LockedMe – Virtual Key for Repositories

(Project code Document)

Version History

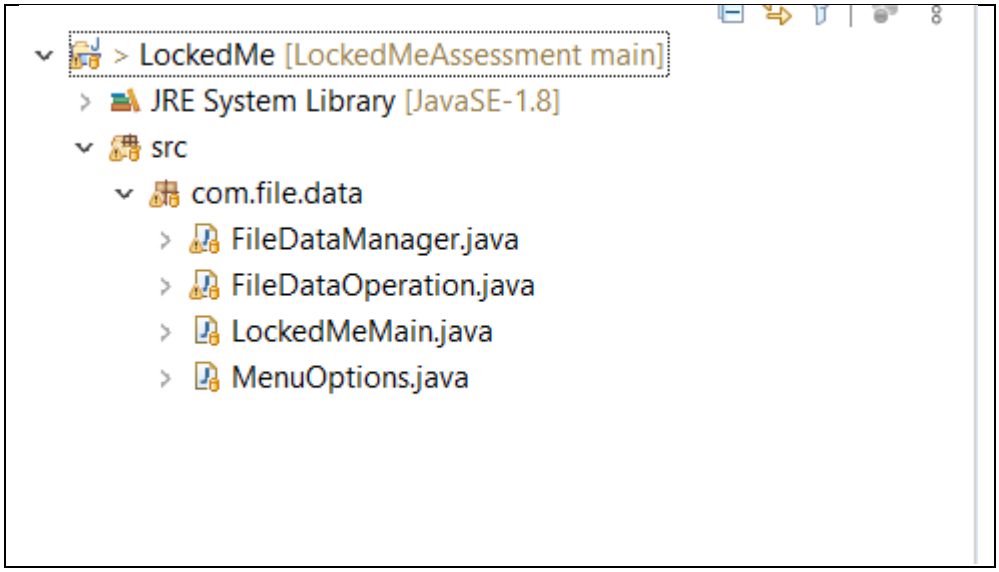
Author	Rabin Kumar Pati
Doc purpose	Source Code Document
Date	13-Aug -2021
Version	01

Table of Contents

1. Document Contents:

- Folder Structure in Project Workspace.
- Project GitHub details.
- Source Code

1.1 Folder Structure



1.2 Project GitHub Details

Repositories Name	RABINPATI/LockedMeAssessment
GitHub Link	https://github.com/RABINPATI/LockedMeAssessment

1.3 Source Code

1.3.1 LockedMeMain.java

LockedMeMain.java

```java
package com.file.data;

public class LockedMeMain {

    // Folder Path for file to access data

    static final String FOLDER_PATH = "D:\\Project\\FSD\\Phase 1\\LockedMeAssessment\\LockedMe

    public static void main(String[] args) {

        // Display Application name

        MenuOptions.displayWelcomeHeaderNote();

        // Display file processing operations

        MenuOptions.menuProcessOperations(FOLDER_PATH);

    }

}
```

1.3.2 MenuOptions.java

```java
package com.file.data;

import java.text.SimpleDateFormat;

public class MenuOptions {

    /**
     * Add the details for application Name in header part.
     */

    public static void displayWelcomeHeaderNote() {

        System.out.println("_____" + "\n");
        System.out.println("                    Application Name: LockedMe.com");
        System.out.println("_____");
    }

    /**
     * Add the details about developer and date in below part.
     */
    public static void displayWelcomeFooterNote() {

        System.out.println("_____");
        System.out
                .println("                                                    Developed By RABIN");
        System.out.println("                                                      Date:"
                + new SimpleDateFormat("dd-MMM-yyyy").format(new Date()));
        System.out.println("_____");

    }

}
```

```java
    /**
     * Display the menu options
     */

    public static void displayFileMenuOptions() {
        String fileMenu = "\n\n****** Select any number from below for file operation and press Ente
                + "1) Retrieve all files \n"
                + "2) Add a file \n"
                + "3) Search a file \n"
                + "4) Delete a file \n"
                + "5) Exit program\n";

        System.out.println(fileMenu);

    }

    /**
     *
     * @param folderPath
     * File processing operation
     * Retrieve files
     * Add files,Search file, Delete file
     */
     public static void menuProcessOperations(String folderPath) {

                boolean running = true;
                Scanner sc = new Scanner(System.in);
                do {
                    try {
                        MenuOptions.displayFileMenuOptions();
                        System.out.println("Enter an option to proceed");
                        int input = sc.nextInt();

                        switch (input) {
                        case 1:
                            // Display all files from LockedMeFileData folder.

                            FileDataOperation.displayFileNames(folderPath);
                            break;
                        case 2:
                            // Add a file into LockedMeFileData folder.

                            FileDataOperation.createFiles(folderPath);
                            break;
                        case 3:
                            // Search a file from LockedMeFileData folder.

                            FileDataOperation.searchFile(folderPath);
                            break;
                    case 4:
                        //Delete a file from LockedMeFileData folder.

                        FileDataOperation.deleteFile(folderPath);
                        break;

                    case 5:
                        System.out.println("Program exited successfully.");
                        running = false;
                        sc.close();
                        displayWelcomeFooterNote();
                        System.exit(0);

                        break;
                    default:
                        System.out.println("Please select a valid option from above.");
                    }
                    } catch (Exception e) {
                        System.out.println("Please select a valid option from above.");
                        menuProcessOperations(folderPath);
                    }
                } while (running == true);
     }

}
```

1.3.3 FileDataOperation.java

LockedMeMain.java | MenuOptions.java | FileDataOperation.java | FileDataManager.java

```java
package com.file.data;

import java.util.ArrayList;

public class FileDataOperation {

    /**
     * @param folderPath
     * Add file
     */

    public static void createFiles(String folderPath) {

        Scanner scanObj = new Scanner(System.in);

        String file_Name;
        int noOfLines;
        List<String> dataContentList = new ArrayList<String>();

        try {
            System.out.println("Enter File Name to create:");
            file_Name = scanObj.nextLine();

            // Read number of Lines from User

            System.out.println("Enter how many lines in the file");
            noOfLines = Integer.parseInt(scanObj.nextLine());

            if (noOfLines > 0) {
                // Read data contents from User
                for (int i = 1; i <= noOfLines; i++) {

                    System.out.println("Enter line" + i + ": ");
                    dataContentList.add(scanObj.nextLine());
                }
                // Save the data content into file

                boolean isFileSaved = FileDataManager.createFiles(folderPath, file_Name, dataContent
                if (isFileSaved)
                    System.out.println("File : " + file_Name + "  saved successfully");
                else {
                    System.out.println("File not saved due to error occued. Please check log for mor
                }
            }
        } catch (Exception e) {
            System.out.println("Please insert file name to create");
        }

    }
    /**
     * @param folderPath
     * Display Allfiles
     */

    public static void displayFileNames(String folderPath) {

        List<String> fileNamesList = FileDataManager.getAllFiles(folderPath);

        // shorting the file in ascending order

        Collections.sort(fileNamesList);

        if (!fileNamesList.isEmpty()) {
            fileNamesList.forEach(fileName -> {
                System.out.println(fileName);
            });
        } else {
            System.out.println("Files not found in directory");
        }
    }
```

```java
    /**
     * @param folderPath
     * Search File
     */
    public static void searchFile(String folderPath) {

        Scanner scanObj = new Scanner(System.in);

        String file_Name;

        try {
            System.out.println("Enter File Name to be search:");
            file_Name = scanObj.nextLine();

            // Search the file from folder

            boolean isFileAvailable = FileDataManager.searchFile(folderPath, file_Name);

            if (isFileAvailable)
                System.out.println(file_Name + " : found in directory");
            else {
                System.out.println(file_Name + " : not found in directory");
            }

        } catch (Exception e) {
            System.out.println("Please insert file name to search");
        }

    }
    /**
     * @param folderPath
     * Delete File
     */
    public static void deleteFile(String folderPath) {

        Scanner scanObj = new Scanner(System.in);

        String file_Name;

        try {
            System.out.println("Enter File Name to delete:");
            file_Name = scanObj.nextLine();

            // Delete the file from folder

            boolean isFileDeleted = FileDataManager.deleteFile(folderPath, file_Name);

            if (isFileDeleted)
                System.out.println(file_Name + " : deleted successfully");
            else {
                System.out.println(file_Name + " : not found in directory");
            }
            // Close the scanner object
        } catch (Exception e) {
            System.out.println("Please insert file name to delete.");
        }

    }

}
```

1.3.4 FileDataManager.java

LockedMeMain.java | MenuOptions.java | FileDataOperation.java | FileDataManager.java

```java
package com.file.data;

import java.io.File;

public class FileDataManager {

    /**
     *
     * @param folderPath
     * @param fileName
     * @param dataContent
     * @return file in directory
     */

    public static boolean createFiles(String folderPath, String fileName, List<String> da

        try {
            File folderName = new File(folderPath, fileName);
            FileWriter fw = new FileWriter(folderName);

            dataContent.forEach(data -> {
                try {
                    fw.write(data + "\n");
                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            });
            fw.close();
            return true;

        } catch (Exception ex) {
            ex.getMessage();
            return false;

        }
    }
    /**
     *
     * @param folderPath
     * @return all files
     */
    public static List<String> getAllFiles(String folderPath) {

        File folder = new File(folderPath);
        File[] listOfFiles = folder.listFiles();

        List<String> filesInFolder = new ArrayList<String>();
        for (File file : listOfFiles) {
            if (file.isFile()) {
                String fileName = file.getName();
                filesInFolder.add(fileName);
            }
        }
        return filesInFolder;

    }
    /**
     *
     * @param folderPath
     * @param file_Name
     * @return true/false based on search results.
     */
    public static boolean searchFile(String folderPath, String file_Name) {

        List<String> filesInFolderList = getAllFiles(folderPath);

        boolean isSerached = false;
```

```
        for (String fileName : filesInFolderList)
        {
            // Check user input file name should match with directory file.

            if (fileName.equals(file_Name)) {
                File fileDetails = new File(folderPath + "\\" + file_Name);

                if (fileDetails.exists())
                    isSerached = true;
                    break;
            } else {
                isSerached = false;
            }
        }
        return isSerached;
    }
    /**
     *
     * @param folderPath
     * @param file_Name
     * @return true/false based on delete results.
     */

    public static boolean deleteFile(String folderPath, String file_Name) {

        // Check all files in directory for case sensitivity.

        List<String> filesInFolderList = getAllFiles(folderPath);

        boolean isDeleted = false;

       for (String fileName : filesInFolderList)
       {
           // Check user input file name should match with directory file.
           if (fileName.equals(file_Name)) {

               File fileDetails = new File(folderPath + "\\" + file_Name);

               if (fileDetails.exists()) {
                   if (fileDetails.delete())
                       isDeleted = true;
                       break;
                   }
               } else {
                   isDeleted = false;
               }
       }
       return isDeleted;
    }
}
```