**IO-Redback/src/Geothermal/ASHRAE/ASHRAEResultView.swift**
**IO-Redback/src/Geothermal/IGSHPAHB/IGSHPAHBResultView.swift**
**IO-Redback/src/Geothermal/IGSHPAHT/IGSHPAHTResultView.swift**
**IO-Redback/src/Geothermal/IGSHPAHT/IGSHPAHTResultView.swift**

1. Documentation Defects:
    a. Naming: The code names are meaningful.
    b. Comment: The code comments are a bit lacking. It's recommended to add comments to explain the purpose and functionality of important code sections.
2. Visual Representation Defects:
    a. Bracket Usage: The code has correct and consistent usage of brackets.
    b. Indentation: The code indentation appears to be consistent and readable.
    c. Long Line: There are no excessively long code statements.

3. Structure Defects:
    a. Dead Code: No unused or unnecessary code statements are found.
    b. Duplication: There are no instances of duplicate code statements that require refactoring.

4. New Functionality:
    a. Use Standard Method: The code follows a standardized method for single-purpose code statements.

5. Resource Defects:
    a. Variable Initialization: All variables are properly initialized before use.
    b. Memory Management: The code does not involve explicit memory management as it is using SwiftUI, which handles memory management automatically.

6. Check Defects:
    a. Check User Input: The code effectively validates and handles user input for validity.

7. Interface Defects:
    a. Parameter: The code correctly uses parameters when calling functions or libraries.

8. Logic Defects:
    a. Compute: The code accurately implements logic during system execution.
    b. Performance: The code exhibits an efficient performance.

Summary of recommended changes to improve code quality:
- Advice to comprehensive commenting to enhance code understanding and maintainability.
- Regularly review and refactor code to eliminate duplication.
- Perform thorough testing to ensure the accuracy of the logic and optimize performance when necessary.

**IO-Redback/src/Geothermal/MailView.swift**

1. Documentation Defects:
    a. Naming: The code names are meaningful.
    b. Comment: The code lacks comments explaining its functionality and purpose.

2. Visual Representation Defects:
    a. Bracket Usage: The code has correct and consistent usage of brackets.
    b. Indentation: The code indentation appears to be consistent and readable.
    c. Long Line: There are no excessively long code statements.

    c. Long Line: No long code statements were identified that hinder readability.

3. Structure Defects:
    a. Dead Code: No dead code statements.
    b. Duplication: No duplicate code statements.

4. New Functionality:
    a. Use Standard Method: There are no single-purpose code statements that require a standardized method.

5. Resource Defects:
    a. Variable Initialization: No uninitialized or incorrectly initialized variables were found in the code.
    b. Memory Management: The code does not involve explicit memory management as it is using SwiftUI, which handles memory management automatically.

6. Check Defects:
    a. Check User Input: The code does not explicitly handle user input validation.

7. Interface Defects:
    a. Parameter: No incorrect or missing parameters were identified in the code when calling functions or libraries.

8. Logic Defects:
    a. Compute: No incorrect logic during system execution was found in the code.
    b. Performance: The code does not contain complex algorithms that require performance evaluation.

Summary of recommended changes to improve code quality:
- Add comments to explain the functionality and purpose of the code.
- Consider adding input validation checks for user input if necessary.

**IO-Redback/src/Geothermal/TermsAndConditionsView.swift**

1. Documentation Defects:

a. Naming: The code names are meaningful.

b. Comment: The code lacks comments explaining the purpose of the code and its functionality. It would be beneficial to add comments to improve code understandability and maintainability.

2. Visual Representation Defects:

a. Bracket Usage: No issues found with bracket usage.

b. Indentation: The code indentation appears to be consistent and readable.

c. Long Line: The code contains long statements that hinder readability. It would be better to break them into multiple lines or restructure the code for improved clarity.

3. Structure Defects:

a. Dead Code: No dead code statements were found in the provided code.

b. Duplication: No duplicate code statements were found in the provided code.

4. New Functionality:

a. Use Standard Method: The code follows a standardized method for single-purpose code statements.

5. Resource Defects:

a. Variable Initialization: No uninitialized or incorrectly initialized variables were found in the code.

b. Memory Management: The code does not involve explicit memory management as it is using SwiftUI, which handles memory management automatically.

6. Check Defects:

a. Check User Input: The code does not explicitly handle user input validation. Consider adding input validation checks if necessary.

7. Interface Defects:

a. Parameter: No incorrect or missing parameters were identified when calling functions or libraries.

8. Logic Defects:

a. Compute: No incorrect logic during system execution was found in the code.

b. Performance: The code does not contain complex algorithms that require performance evaluation.

Summary of recommended changes to improve code quality:
- Add comments to explain the purpose and functionality of the code.
- Break long code statements into multiple lines for improved readability.
- Consider adding input validation checks for user input if necessary.