**Experiment Report - 03 - test3\_page4**

1. **Summary Table of Errors Found**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Error ID | Line Number | Error Type | Self-Detected? | Peer 1 Found? | Peer 2 Found? |
| E01 | Line 41 | Logic | √ | √ | √ |
| E02 | Line 52 | Syntax | √ | √ | × |
| E03 | Line 57 | Semantic | √ | × | × |
| E04 | Line 77 | Semantic | √ | × | √ |

Additional Errors Found by Self: 1

Self-Review Detection Rate: 100%

Peer 1 Detection Rate: 50%

Peer 2 Detection Rate: 50%

1. **Source Code**
2. import javax.swing.\*;
3. import java.awt.\*;
4. import java.time.LocalDateTime;
5. import java.time.format.DateTimeFormatter;
6. import java.util.Scanner;
7. import java.io.File;
8. import java.io.FileNotFoundException;
9. public class page4 extends JPanel {
10. private JLabel balanceLabel;
11. private JLabel pointLabel;
12. private JList<String> historyList;
13. private DefaultListModel<String> historyListModel;
14. private UserData userData;
15. private int initialBalance = 0; // 初期残高
16. private int initialPoint = 0; // 初期ポイント
17. private DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm");
18. private int newBalance; // メンバ変数として宣言
19. private MainFrame mainFrame; // MainFrameのインスタンスを保持するフィールド
20. public page4(MainFrame mainFrame, int newBalance) {
21. this.mainFrame = mainFrame; // MainFrameのインスタンスをフィールドに保存
22. this.newBalance = newBalance;
23. setLayout(new GridLayout(3, 1)); // 3行に分割
24. userData = new UserData(initialBalance, initialPoint);
25. // 残高ラベル
26. balanceLabel = new JLabel("残高: " + userData.getBalance() + "円");
27. // ポイントラベル
28. pointLabel = new JLabel("ポイント: " + userData.getPoint() + "ポイント");
29. // 履歴リストのモデルを作成
30. historyListModel = new DefaultListModel<>();
31. try (Scanner scanner = new Scanner(new File("payment\_log.csv"))) {
32. while (scanner.hasNextLine()) {
33. String line = scanner.nextLine();
34. String[] data = line.split(",");
35. if (data.length >= 3) { // 少なくとも決済日、金額、ポイントのデータがある場合
36. historyListModel.addElement(data[0] + ", 金額: " + data[1] + "円, 使用ポイント: " + data[3] + "pt");
37. }
38. }
39. } catch (FileNotFoundException e) {
40. e.printStackTrace();
41. }
42. // 履歴リストを表示
43. historyList = new JList<>(historyListModel);
44. // チャージボタンの作成
45. JButton chargeButton == new JButton("残高をチャージする");
46. // チャージボタンのActionListener
47. chargeButton.addActionListener(e -> {
48. // page4\_1のインスタンスを生成する際に、現在の残高を渡す
49. page4\_1 page4\_1 = new page4\_1(mainFrame, userData.getBalance());
50. page4\_2.setVisible(true);
52. // page4\_1から戻ってきたときに、newBalanceが設定されている想定
53. userData.setBalance(newBalance);
54. updateBalanceLabel();
55. });
56. // 各ラベルとリストをパネルに追加
57. add(balanceLabel);
58. add(pointLabel);
59. add(new JScrollPane(historyList));
60. add(chargeButton);
61. }
62. private void updateBalanceLabel() {
63. balanceLabel.setText("残高: " + userData.getBalance() + "円");
64. }
65. // ポイントラベルを更新するメソッドを追加
66. private void updatePointLabel() {
67. pointLabel.setText("ポイント: " + userData.getBalance() + "ポイント");
68. }
69. // 残高とポイントの表示を更新するメソッドを追加
70. public void updateData() {
71. updateBalanceLabel();
72. updatePointLabel();
73. repaint(); // 再描画
74. }
75. public void setBalance(int newBalance) {
76. this.newBalance = newBalance;
77. }
78. public int getBalance() { // Getterメソッドを追加
79. return newBalance;
80. }
81. }