Your Answer

▼ Code review L1 25/10/24

- Parties
  - Reviewer: Shane
  - Author: Lala
  - Scribe: Nana
- Selected FilePanel.java and FileTable.java to critique (both linked to each other)
- Critiques/Resolutions
  - **Left Panel Alignment and Table Borders**

    C: The Left Panel in the first page table design is too much to the left + table borders

    D: Is working on familiarising herself with Java formatting and different layout options.

    The table can be adjusted by changing the layout of the upper panel because the table panel is being added to the main panel through the upper panel which may be confusing because the children panel might be overriding the parent panel so therefore the size might not be limiting properly.

    Will also adjust the padding and margin properties to address the table borders such as using BorderFactory.createEmptyBorder().

```
//----------------------Table Panel-----------------------

fileTable = new FileTable();
JScrollPane tableScrollPane = fileTable.getScrollPane();

//------------------------Sub Panels-------------------------

JPanel upperPanel = new JPanel();
upperPanel.setLayout(new BoxLayout(upperPanel,BoxLayout.Y_AXIS));
upperPanel.add(currDirPanel);
upperPanel.add(searchPanel);

JPanel lowerPanel = new JPanel();
lowerPanel.setLayout(new BorderLayout());
lowerPanel.add(tableScrollPane, BorderLayout.CENTER);

//------------------------Main Panel-------------------------

this.setLayout(new BorderLayout());
this.add(upperPanel, BorderLayout.NORTH);
this.add(lowerPanel, BorderLayout.CENTER);
```

- **Table interactivity**

  C: The first table on the first page is all interactive (it should not be, including extension, file size, etc.).

  D: From building the tables manually on JPanel, she will explore new solutions with JTable because it provides the set of components and features. There should be a built-in method to make the table not be editable. Will find a solution and apply to all rows/cells.

```
table = new JTable(data, columns);
table.setRowHeight(20);
table.setBackground(Color.white);

JTableHeader tableHeader = table.getTableHeader();
tableHeader.setBackground(Color.decode("#3E3E3E"));
tableHeader.setForeground(Color.white);

table.setFillsViewportHeight(true); // Ensures the table occupies full height in the viewport
//table.setBackground(Color.decode("#CFCFCF"));

scrollPane = new JScrollPane(table);
scrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);
scrollPane.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);
```

- **Interface Logic for Dynamic Data**

  C: Interface to logic not done yet, only has fixed data

  D: Will complete the UI interface so it can handle dynamic data and can also provide a the framework for the backend to be connected

```
//TO DO: make rows dynamic based on the number of files in the directory
//TO DO: be able to convert strings to integer and date format

String[] columns = {"Name", "Ext.", "Size", "Last Edited Date"};
Object[][] data =
        {
        {"Program", "C", "176B", "21.08.2019 17:00"},
        {"Document", "txt", "5KB", "10.11.2020 09:00"},
        {"Image", "png", "1.2MB", "05.06.2021 15:30"},
        {"Music", "mp3", "3MB", "12.12.2022 12:00"}
};
```

- **Row selection and borders**

  C: When row is selected, it highlights each cell by itself instead of the entire row and there are border between columns making it separated (confusing to user)

D: Will use JTable's features and methods to allow only selection of the row. Might have to implement custom features. The border between columns will also be fixed accordingly.

```
table = new JTable(data, columns);
table.setRowHeight(20);
table.setBackground(Color.white);

JTableHeader tableHeader = table.getTableHeader();
tableHeader.setBackground(Color.decode("#3E3E3E"));
tableHeader.setForeground(Color.white);

table.setFillsViewportHeight(true); // Ensures the table occupies full height in the viewport
//table.setBackground(Color.decode("#CFCFCF"));

scrollPane = new JScrollPane(table);
scrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);
scrollPane.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);
```

- **Separation of CSS and logic**

  C: Putting CSS and logic in the same file can may make maintenance and readability difficult

  D: Is unfamiliar with the way Java is structured so can decide together which part of code or which elements (UI elements, data handling, logic) should be separate.

```
public class FilePanel extends JPanel implements ActionListener
{
    JButton undoButton;
    JButton redoButton;
    MyTextField searchField;
    MyTextField suffixField;
    FileTable fileTable;

    FilePanel()
    {
```

```
    @Override
    public void actionPerformed(ActionEvent e)
    {
        //TO DO: implement undo and redo functionality
        if (e.getSource() == undoButton)
        {
            System.out.println("Undo button clicked");
        }
        else if (e.getSource() == redoButton)
        {
            System.out.println("Redo button clicked");

        }
        //TO DO: implement search functionality

        //TO DO: implement searching by suffix functionality
    }

}
```