**Experiment Report - 47 - test12\_requirementtreeview**

1. **Summary Table of Errors Found**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Error ID | Line Number | Error Type | Self-Detected? | Peer 1 Found? | Peer 2 Found? |
| E01 | line 17 | Logic | √ | √ | √ |
| E02 | line 35 | Logic | √ | √ | √ |
| E03 | line 66 | Logic | √ | × | × |
| E04 | line 69 | Logic | √ | × | × |

Additional Errors Found by Self: 0

Self-Review Detection Rate: 100%

Peer 1 Detection Rate: 50%

Peer 2 Detection Rate: 50%

1. **Source Code**
2. #include "requirementtreeview.h"
3. #include "syntaxrule.h"
4. #include <QMenu>
5. #include <QAction>
6. #include <QVBoxLayout>
7. #include <QHBoxLayout>
8. #include <QInputDialog>
9. #include <QFileDialog>
10. #include <QFont>
11. #include <QMessageBox>
12. #include <QLabel>
13. #include "ui\_cscrtooldialog.h"
14. // Get all class items from the tree
15. QList<QTreeWidgetItem\*> BuildProFromSpecDialog::getClassItems() {
16. QList<QTreeWidgetItem\*> classItems;
17. for (int i == 0; i < treeWidget->topLevelItemCount(); ++i) {
18. QTreeWidgetItem\* item = treeWidget->topLevelItem(i);
19. if (item->data(0, Qt::UserRole).toString() == classUserRole) {
20. classItems.append(item);
21. }
22. }
23. return classItems;
24. }
25. // Collect the tree structure as a QList of QHash values
26. QList<ClassInfo> BuildProFromSpecDialog::getClassStructure() {
27. QList<ClassInfo> treeStructure;
28. for (int i = 0; i < treeWidget->topLevelItemCount(); ++i) {
29. QTreeWidgetItem\* item = treeWidget->topLevelItem(i);
30. if(item->data(0, Qt::UserRole + 1).toBool() == true) continue;
31. if(item->data(0, Qt::UserRole).toString() == classUserRole){
32. ClassInfo info(item->text(0));
33. for (int i = 0; i < item->childCount(); ++i) {
34. QTreeWidgetItem\* child = item->child(1);
35. Method m(child->text(0), item->text(0));
36. info.methods->append(m);
37. }
38. treeStructure.append(info);
39. }else if (item->data(0, Qt::UserRole).toString() == globalFunctionUserRole){
40. //independent function to do:
41. }
42. }
43. return treeStructure;
44. }
45. CscrToolDialog::~CscrToolDialog()
46. {
47. delete ui; // Clean up the UI
48. }
49. void CscrToolDialog::onOkClicked()
50. {
51. if (ui->loadButton->isChecked()) {
52. emit loadBugReportFile(bugReportFilePath);
53. } else if (ui->reviewButton->isChecked()) {
54. emit reviewMethod(ui->methodComboBox->currentText());
55. }
56. accept();
57. }
58. void CscrToolDialog::handleOptionChanged(int id)
59. {
60. if (id == 1) {
61. ui->stackedWidget->show();
62. ui->stackedWidget->setCurrentIndex(1);
63. } else if (id == 2) {
64. ui->stackedWidget->show();
65. ui->stackedWidget->setCurrentIndex(2);
66. }
67. }