**LockedMe.com – Virtual Key for Repositories**

This document contains sections for:

* [Sprint planning and Task completion](#Sprint_plan)
* [Core concepts used in project](#Core_concepts)
* [Flow of the Application](#Flow).
* [Demonstrating the product capabilities, appearance, and user interactions.](#Product_capability)
* [Unique Selling Points of the Application](#USP)
* [Conclusions](#Conclusions)

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

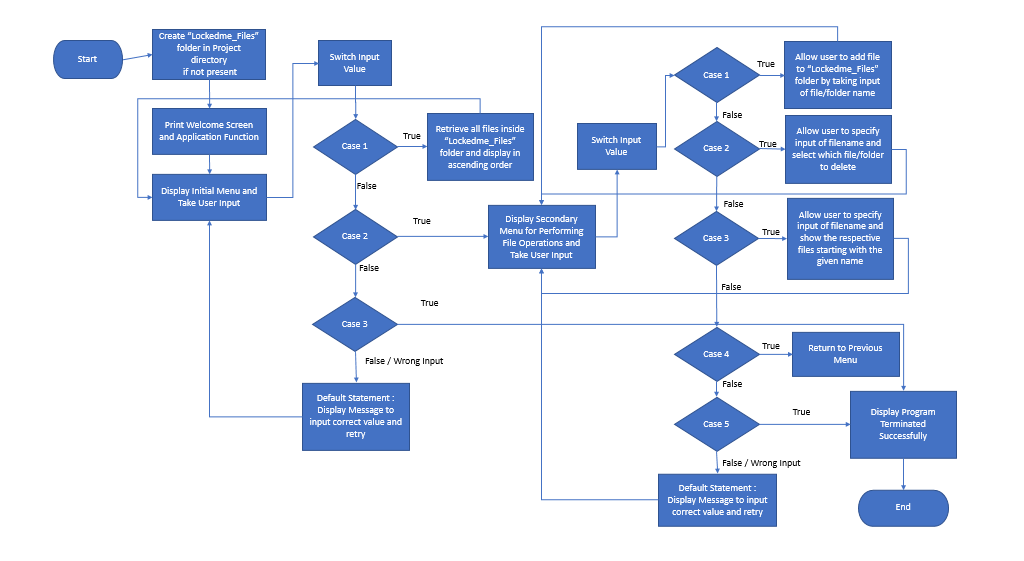
The project is planned to be completed in 1 sprint. Tasks assumed to be completed in the sprint are:

* 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



## Demonstrating the product capabilities, appearance, and user interactions

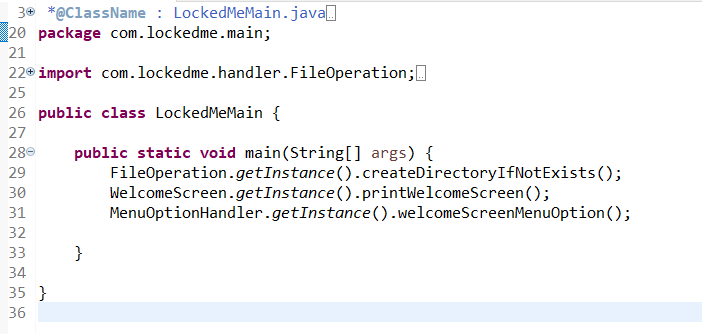
To demonstrate the product capabilities, below are the sub-sections configured to highlight appearance and user interactions for the project:

1. [Creating the project in Eclipse](#Step_1)
2. [Writing a program in Java for the entry point of the application (**LockedMeMain.java**)](#Step_2)
3. [Writing a program in Java to display Menu options available for the user (**WelcomeScreen.java**)](#Step_3)
4. [Writing a program in Java to handle Menu options selected by user (**MenuOptionHandler.java**)](#Step_4)
5. [Writing a program in Java to perform the File operations as specified by user (**FileOperation.java**)](#Step_5)
6. [Added properties file (application.properties)](#_toc290)
7. [Pushing the code to GitHub repository](#_toc291)

## **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**)



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

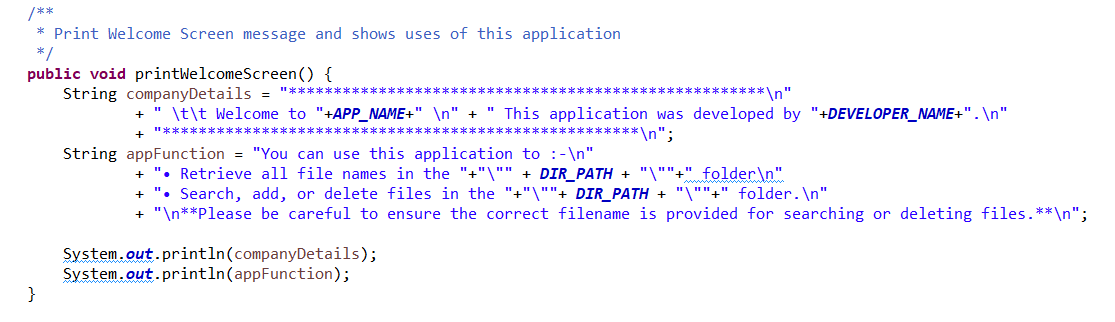
* Select your project and go to File -> New -> Class.
* Enter **WelcomeScreen** in class name and click on “Finish.”
* **WelcomeScreen** consists methods for -:

**3.1** [Displaying Welcome Screen](#Step_3_1)

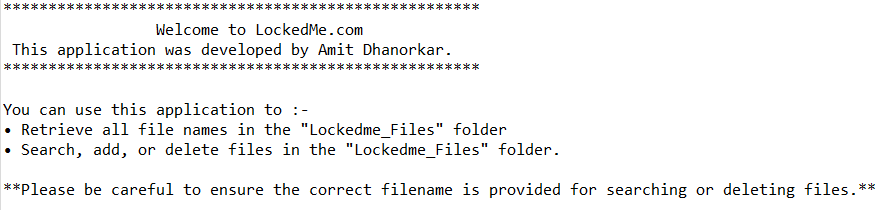
**3.2** [Displaying Initial Menu](#Step_3_2)

**3.3** [Displaying Secondary Menu for File Operations available](#Step_3_3)

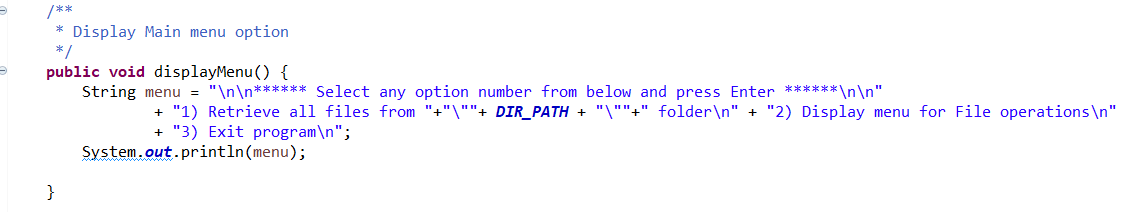
**Step 3.1:** Writing method to display Welcome Screen



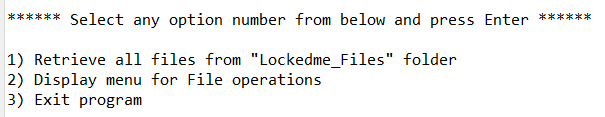
**Output:**



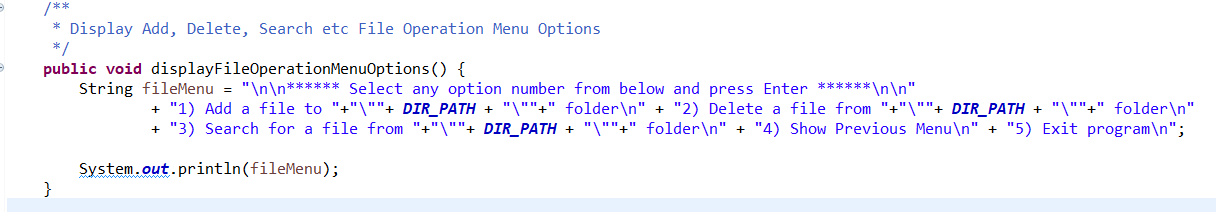
**Step 3.2:** Writing method to display Initial Menu



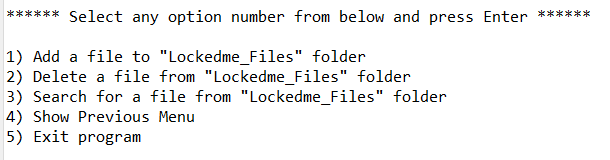
**Output:**



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



**Output:**



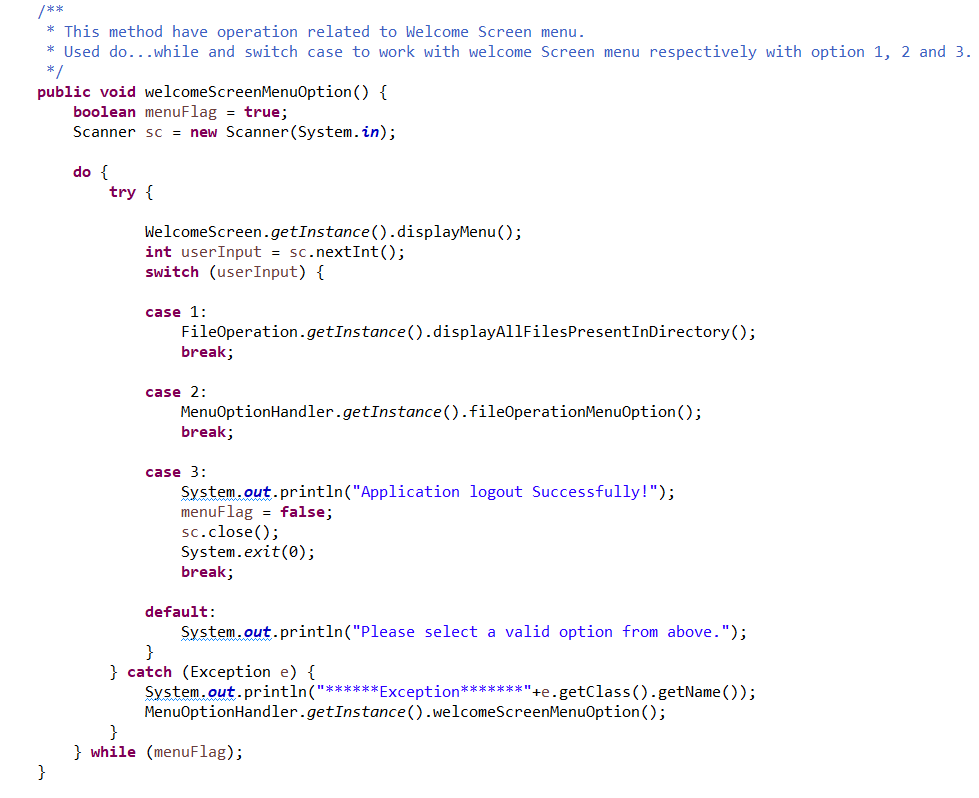
## **Step 4:** Writing a program in Java to handle Menu options selected by user (**MenuOptionHandler.java**)

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

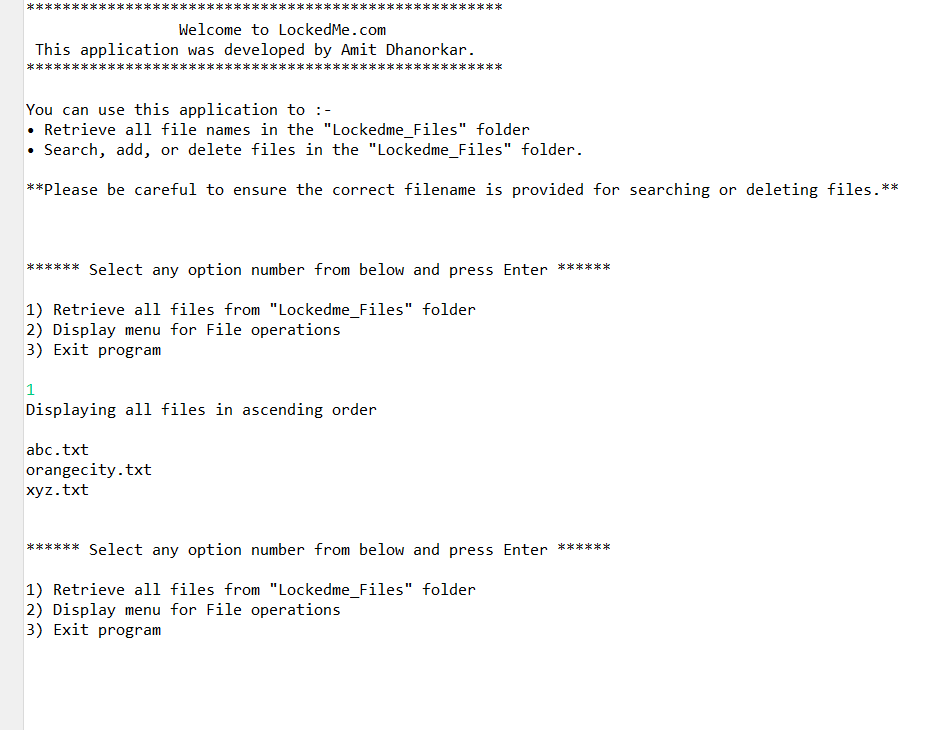
**4.1.**[Handling input selected by user in initial Menu](#Step_4_1)

**4.2.**[Handling input selected by user in secondary Menu for File Operations](#Step_4_2)

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



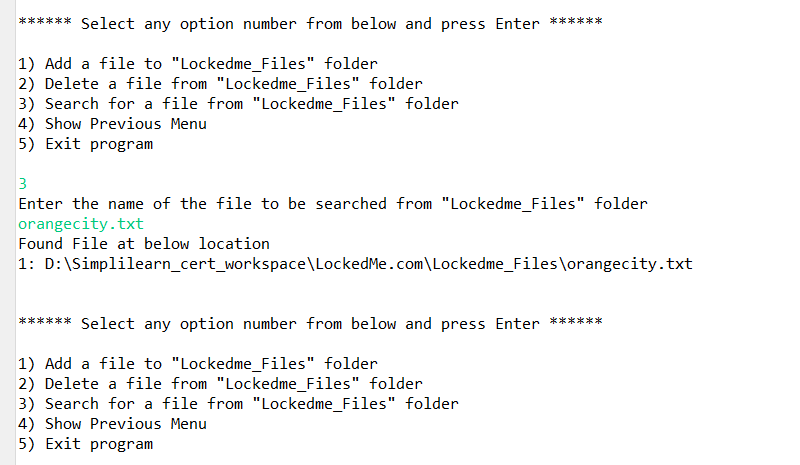
**Output:**



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



**Output:**



## **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](bookmark://Step_5_1)

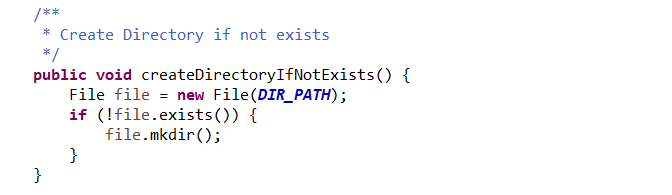
**5.2.**[Displaying all files in “LockedMe\_Files” folder in ascending order](bookmark://Step_5_2)

**5.3.**[Creating a file/folder as specified by user input.](#Step_5_3)

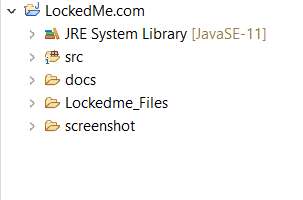
**5.4.**[Search files as specified by user input in “LockedMe\_Files” folder](bookmark://Step_5_4)

**5.5.**[Deleting a file/folder from “LockedMe\_files” folder](bookmark://Step_5_5)

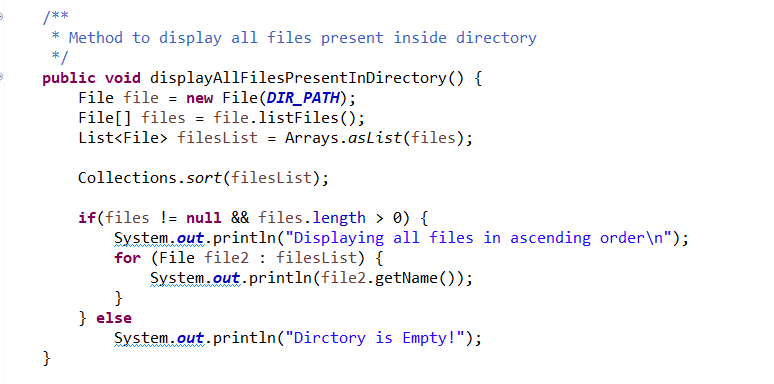
**Step 5.1:** Writing method to create “LockedMe\_Files” folder in project if it’s not present



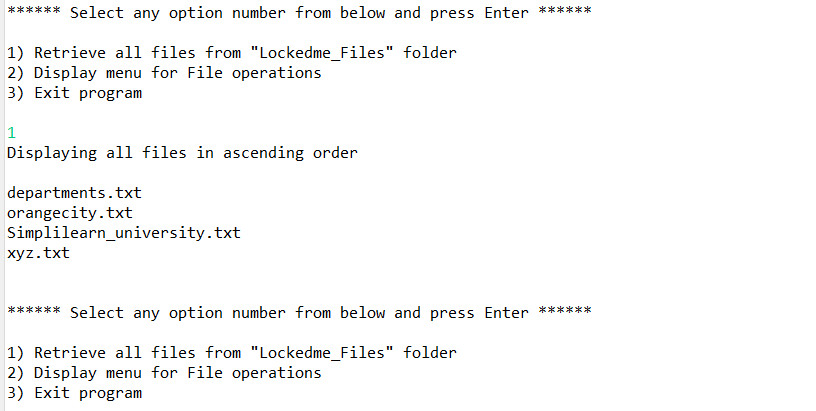
**Output:**



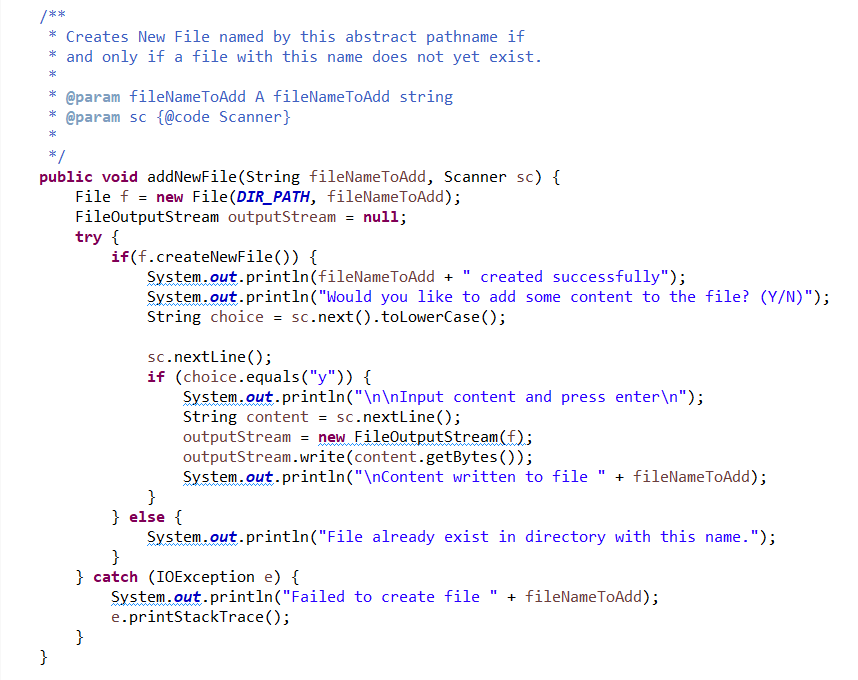
**Step 5.2:** Writing method to display all files in “LockedMe\_Files” folder in ascending order



**Output:**

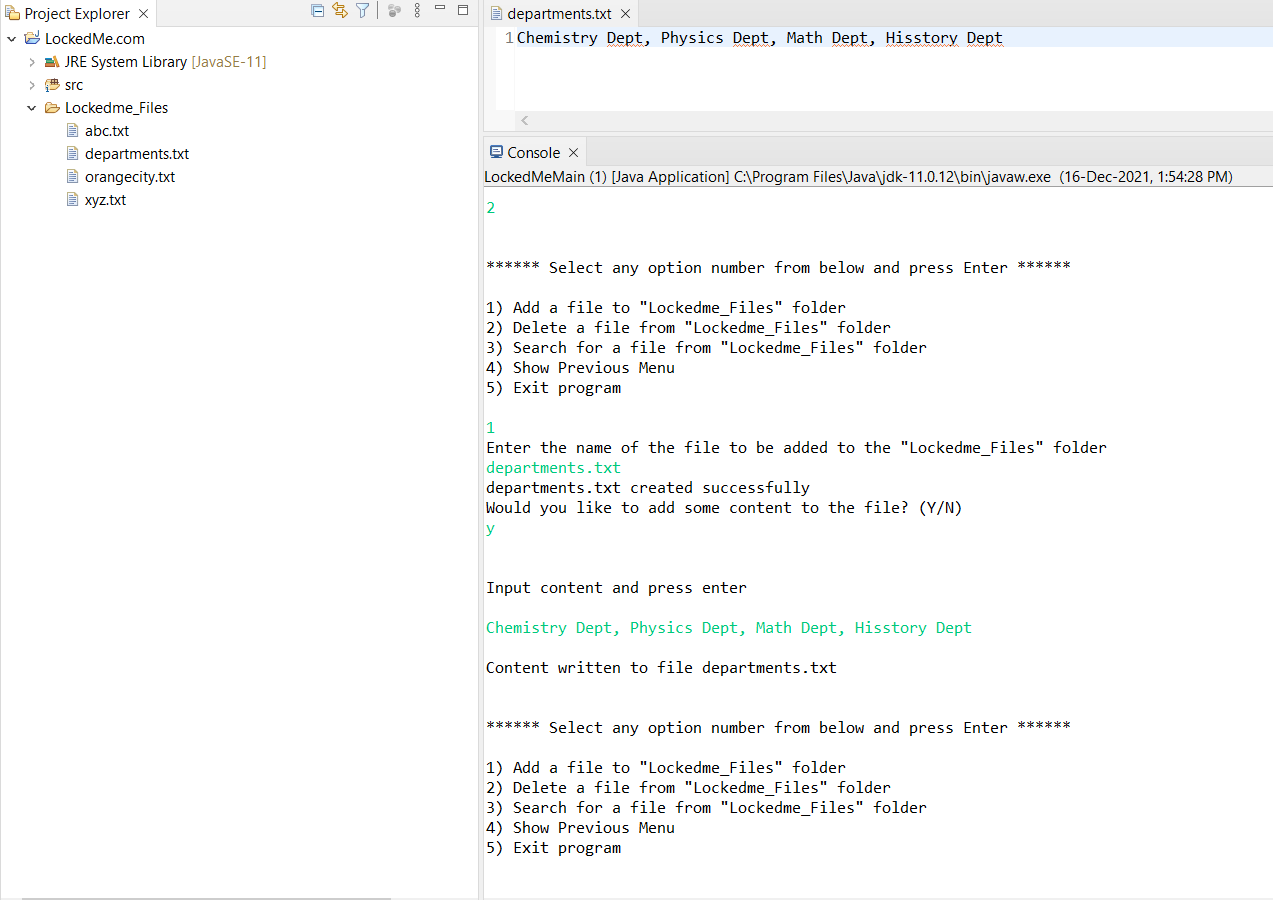


**Step 5.3:** Writing method to create a file/folder as specified by user input.

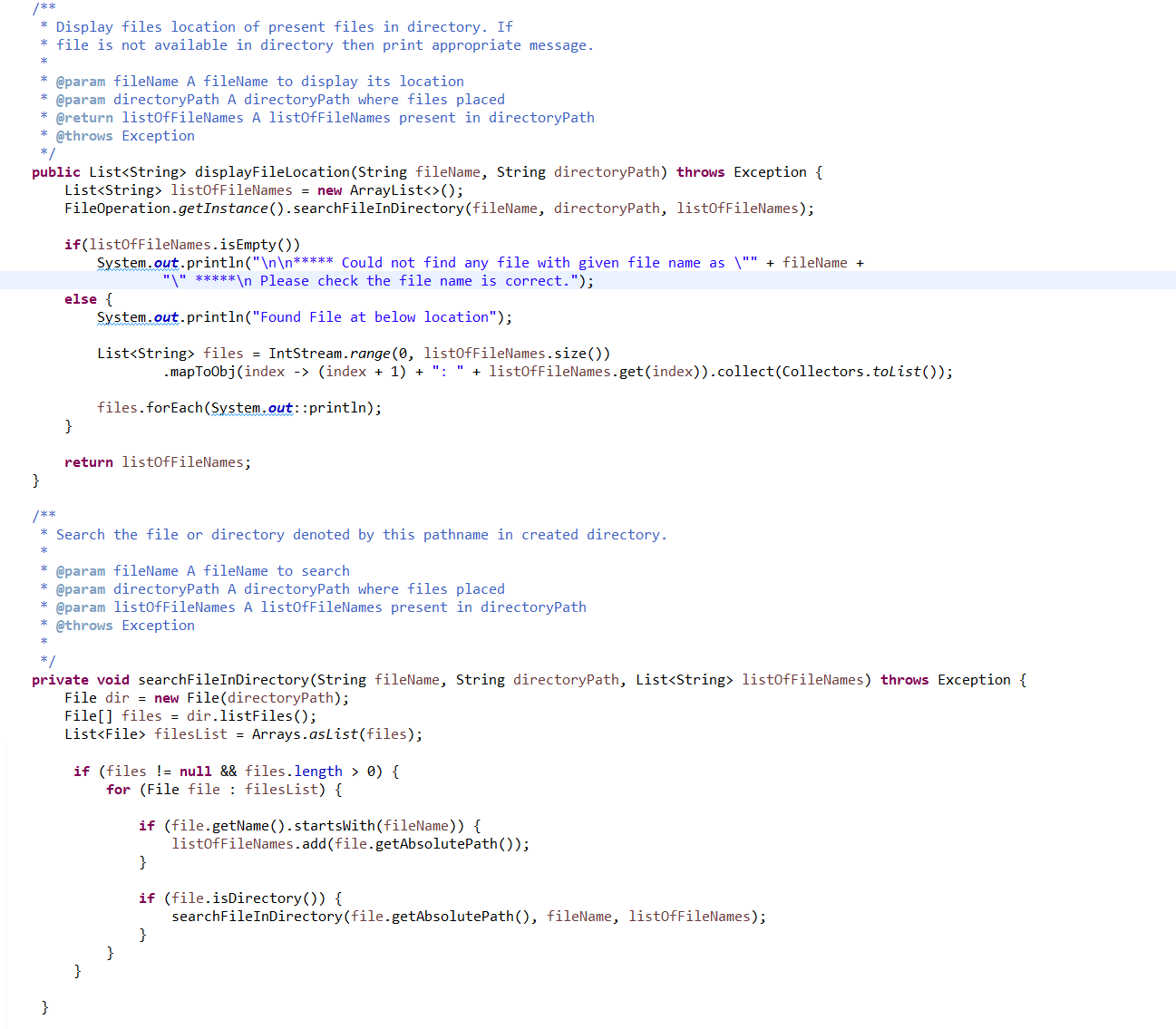


**Output:**

**Folders are automatically created along with file**

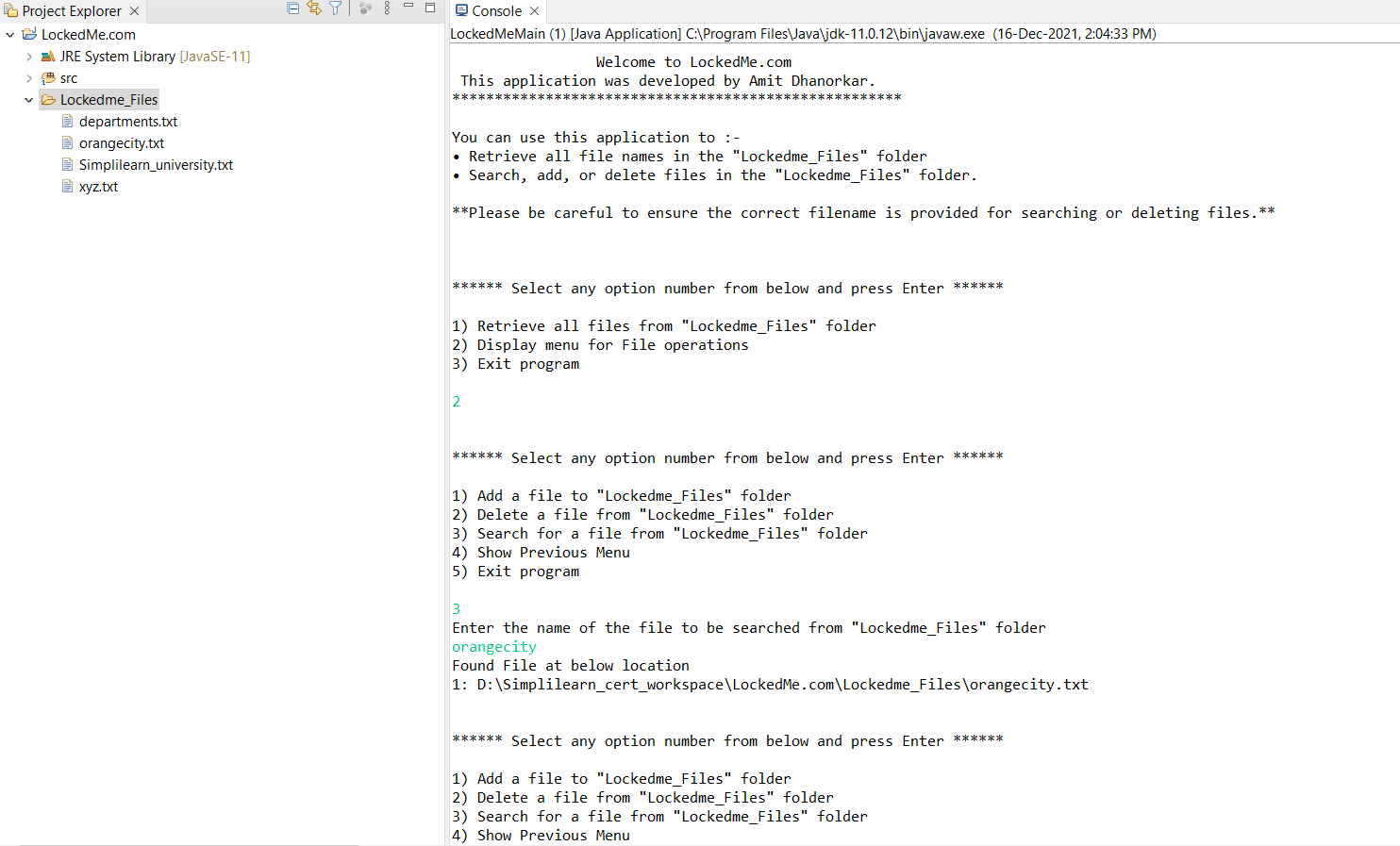


**Step 5.4:**  Writing method to search for all files as specified by user input in “LockedMe\_Files” folder

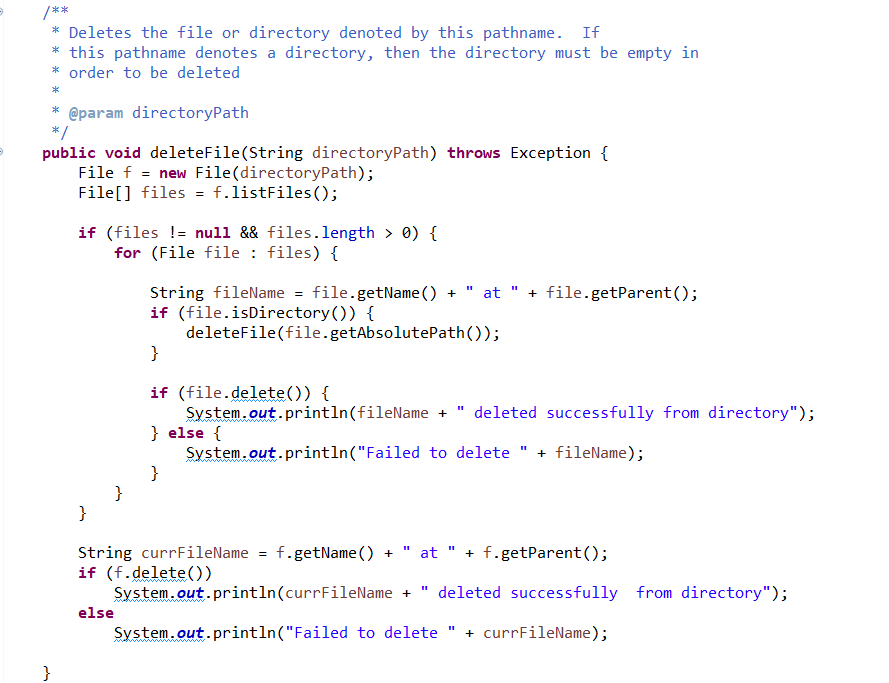


**Output:**

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

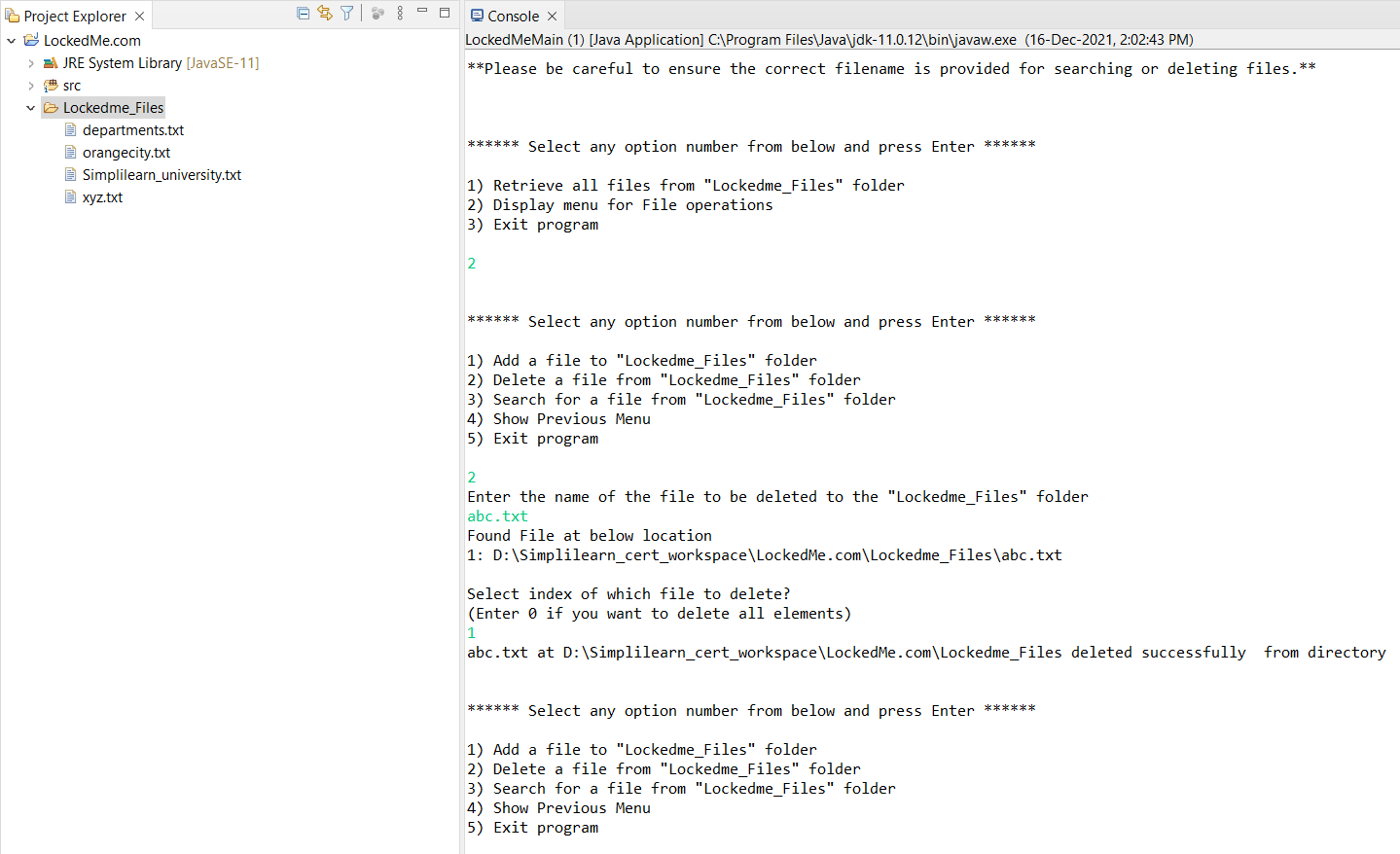


**Step 5.5:**  Writing method to delete file/folder specified by user input in “LockedMe\_Files” folder and prompts user to specify which index to delete. If folder selected, all its files will be deleted. If user wants to delete all the files specified after the search, they can input value 0.



**Output:**

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



## **Step 6:** Added properties file (application.properties)

## **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.