LockedMe.com – Virtual Key for Repositories

This document contains sections for:

The code for this project is hosted at https://github.com/AmitDhanorkar/LockedMe.com
The project is developed by Amit Dhanorkar.

Sprints planning and Task completion

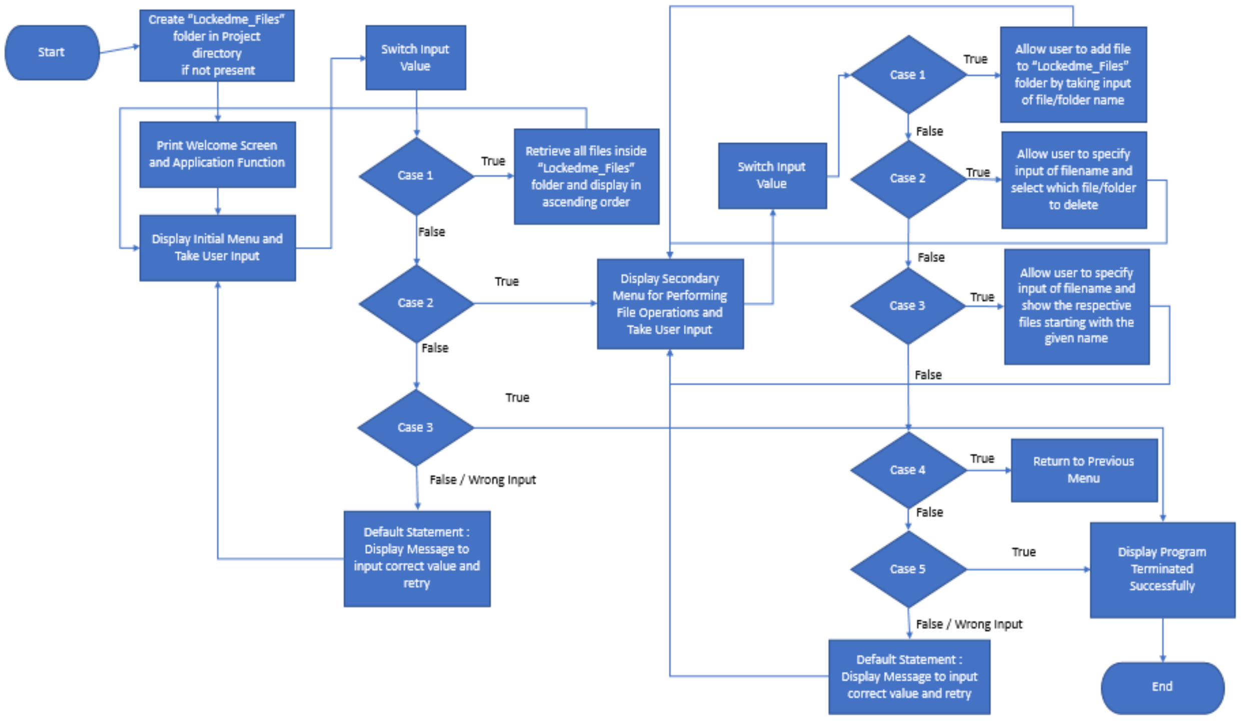
The project is planned to be completed in 1 sprint. Tasks assumed to be completed in the sprint are:

- Creating the flow of the application
- Initializing git repository to track changes as development progresses.
- Writing the Java program to fulfill the requirements of the project.
- Testing the Java program with different kinds of User input
- Pushing code to GitHub.
- Creating this specification document highlighting application capabilities, appearance, and user interactions.

Core concepts used in project

Collections framework, File Handling, Sorting, Flow Control, Properties, Exception Handling, Streams API, Singleton class

Flow of the Application


Start
Create "Lockedme_Files" folder in Project directory if not present
Print Welcome Screen and Application Function
Display Initial Menu and Take User Input
Switch Input Value
Case 1
True
Retrieve all files inside "Lockedme_Files" folder and display in ascending order
False
Case 2
True
Display Secondary Menu for Performing File Operations and Take User Input
False
Case 3
True
False / Wrong Input
Default Statement : Display Message to input correct value and retry
Switch Input Value
Case 1
True
Allow user to add file to "Lockedme_Files" folder by taking input of file/folder name
False
Case 2
True
Allow user to specify input of filename and select which file/folder to delete
False
Case 3
True
Allow user to specify input of filename and show the respective files starting with the given name
False
Case 4
True
Return to Previous Menu
False
Case 5
True
Display Program Terminated Successfully
False / Wrong Input
Default Statement : Display Message to input correct value and retry
End

Demonstrating the product capabilities, appearance, and user interactions

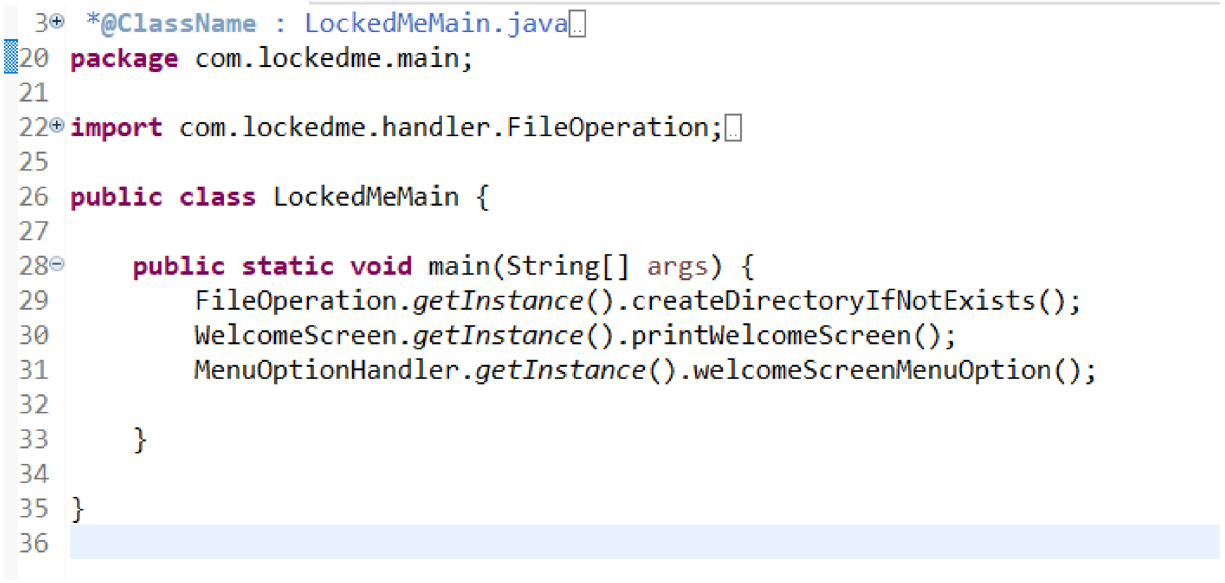
To demonstrate the product capabilities, below are the sub-sections configured to highlight appearance and user interactions for the project:

1 Creating the project in Eclipse
2 Writing a program in Java for the entry point of the application (**LockedMeMain.java**)
3 Writing a program in Java to display Menu options available for the user (**WelcomeScreen.java**)

Step 1: Creating a new project in Eclipse

- Open Eclipse
- Go to File -> New -> Project -> Java Project -> Next.
- Type in any project name and click on "Finish."
- Select your project and go to File -> New -> Class.
- Enter **LockedMe.com** in any class name, check the checkbox "public static void main(String[] args)", and click on "Finish."

Step 2: Writing a program in Java for the entry point of the application (**LockedMeMain.java**)



```
 *@ClassName : LockedMeMain.java
package com.lockedme.main;

import com.lockedme.handler.FileOperation;

public class LockedMeMain {

    public static void main(String[] args) {
        FileOperation.getInstance().createDirectoryIfNotExists();
        WelcomeScreen.getInstance().printWelcomeScreen();
        MenuOptionHandler.getInstance().welcomeScreenMenuOption();

    }

}
```

Step 3: Writing a program in Java to display Menu options available for the user (**MenuOptionHandler.java**)

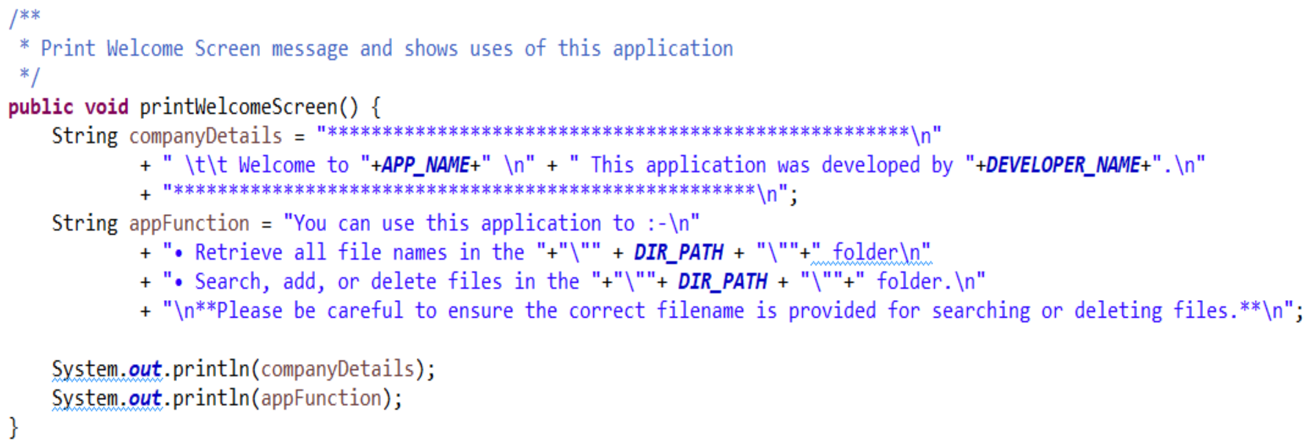
- Select your project and go to File -> New -> Class.
- Enter **WelcomeScreen** in class name and click on "Finish."

- **WelcomeScreen** consists methods for -: **3.1** Displaying Welcome Screen

3.2 Displaying Initial Menu

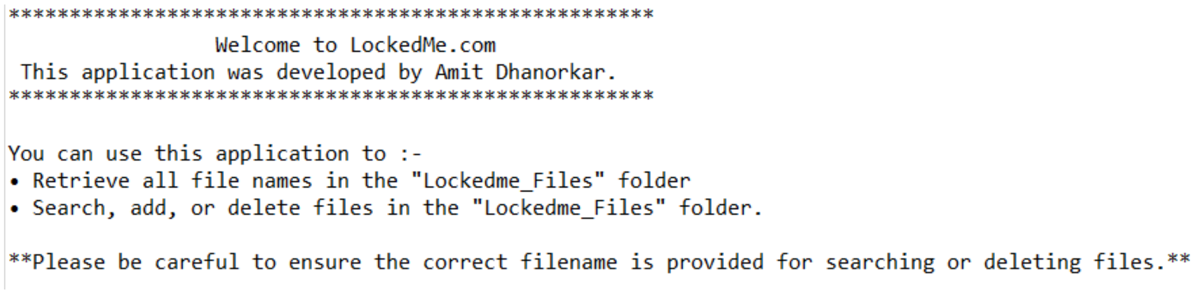
3.3 Displaying Secondary Menu for File Operations available

Step 3.1: Writing method to display Welcome Screen



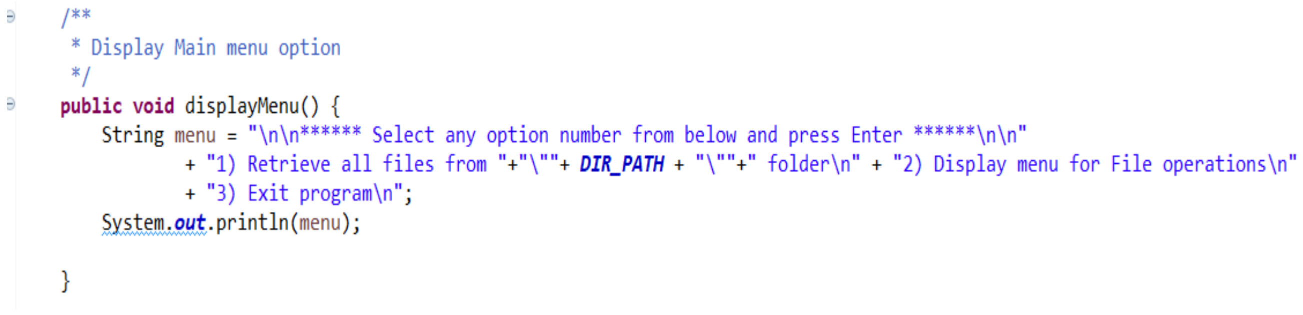
```
/**
 * Print Welcome Screen message and shows uses of this application
 */
public void printWelcomeScreen() {
    String companyDetails = "*****************************************************\n"
            + " \t\t Welcome to "+APP_NAME+" \n" + " This application was developed by "+DEVELOPER_NAME+".\n"
            + "*****************************************************\n";
    String appFunction = "You can use this application to :-\n"
            + "• Retrieve all file names in the "+"\"" + DIR_PATH + "\""+" folder\n"
            + "• Search, add, or delete files in the "+"\""+ DIR_PATH + "\""+" folder.\n"
            + "\n**Please be careful to ensure the correct filename is provided for searching or deleting files.**\n";

    System.out.println(companyDetails);
    System.out.println(appFunction);
}
```

Output:



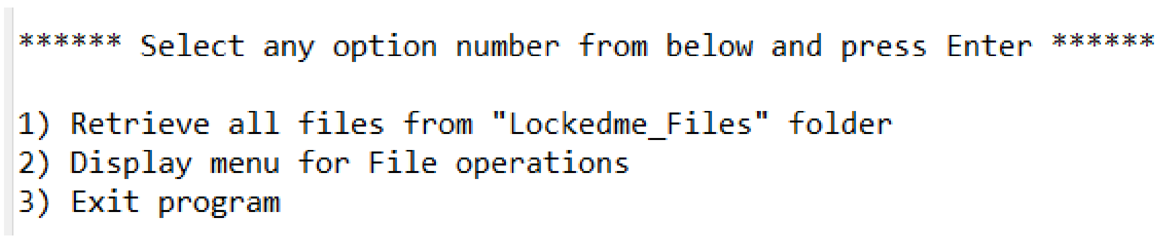
```
*****************************************************
                 Welcome to LockedMe.com
 This application was developed by Amit Dhanorkar.
*****************************************************

You can use this application to :-
• Retrieve all file names in the "Lockedme_Files" folder
• Search, add, or delete files in the "Lockedme_Files" folder.

**Please be careful to ensure the correct filename is provided for searching or deleting files.**
```

Step 3.2: Writing method to display Initial Menu



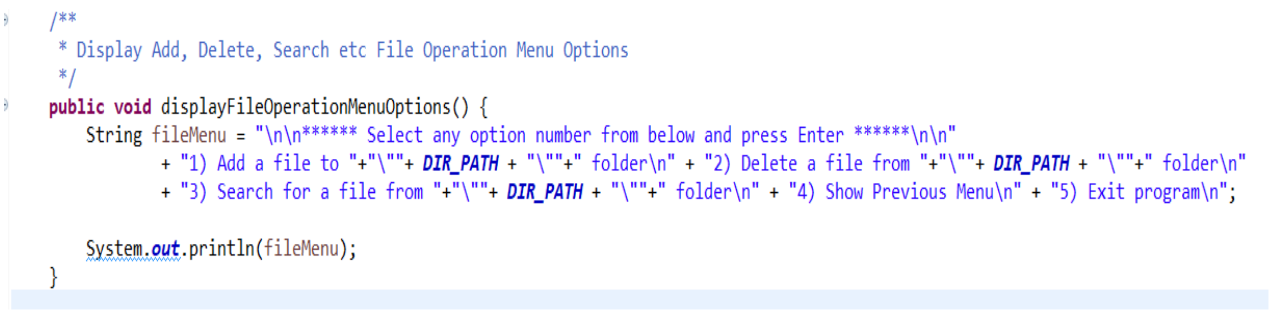
```
/**
 * Display Main menu option
 */
public void displayMenu() {
    String menu = "\n\n****** Select any option number from below and press Enter ******\n\n"
            + "1) Retrieve all files from "+"\""+ DIR_PATH + "\""+" folder\n" + "2) Display menu for File operations\n"
            + "3) Exit program\n";
    System.out.println(menu);

}
```

Output:



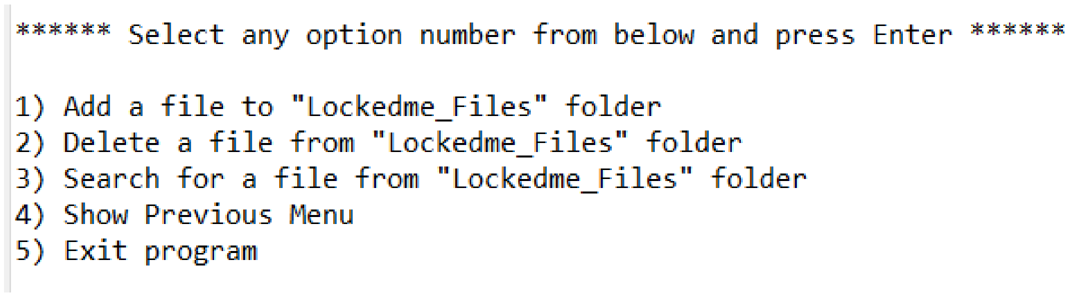
```
****** Select any option number from below and press Enter ******

1) Retrieve all files from "Lockedme_Files" folder
2) Display menu for File operations
3) Exit program
```

Step 3.3: Writing method to display Secondary Menu for File Operations



```java
/**
 * Display Add, Delete, Search etc File Operation Menu Options
 */
public void displayFileOperationMenuOptions() {
    String fileMenu = "\n\n****** Select any option number from below and press Enter ******\n\n"
            + "1) Add a file to "+"\""+ DIR_PATH + "\""+" folder\n" + "2) Delete a file from "+"\""+ DIR_PATH + "\""+" folder\n"
            + "3) Search for a file from "+"\""+ DIR_PATH + "\""+" folder\n" + "4) Show Previous Menu\n" + "5) Exit program\n";

    System.out.println(fileMenu);
}
```

Output:



```
****** Select any option number from below and press Enter ******

1) Add a file to "Lockedme_Files" folder
2) Delete a file from "Lockedme_Files" folder
3) Search for a file from "Lockedme_Files" folder
4) Show Previous Menu
5) Exit program
```

Step 4: Writing a program in Java to handle Menu options selected by user (**MenuOptionHandler.java**)

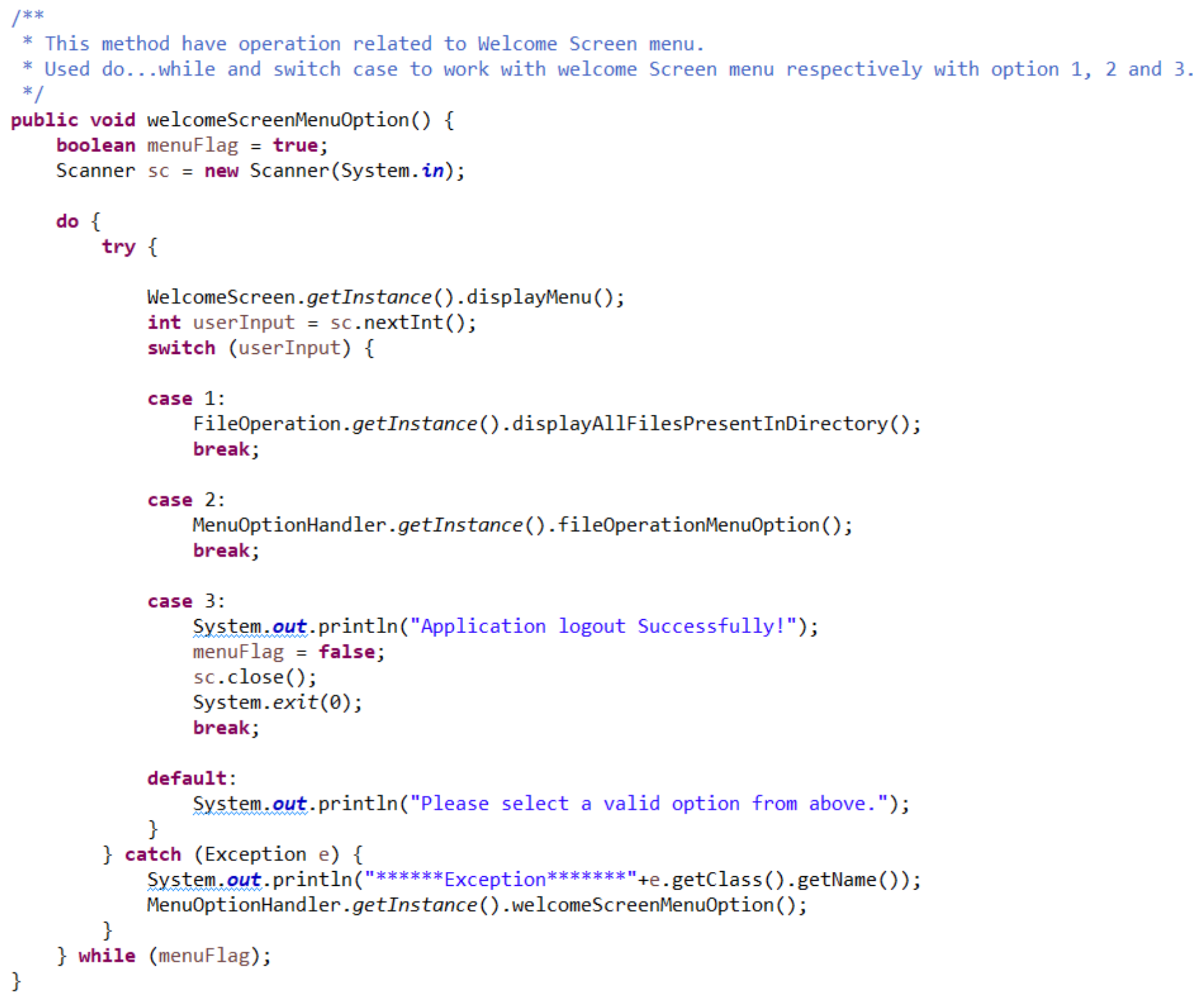
- Select your project and go to File -> New -> Class.
- Enter **MenuOptionHandler** in class name and click on “Finish.”

- **MenuOptionHandler** consists methods for -:

4.1. Handling input selected by user in initial Menu

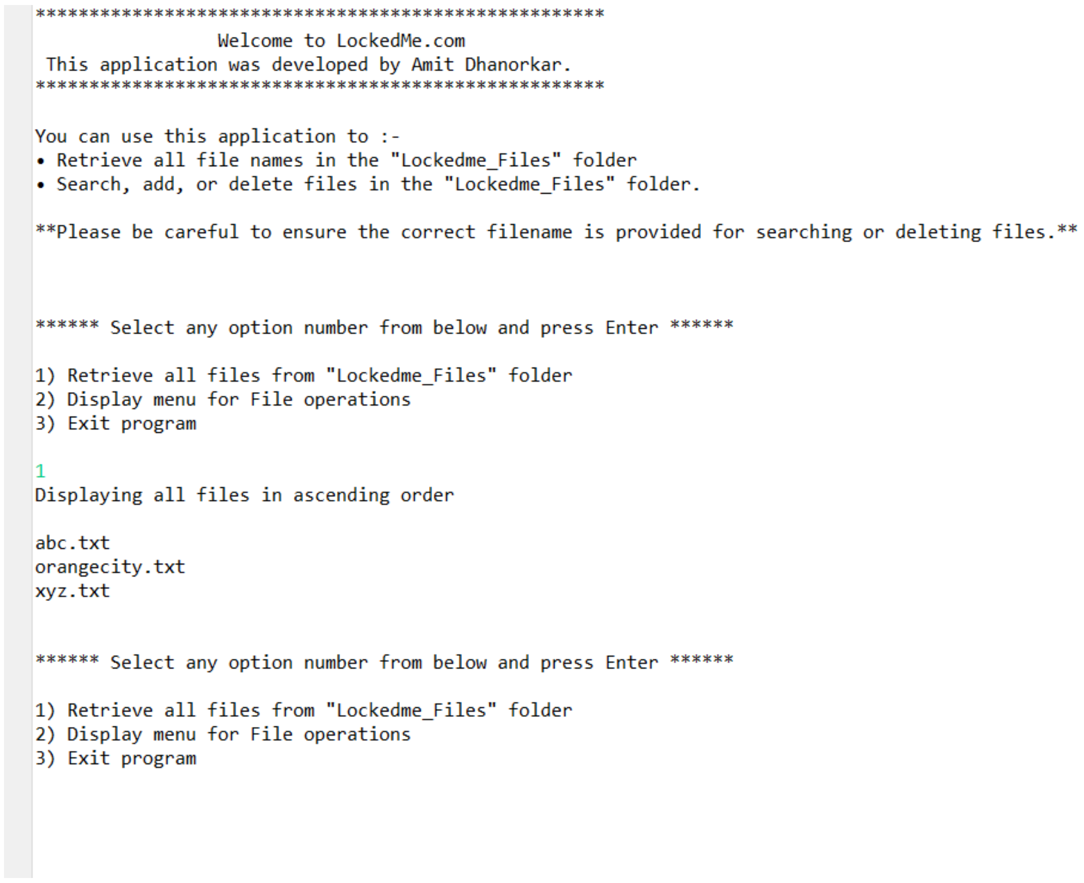
4.2. Handling input selected by user in secondary Menu for File Operations

Step 4.1: Writing method to handle user input in initial Menu



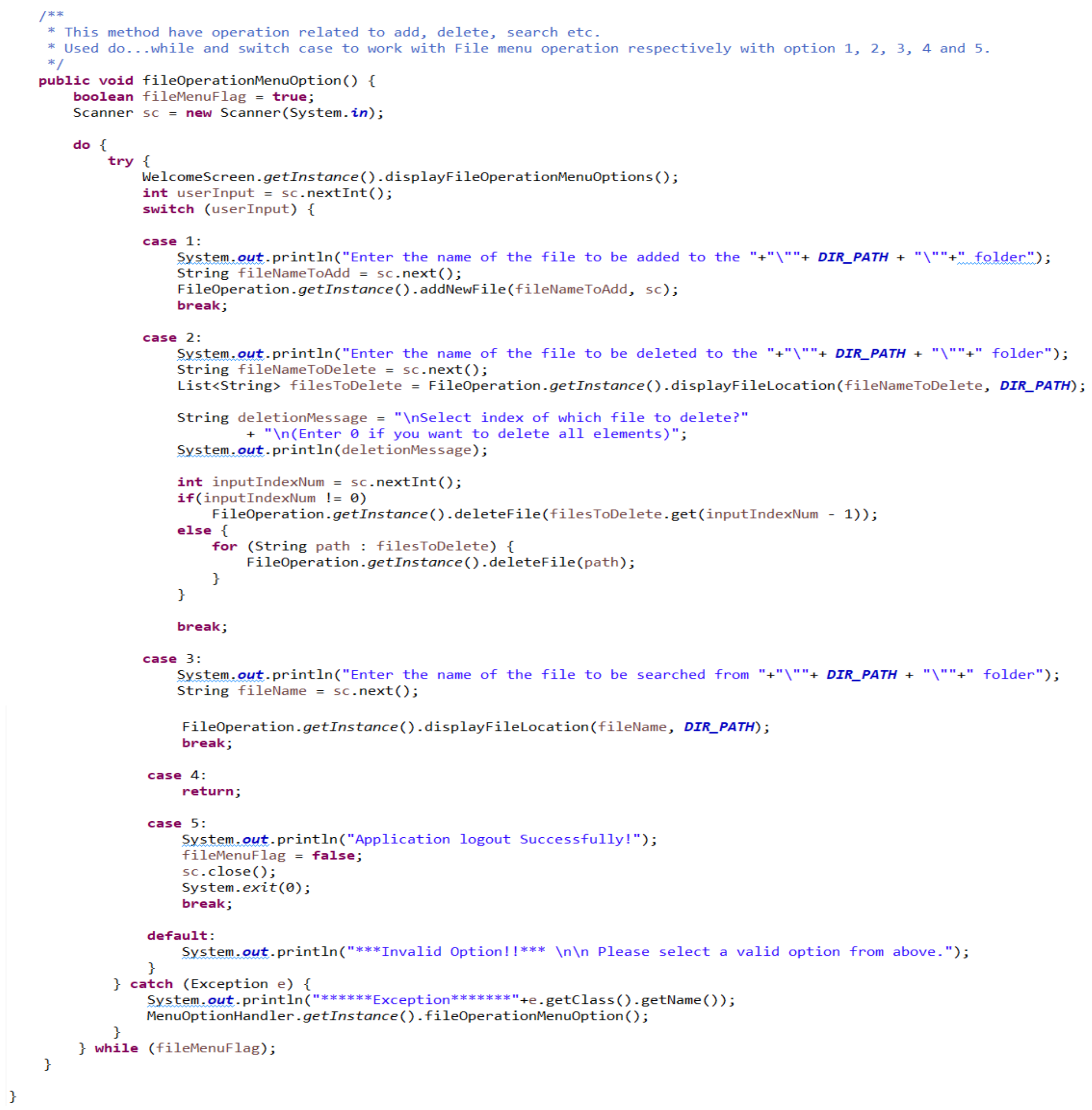
```
/**
 * This method have operation related to Welcome Screen menu.
 * Used do...while and switch case to work with welcome Screen menu respectively with option 1, 2 and 3.
 */
public void welcomeScreenMenuOption() {
    boolean menuFlag = true;
    Scanner sc = new Scanner(System.in);

    do {
        try {

            WelcomeScreen.getInstance().displayMenu();
            int userInput = sc.nextInt();
            switch (userInput) {

            case 1:
                FileOperation.getInstance().displayAllFilesPresentInDirectory();
                break;

            case 2:
                MenuOptionHandler.getInstance().fileOperationMenuOption();
                break;

            case 3:
                System.out.println("Application logout Successfully!");
                menuFlag = false;
                sc.close();
                System.exit(0);
                break;

            default:
                System.out.println("Please select a valid option from above.");
            }
        } catch (Exception e) {
            System.out.println("******Exception*******"+e.getClass().getName());
            MenuOptionHandler.getInstance().welcomeScreenMenuOption();
        }
    } while (menuFlag);
}
```

Output:



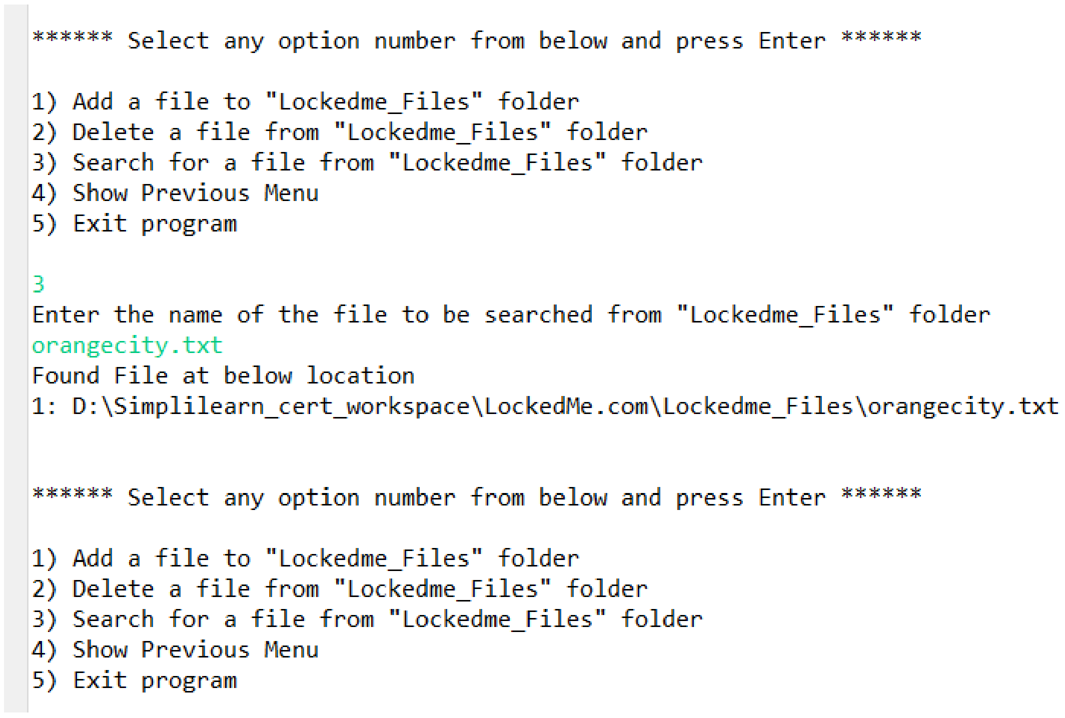
```
*****************************************************
                 Welcome to LockedMe.com
 This application was developed by Amit Dhanorkar.
*****************************************************

You can use this application to :-
• Retrieve all file names in the "Lockedme_Files" folder
• Search, add, or delete files in the "Lockedme_Files" folder.

**Please be careful to ensure the correct filename is provided for searching or deleting files.**



****** Select any option number from below and press Enter ******

1) Retrieve all files from "Lockedme_Files" folder
2) Display menu for File operations
3) Exit program

1
Displaying all files in ascending order

abc.txt
orangecity.txt
xyz.txt


****** Select any option number from below and press Enter ******

1) Retrieve all files from "Lockedme_Files" folder
2) Display menu for File operations
3) Exit program
```

Step 4.2: Writing method to handle user input in Secondary Menu for File Operations



```
    /**
     * This method have operation related to add, delete, search etc.
     * Used do...while and switch case to work with File menu operation respectively with option 1, 2, 3, 4 and 5.
     */
    public void fileOperationMenuOption() {
        boolean fileMenuFlag = true;
        Scanner sc = new Scanner(System.in);

        do {
            try {
                WelcomeScreen.getInstance().displayFileOperationMenuOptions();
                int userInput = sc.nextInt();
                switch (userInput) {

                case 1:
                    System.out.println("Enter the name of the file to be added to the "+"\""+ DIR_PATH + "\""+" folder");
                    String fileNameToAdd = sc.next();
                    FileOperation.getInstance().addNewFile(fileNameToAdd, sc);
                    break;

                case 2:
                    System.out.println("Enter the name of the file to be deleted to the "+"\""+ DIR_PATH + "\""+" folder");
                    String fileNameToDelete = sc.next();
                    List<String> filesToDelete = FileOperation.getInstance().displayFileLocation(fileNameToDelete, DIR_PATH);

                    String deletionMessage = "\nSelect index of which file to delete?"
                            + "\n(Enter 0 if you want to delete all elements)";
                    System.out.println(deletionMessage);

                    int inputIndexNum = sc.nextInt();
                    if(inputIndexNum != 0)
                        FileOperation.getInstance().deleteFile(filesToDelete.get(inputIndexNum - 1));
                    else {
                        for (String path : filesToDelete) {
                            FileOperation.getInstance().deleteFile(path);
                        }
                    }

                    break;

                case 3:
                    System.out.println("Enter the name of the file to be searched from "+"\""+ DIR_PATH + "\""+" folder");
                    String fileName = sc.next();

                    FileOperation.getInstance().displayFileLocation(fileName, DIR_PATH);
                    break;

                case 4:
                    return;

                case 5:
                    System.out.println("Application logout Successfully!");
                    fileMenuFlag = false;
                    sc.close();
                    System.exit(0);
                    break;

                default:
                    System.out.println("***Invalid Option!!*** \n\n Please select a valid option from above.");
                }
            } catch (Exception e) {
                System.out.println("******Exception*******"+e.getClass().getName());
                MenuOptionHandler.getInstance().fileOperationMenuOption();
            }
        } while (fileMenuFlag);
    }

}
```

Output:



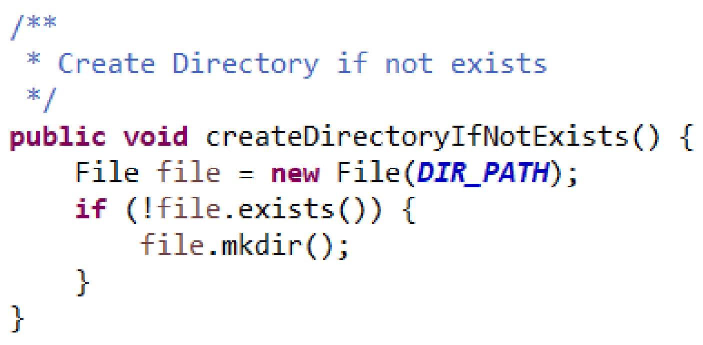
```
****** Select any option number from below and press Enter ******

1) Add a file to "Lockedme_Files" folder
2) Delete a file from "Lockedme_Files" folder
3) Search for a file from "Lockedme_Files" folder
4) Show Previous Menu
5) Exit program

3
Enter the name of the file to be searched from "Lockedme_Files" folder
orangecity.txt
Found File at below location
1: D:\Simplilearn_cert_workspace\LockedMe.com\Lockedme_Files\orangecity.txt


****** Select any option number from below and press Enter ******

1) Add a file to "Lockedme_Files" folder
2) Delete a file from "Lockedme_Files" folder
3) Search for a file from "Lockedme_Files" folder
4) Show Previous Menu
5) Exit program
```

Step 5: Writing a program in Java to perform the File operations as specified by user (**FileOperation.java**)

- Select your project and go to File -> New -> Class.
- Enter **FileOperation** in class name and click on "Finish."
- **FileOperation** consists methods for -:

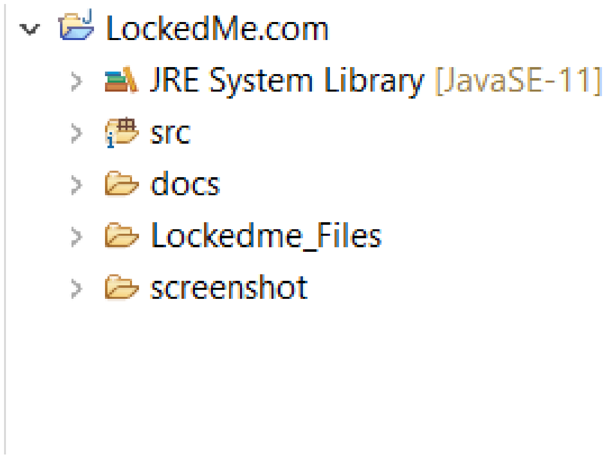
5.1. Creating "LockedMe_Files" folder in project if it's not already present
5.2. Displaying all files in "LockedMe_Files" folder in ascending order
5.3. Creating a file/folder as specified by user input.
5.4. Search files as specified by user input in "LockedMe_Files" folder
5.5. Deleting a file/folder from "LockedMe_files" folder

Step 5.1: Writing method to create "LockedMe_Files" folder in project if it's not present



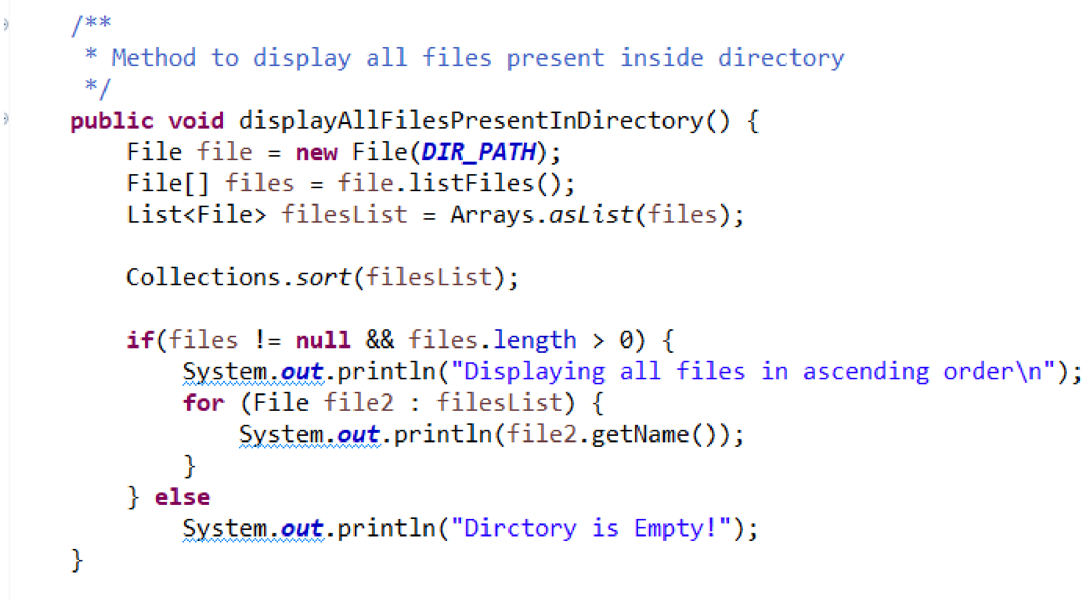
```java
/**
 * Create Directory if not exists
 */
public void createDirectoryIfNotExists() {
    File file = new File(DIR_PATH);
    if (!file.exists()) {
        file.mkdir();
    }
}
```

Output:



LockedMe.com
JRE System Library [JavaSE-11]
src
docs
Lockedme_Files
screenshot

Step 5.2: Writing method to display all files in “LockedMe_Files” folder in ascending order



```java
/**
 * Method to display all files present inside directory
 */
public void displayAllFilesPresentInDirectory() {
    File file = new File(DIR_PATH);
    File[] files = file.listFiles();
    List<File> filesList = Arrays.asList(files);

    Collections.sort(filesList);

    if(files != null && files.length > 0) {
        System.out.println("Displaying all files in ascending order\n");
        for (File file2 : filesList) {
            System.out.println(file2.getName());
        }
    } else
        System.out.println("Dirctory is Empty!");
}
```

Output:

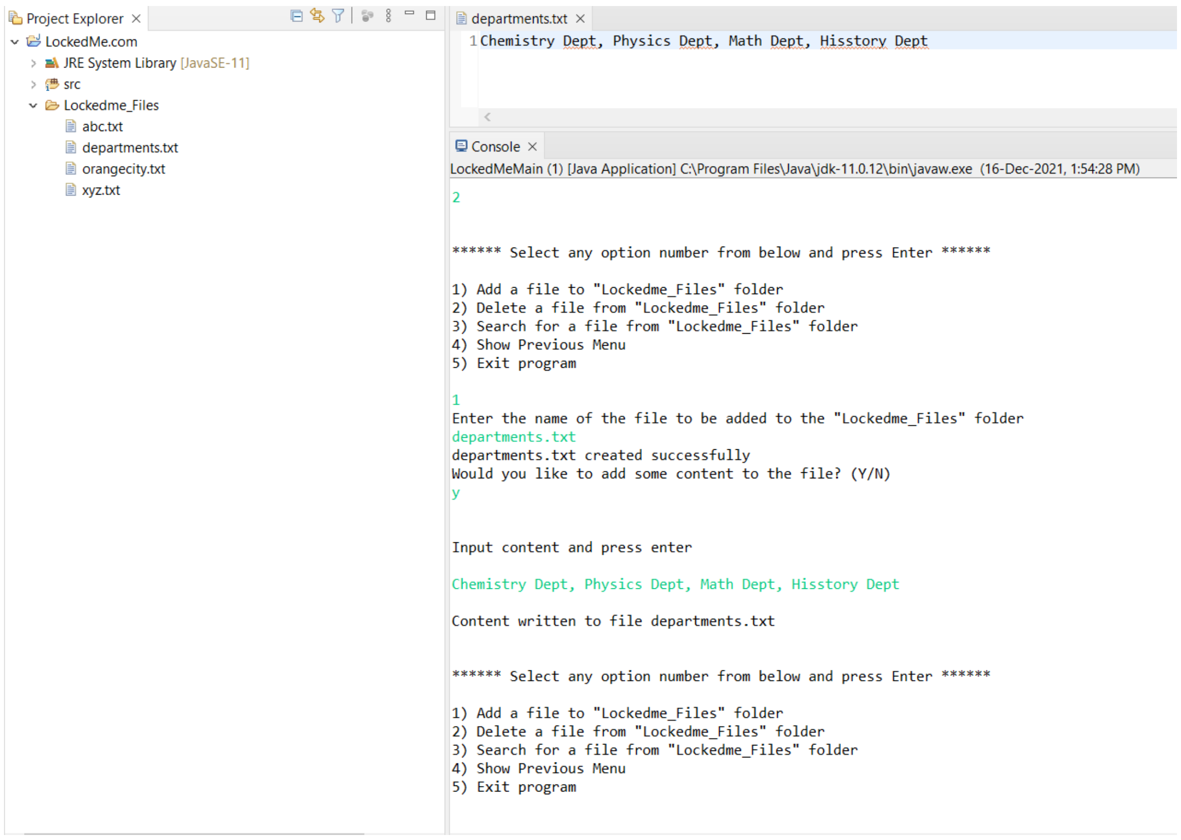
```
****** Select any option number from below and press Enter ******

1) Retrieve all files from "Lockedme_Files" folder
2) Display menu for File operations
3) Exit program

1
Displaying all files in ascending order

departments.txt
orangecity.txt
Simplilearn_university.txt
xyz.txt


****** Select any option number from below and press Enter ******

1) Retrieve all files from "Lockedme_Files" folder
2) Display menu for File operations
3) Exit program
```

Step 5.3: Writing method to create a file/folder as specified by user input.

```java
/**
 * Creates New File named by this abstract pathname if
 * and only if a file with this name does not yet exist.
 *
 * @param fileNameToAdd A fileNameToAdd string
 * @param sc {@code Scanner}
 *
 */
public void addNewFile(String fileNameToAdd, Scanner sc) {
    File f = new File(DIR_PATH, fileNameToAdd);
    FileOutputStream outputStream = null;
    try {
        if(f.createNewFile()) {
            System.out.println(fileNameToAdd + " created successfully");
            System.out.println("Would you like to add some content to the file? (Y/N)");
            String choice = sc.next().toLowerCase();

            sc.nextLine();
            if (choice.equals("y")) {
                System.out.println("\n\nInput content and press enter\n");
                String content = sc.nextLine();
                outputStream = new FileOutputStream(f);
                outputStream.write(content.getBytes());
                System.out.println("\nContent written to file " + fileNameToAdd);
            }
        } else {
            System.out.println("File already exist in directory with this name.");
        }
    } catch (IOException e) {
        System.out.println("Failed to create file " + fileNameToAdd);
        e.printStackTrace();
    }
}
```

Output:

Folders are automatically created along with file



```
Chemistry Dept, Physics Dept, Math Dept, Hisstory Dept
```

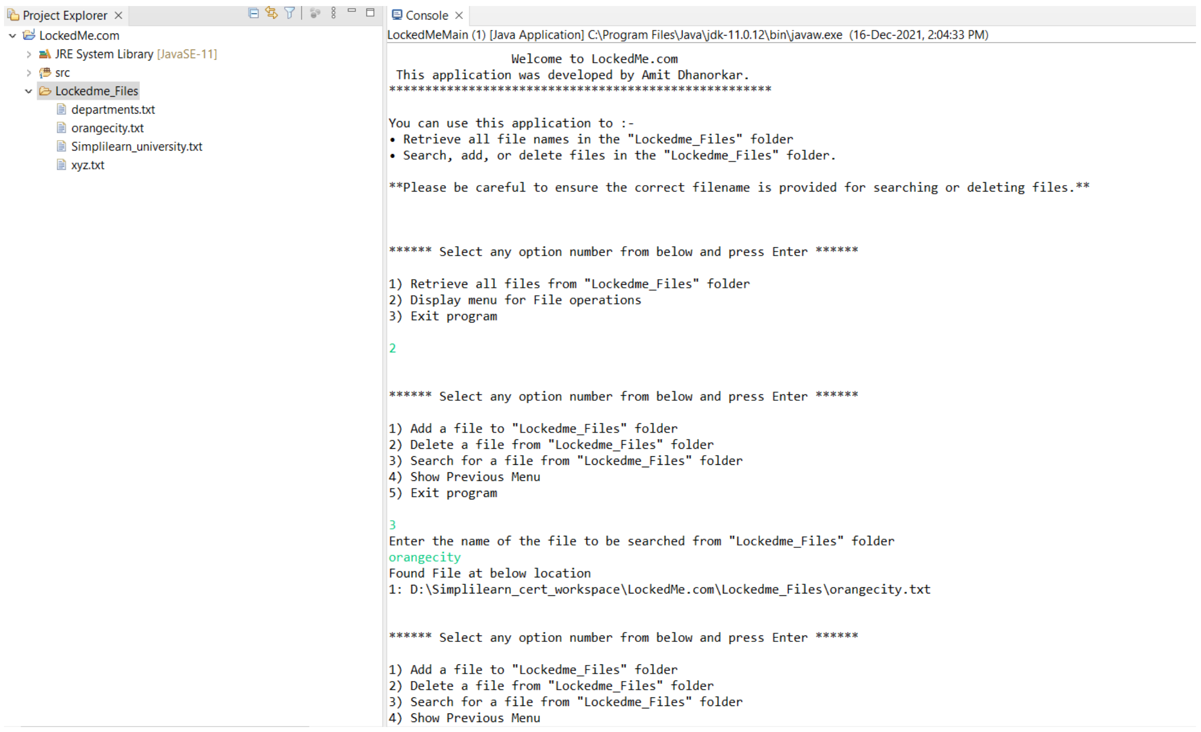
```
LockedMeMain (1) [Java Application] C:\Program Files\Java\jdk-11.0.12\bin\javaw.exe (16-Dec-2021, 1:54:28 PM)
2


****** Select any option number from below and press Enter ******

1) Add a file to "Lockedme_Files" folder
2) Delete a file from "Lockedme_Files" folder
3) Search for a file from "Lockedme_Files" folder
4) Show Previous Menu
5) Exit program

1
Enter the name of the file to be added to the "Lockedme_Files" folder
departments.txt
departments.txt created successfully
Would you like to add some content to the file? (Y/N)
y


Input content and press enter

Chemistry Dept, Physics Dept, Math Dept, Hisstory Dept

Content written to file departments.txt


****** Select any option number from below and press Enter ******

1) Add a file to "Lockedme_Files" folder
2) Delete a file from "Lockedme_Files" folder
3) Search for a file from "Lockedme_Files" folder
4) Show Previous Menu
5) Exit program
```

Step 5.4: Writing method to search for all files as specified by user input in “LockedMe_Files” folder

```
    /**
     * Display files location of present files in directory. If
     * file is not available in directory then print appropriate message.
     *
     * @param fileName A fileName to display its location
     * @param directoryPath A directoryPath where files placed
     * @return listOfFileNames A listOfFileNames present in directoryPath
     * @throws Exception
     */
    public List<String> displayFileLocation(String fileName, String directoryPath) throws Exception {
        List<String> listOfFileNames = new ArrayList<>();
        FileOperation.getInstance().searchFileInDirectory(fileName, directoryPath, listOfFileNames);

        if(listOfFileNames.isEmpty())
            System.out.println("\n\n***** Could not find any file with given file name as \"" + fileName +
                    "\" *****\n Please check the file name is correct.");
        else {
            System.out.println("Found File at below location");

            List<String> files = IntStream.range(0, listOfFileNames.size())
                    .mapToObj(index -> (index + 1) + ": " + listOfFileNames.get(index)).collect(Collectors.toList());

            files.forEach(System.out::println);
        }

        return listOfFileNames;
    }

    /**
     * Search the file or directory denoted by this pathname in created directory.
     *
     * @param fileName A fileName to search
     * @param directoryPath A directoryPath where files placed
     * @param listOfFileNames A listOfFileNames present in directoryPath
     * @throws Exception
     *
     */
    private void searchFileInDirectory(String fileName, String directoryPath, List<String> listOfFileNames) throws Exception {
        File dir = new File(directoryPath);
        File[] files = dir.listFiles();
        List<File> filesList = Arrays.asList(files);

         if (files != null && files.length > 0) {
             for (File file : filesList) {

                 if (file.getName().startsWith(fileName)) {
                     listOfFileNames.add(file.getAbsolutePath());
                 }

                 if (file.isDirectory()) {
                     searchFileInDirectory(file.getAbsolutePath(), fileName, listOfFileNames);
                 }
             }
         }

     }
```

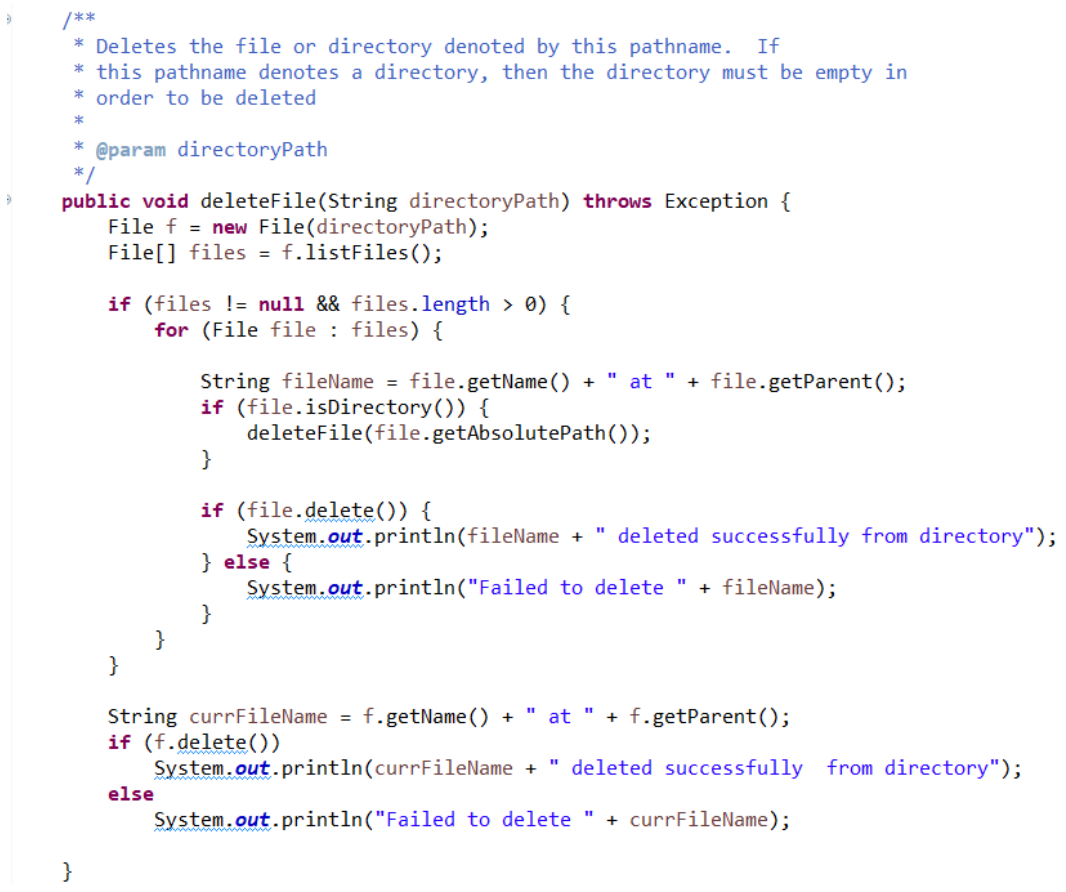
Output:

All files starting with the user input are displayed along with index



```
LockedMeMain (1) [Java Application] C:\Program Files\Java\jdk-11.0.12\bin\javaw.exe  (16-Dec-2021, 2:04:33 PM)
                 Welcome to LockedMe.com
 This application was developed by Amit Dhanorkar.
*****************************************************

You can use this application to :-
• Retrieve all file names in the "Lockedme_Files" folder
• Search, add, or delete files in the "Lockedme_Files" folder.

**Please be careful to ensure the correct filename is provided for searching or deleting files.**


****** Select any option number from below and press Enter ******

1) Retrieve all files from "Lockedme_Files" folder
2) Display menu for File operations
3) Exit program

2


****** Select any option number from below and press Enter ******

1) Add a file to "Lockedme_Files" folder
2) Delete a file from "Lockedme_Files" folder
3) Search for a file from "Lockedme_Files" folder
4) Show Previous Menu
5) Exit program

3
Enter the name of the file to be searched from "Lockedme_Files" folder
orangecity
Found File at below location
1: D:\Simplilearn_cert_workspace\LockedMe.com\Lockedme_Files\orangecity.txt


****** Select any option number from below and press Enter ******

1) Add a file to "Lockedme_Files" folder
2) Delete a file from "Lockedme_Files" folder
3) Search for a file from "Lockedme_Files" folder
4) Show Previous Menu
```

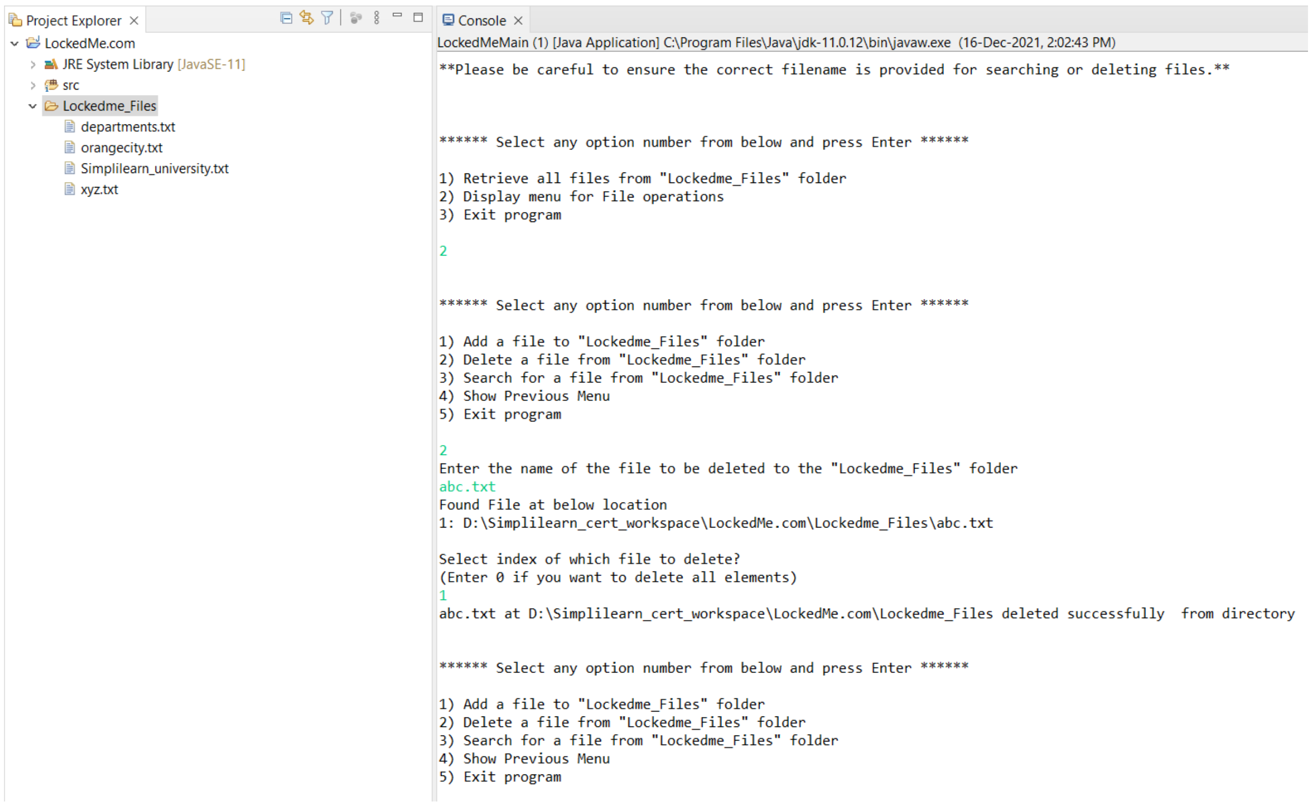
Step 5.5: Writing method to delete file/folder specified by user input in “LockedMe_Files” folder and prompts user to specify which index to delete. If folder selected, all its files will be deleted. If user wants to delete all the files specified after the search, they can input value 0.



```java
/**
 * Deletes the file or directory denoted by this pathname.  If
 * this pathname denotes a directory, then the directory must be empty in
 * order to be deleted
 *
 * @param directoryPath
 */
public void deleteFile(String directoryPath) throws Exception {
    File f = new File(directoryPath);
    File[] files = f.listFiles();

    if (files != null && files.length > 0) {
        for (File file : files) {

            String fileName = file.getName() + " at " + file.getParent();
            if (file.isDirectory()) {
                deleteFile(file.getAbsolutePath());
            }

            if (file.delete()) {
                System.out.println(fileName + " deleted successfully from directory");
            } else {
                System.out.println("Failed to delete " + fileName);
            }
        }
    }

    String currFileName = f.getName() + " at " + f.getParent();
    if (f.delete())
        System.out.println(currFileName + " deleted successfully  from directory");
    else
        System.out.println("Failed to delete " + currFileName);

}
```

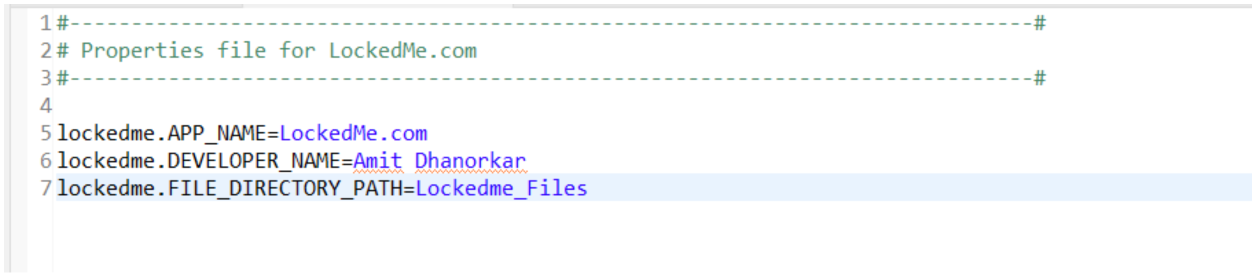
Output:

To verify if file is deleted on Eclipse, right click on Project and click “Refresh”.



```
LockedMeMain (1) [Java Application] C:\Program Files\Java\jdk-11.0.12\bin\javaw.exe  (16-Dec-2021, 2:02:43 PM)
**Please be careful to ensure the correct filename is provided for searching or deleting files.**


****** Select any option number from below and press Enter ******

1) Retrieve all files from "Lockedme_Files" folder
2) Display menu for File operations
3) Exit program

2


****** Select any option number from below and press Enter ******

1) Add a file to "Lockedme_Files" folder
2) Delete a file from "Lockedme_Files" folder
3) Search for a file from "Lockedme_Files" folder
4) Show Previous Menu
5) Exit program

2
Enter the name of the file to be deleted to the "Lockedme_Files" folder
abc.txt
Found File at below location
1: D:\Simplilearn_cert_workspace\LockedMe.com\Lockedme_Files\abc.txt

Select index of which file to delete?
(Enter 0 if you want to delete all elements)
1
abc.txt at D:\Simplilearn_cert_workspace\LockedMe.com\Lockedme_Files deleted successfully  from directory


****** Select any option number from below and press Enter ******

1) Add a file to "Lockedme_Files" folder
2) Delete a file from "Lockedme_Files" folder
3) Search for a file from "Lockedme_Files" folder
4) Show Previous Menu
5) Exit program
```

Step 6: Added properties file (application.properties)



```
#---------------------------------------------------------------------------------#
# Properties file for LockedMe.com
#---------------------------------------------------------------------------------#

lockedme.APP_NAME=LockedMe.com
lockedme.DEVELOPER_NAME=Amit Dhanorkar
lockedme.FILE_DIRECTORY_PATH=Lockedme_Files
```

Step 7: Pushing the code to GitHub repository

- Open your command prompt and navigate to the folder where you have created your files.

  cd <folder path>

- Initialize repository using the following command:

  git init

- Add all the files to your git repository using the following command:

  git add .

- Commit the changes using the following command:

  git commit . -m <commit message>

- Push the files to the folder you initially created using the following command:

  git push -u origin master

Unique Selling Points of the Application

1. The application is designed to keep on running and taking user inputs even after exceptions occur. To terminate the application, appropriate option needs to be selected.

2. The application can take file name as input.

3. User is also provided the option to write content if they want into the newly created file.

4. The application doesn't restrict user to specify the exact filename to search/delete file. They can specify the starting input, and the program searches all files starting with the value and displays it. The user is then provided the option to select all files or to select a specific index to delete.

5. The user is able to seamlessly switch between options or return to previous menu even after any required operation like adding, searching, deleting or retrieving of files is performed.

6. The application is designed with modularity in mind. Even if one wants to update the path, they can change it through the application.properties file. Application has been developed keeping in mind that there should be very less “hardcoding” of data.

Conclusions

Further enhancements to the application can be made which may include:

- Conditions to check if user is allowed to delete the file or add the file at the specific locations.
- Retrieving files by different criteria like Last Modified, Type, etc.
- Allowing user to append data to the file.