Change request log

1. Concept Location

Step #	Description	Rationale
1	We compiled the code and ran the system.	
2	We interacted with system: creating files, exploring different functions.	To get familiar with some of the features of the system and identify the screens or graphical elements we had to change.
3	We searched for the term "Recent Files".	Because we identified the unique term "Recent Files" in the change request.
4	From the displayed 24 results in 13 files, we selected RecentFilesProvider class.	We selected this class because we found comment mentioning that this class is fore Recent file list menu.
5	We inspected the class RecentFilesProvider, and looked through it.	We noticed that there is a text variable which holds the TextField's class instance. On that class, the event listener is registered for keyReleased, which marks the files matching to the text as per the Regular Expression. So, this can be the class of interest.
6	We marked the debugger point at the event listener keyReleased().	We noticed that after writing some text inside the text box inside recent files menu, this event listener is being called and files are being marked as enabled as per the match of regular expression.
7	We marked the class RecentFilesProvider as "located".	We confirmed this class had to be modified.

Time spent (in minutes): 33

Classes and methods inspected:

- \org\gjt\sp\jedit\menu\RecentFilesProvider.java
    - void update()

2. Impact Analysis

Step #	Description	Rationale
1	We made a list of classes affected by a change in class RecentFilesProvider.	To track the classes that could be impacted by the change.
2	We searched for the term "RecentFilesProvider".	To list down the classes which are instantiating the class RecentFilesProvider, but didn't get any class which is making an instance of this class. Also, the change is changing the view level logic.
3	We inspected jedit_gui.props file from the search results and marked as "inspected and unchanged"	This file is linking the recent file menu button with the class RecentFilesProvider, and not using any property or method of the class.

Time spent (in minutes): 19

- \org\gjt\sp\jedit\menu\RecentFilesProvider.java
    - void update()
- \org\gjt\sp\jedit\jedit_gui.props

3. Actualization

Step #	Description	Rationale
1	Updated the regex to match the text anywhere in the file name.	Upon finding the exact regex that was being used in matching with the file name, we have updated the regex to match with the text anywhere in the file name not only to the beginning of the file name. * text * at the beginning and end of the regex pattern, which allows for matching any characters before and after the typedText. We have used Pattern.quote(typedText) to ensure that any special characters in the typedText are treated as literals, which is necessary if the file names can contain regex special characters.
2	Updated the logic to enable/highlight the menu items based on matching text.	After implementing the regex to match with the text present anywhere in the file name, we have updated the logic which is responsible for enabling the menu items based on the matching file names. We have replaced pattern.matcher(recent.getText()).matches() with pattern.matcher(recent.getText()).find() to check for the occurrence of the pattern anywhere in the file name rather than matching the entire string also added the logic if nothing is typed then enabling all the menu items.
3	We performed functional testing and manual testing.	To make sure everything works.

Time spent (in minutes): 20

Inspected: -

- \org\gjt\sp\jedit\menu\RecentFilesProvider.java
  - void update()
- \org\gjt\sp\jedit\jedit_gui.props

Inspected and Changed: -

- \org\gjt\sp\jedit\menu\RecentFilesProvider.java
  - void update()

4. Validation

Step #	Description	Rationale
1	Test case defined: All those items should enable which contains the entered text anywhere in the file name. Expected output: All those items are getting enabled.	This is the regular expected behavior. The test passed.
2	Test case defined: No item should enable if there no matching text in any file name. Expected output: No item is enabling when there is no matching text.	This is the regular expected behavior. The test passed.
3	Test case defined: All items should be enabled if nothing is entered.	This is the regular expected behavior.

	Expected output: All items are getting enabled when nothing is added.	The test passed.
4	Test case defined: Matching items should enable if any special character is typed. Expected output: Items with matching special symbol are getting enabled.	This is the regular expected behavior. The test passed.
5	Test case defined: Item should enable if the text matching at the very end of file name. Expected output: Item is enabling when there is matching text at the end of file name.	This is the regular expected behavior. The test passed.
6	Test case defined: Item should enable if the text matching at the middle of file name. Expected output: Item is enabling when there is matching text at the middle of file name.	This is the regular expected behavior. The test passed.
7	Test case defined: Item should enable if the text matching at the start of file name. Expected output: Item is enabling when there is matching text at the start of file name.	This is the regular expected behavior. The test passed.

Time spent (in minutes): 20

5. Summary of the change request

Phase	Time (minutes)	No. of classes inspected	No. of classes changed	No. of methods inspected	No. of methods changes
Concept location	33	1	0	1	0
Impact Analysis	19	1	0	1	0
Actualization	20	1	1	1	1
Verification	20	1	0	1	0
Total	92	4	1	4	1

6. Conclusions

For this change, the concept location was not as challenging as the file for the recent files provider, we just need to find the exact method for implementing the change. The Impact analysis was the easiest part as this class was not being used very much in other files. The Actualization phase was also easy as we have to modify the regex to match with the whole file name rather than matching with the beginning of the file name. The verification process was the crucial aspect of this change as we must check all possible cases for matching the string along with the special characters also. Overall, this change was easily doable.