Virtual Key for Your Repositories

The Lockedme.com application is finished up for Locker Pvt. Ltd. In the aim to digitize their products. The application need to provide the user options to view the files in the folder in ascending order, do file operations in the folder such as to add a file, delete a file and look for a file. The user should also be able to close the application.

The project needs to be divided into multiple sprints to ensure the proper workflow and success of the project. The project is completed in a total of 3 sprints with each sprint as long as 5 days. The work done during each sprint can be shown as below:

SPRINT 1 :

- Build a welcome screen for the application.
- Add the application name and Developer details
- Display the Main Menu with 3 options.
- Add option 1 in main menu (Show files in ascending order)
- Support closing the application.

SPRINT 2:

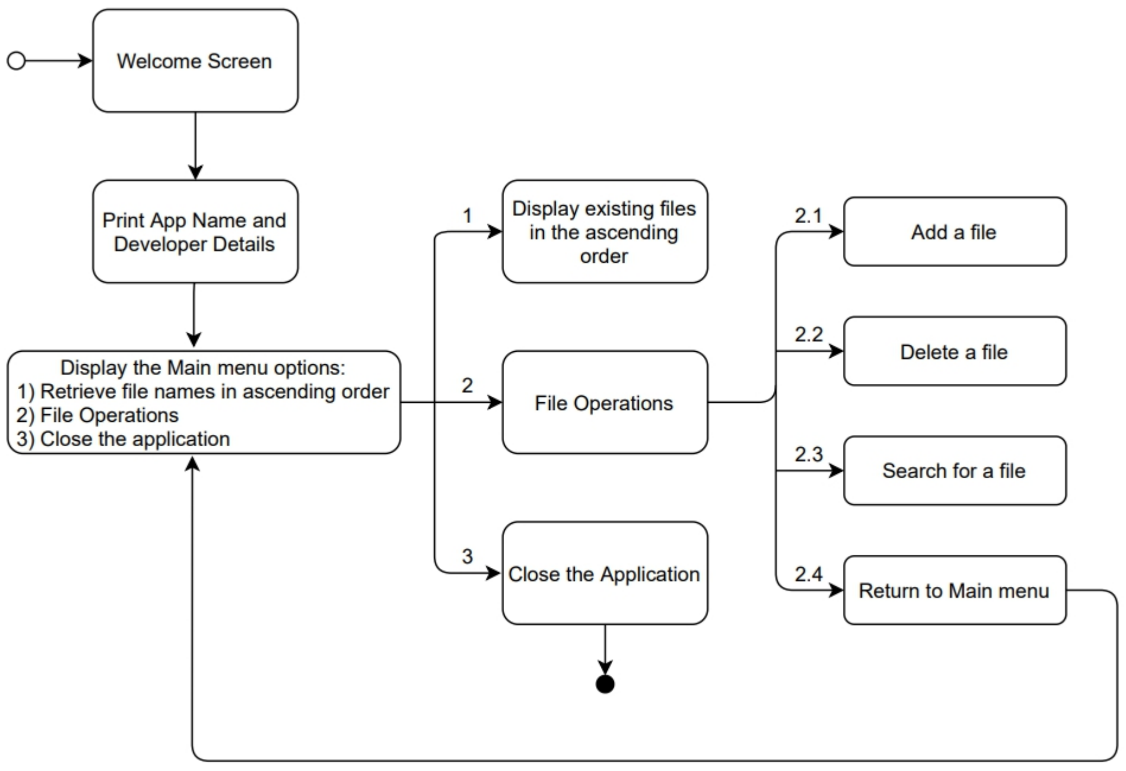
- Add option 2 in main menu (File Operations)
- Display File Operations Menu
- Add option 1 in file operations to Add a file
- Add option 2 in file operations to Delete a file
- Add another option in the file operations menu to take the user back to Main Menu

SPRINT 3:

- Add an option in the file operations menu to search for a file
- Work on the overall functioning and transitions of the application.
- Check case sensitivity and other important details of each phase
- Test the working
- Get the approval of the Scrum Master
- Finish up the project

The flowchart for the same project is displayed below with each operation listed. The flowchart gives you a better understanding of the sprints and the flow of the application. The program flow starts from the hollow circle and ends at the filled circle. A third party can understand the sprint division and flow of the application from the flowchart easily.

FLOWCHART:



CODE REVIEW AND METHODOLOGIES USED:

- Methods are used to display the welcome screen and developer details, the Main Menu of the application containing 3 operations, and also the file operations menu.

- For the main menu, the option to be proceeded is taken from the user. The user can display the files in ascending order, perform file operations or close the application. If the user provides an option that is not listed, the application gives invalid input warning and asks the user to input any of the given options.

- The program steps into the file operations method when the user chooses File Operations from the Main Menu. The user is given options to add a file to the directory, delete a file from the directory and to search for a file in the directory. The user can also go back to the main menu from this method. If the user provides an option that is not listed, the application gives invalid input warning and asks the user to input any of the given options.

- A folder called HCLFolder is created and it can have some files in it as well as empty. The path to the Folder is defined. **TreeSet()** constructor is used to build an empty TreeSet object, here sorted, in which elements will get stored in default natural sorting order.

- The createset() method creates a treeset of all the files in our folder. The **File(FOLDER).listFiles()** method creates a list of pathnames of all the files in the FOLDER and makes it to an array 'files' which now contains all the files in the FOLDER. The for loop checks if every element in the folder is a file, and if it returns true the file is added to the treeset sorted.

- The **filesInAscending** order takes the treeset with all the files in the order from the createset() method, since a treeset sorts the list of files in its natural order already, you just have to print the files in this method.

- An alternate method shows how the same sorting can be done using Array instead of a Treeset. The arraylist has a time complexity of O(n) for .contains(), that comes up later, whereas the same for Treeset is O(logn). For the input of size n, an algorithm of O (n) will perform steps proportional to n, while another algorithm of O (log (n)) will perform steps roughly log (n). Clearly log (n) is smaller than n hence algorithm of complexity O (log (n)) is better. Since it will be much faster.

- The **addFile** method is to add a file to the folder. The file to be added is specified by its path and is given by the user. The method checks if the path exists and if it returns true, the file is added to the folder.

- In the **deleteFile** method, the set of files in the folder is initialized using the createset() method. The file to be deleted is then taken from the user. The **.contains()** method checks whether the given file exist in the folder and if this returns true, the file is deleted using **file.delete().** the deleted file is also removed from our set using **.remove().** Case sensitivity of the file to be deleted is taken into account. The method returns a FNF ("File not found") if the file does not exist in the folder.

- In the **searchFile** method, the set of files in the folder is initialized using the createset() method. The file to be searched for is then taken from the user. The .contains() method checks whether the given file exist in the folder and if this returns true, output "File Found in Directory" is printed and the path of the same file is displayed alongside. If the return value is false, the output "File not Found in Directory" is printed.

The link to the github repository: https://github.com/anita2498/LockedMe_PhaseEnd.git