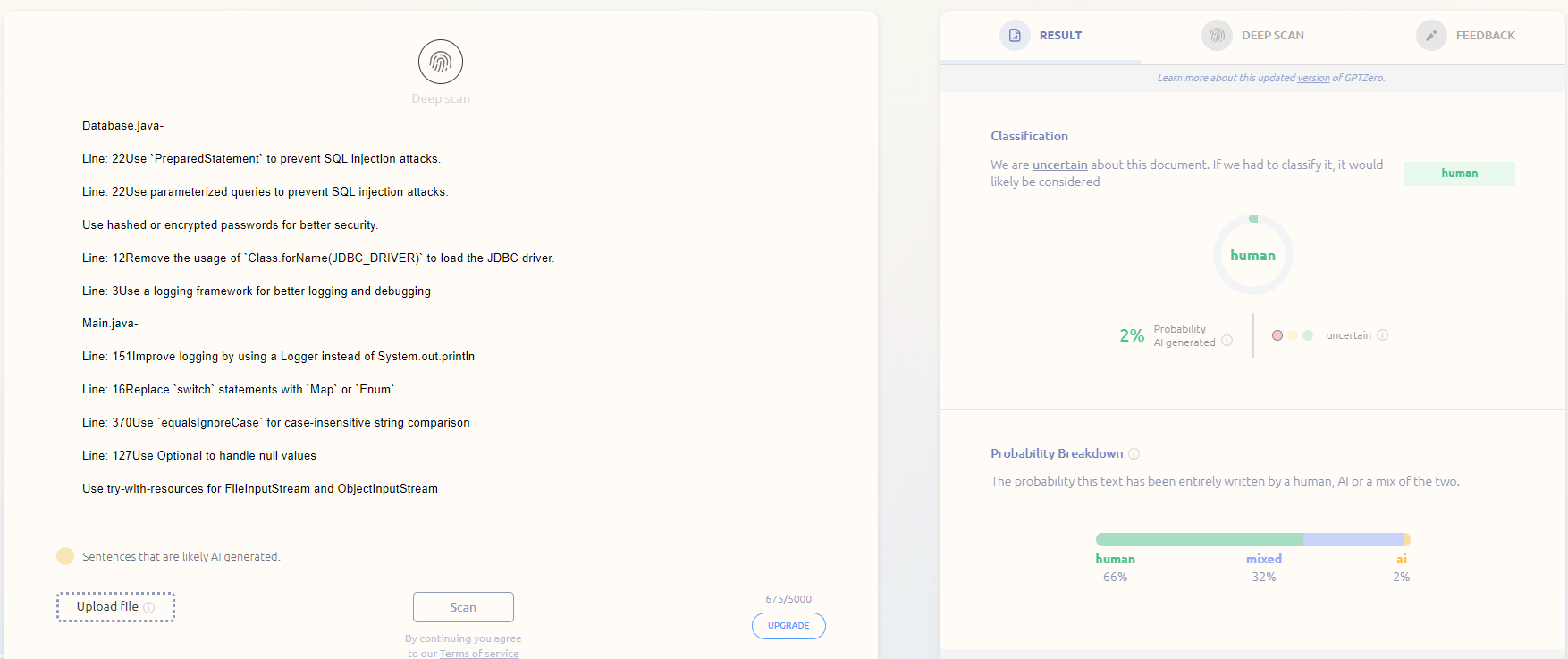
However, there are still some areas that can be further improved:

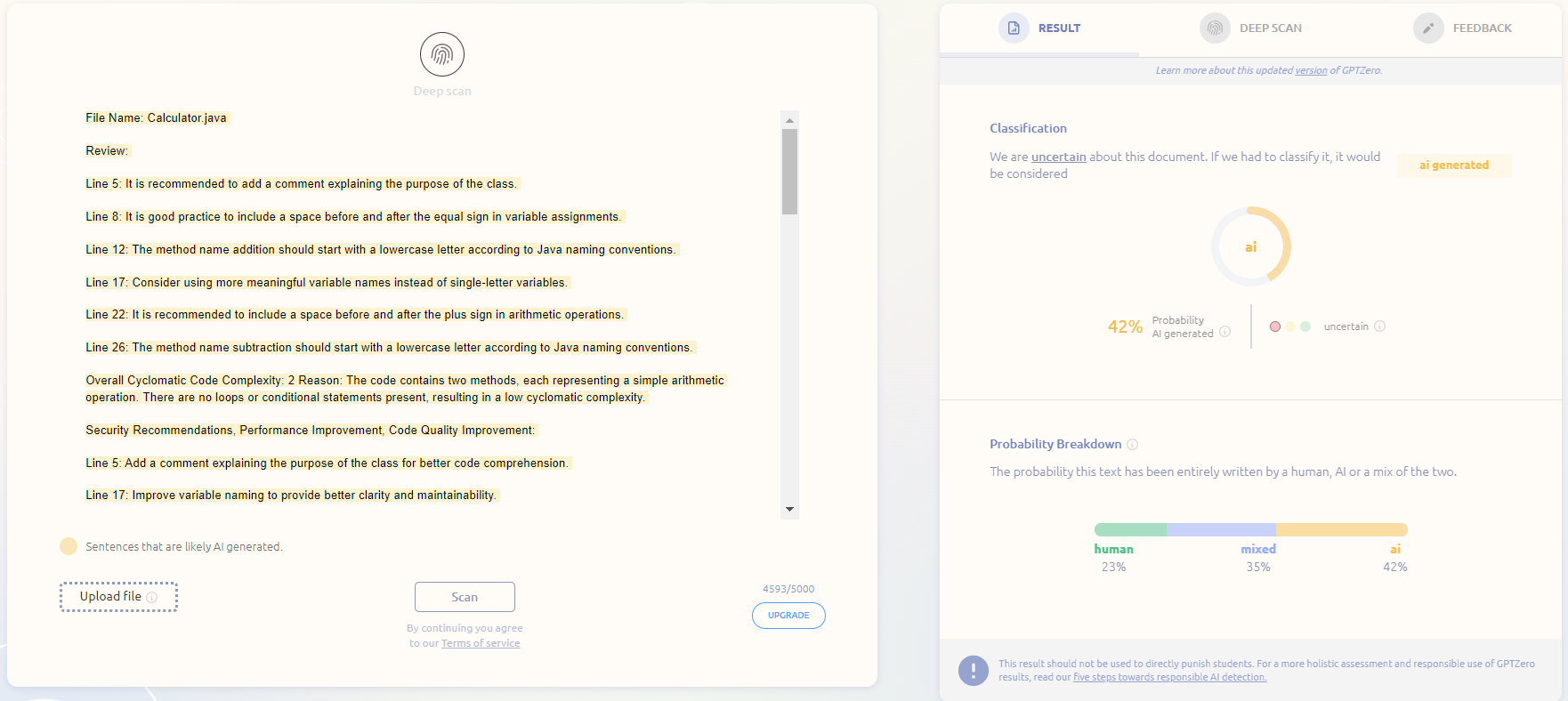
Variable naming in both files can be made more descriptive.

Consider adding error handling and validation logic where appropriate.

The code could benefit from additional test coverage to ensure its correctness.

Please make the necessary revisions to address the feedback provided.

**GPTZero result for codium analysis –**

**GPTZero result for OpenAI Pr Reviewer-**