**Lockers Pvt Ltd Project – Specification Document**

This document contains sections for:

* [Sprint planning and Task completion](#Sprint_plan)
* [Core concepts used in project](#Core_concepts)
* [Flow of the](#Flow) Project
* [Demonstrating the project code and its output.](#Product_capability)
* [Unique Selling Points of the Application](#USP)

The code for this project is located at https://github.com/\*\*\*\*\*\*\*\*\*\*

The project is developed by Santhi mogili.

## **Sprints planning and Task completion**

The project is planned to be complete the project in 2 sprints.

Tasks to be completed in the sprint are:

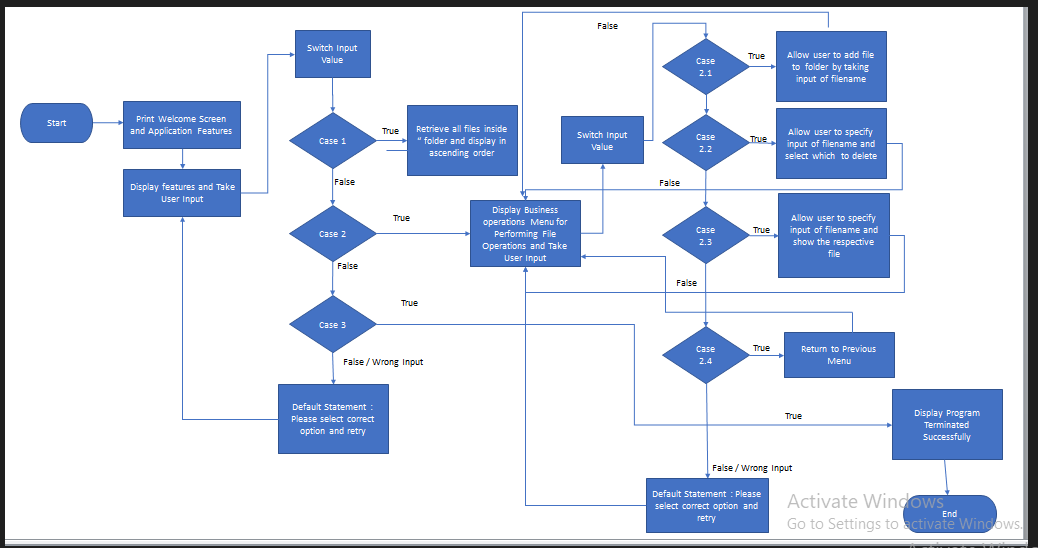
* Creating the flow of the application.
* Writing the Java program to fulfill the requirements of the project.
* Testing the Java program with different kinds of User input.
* Pushing code to GitHub.
* Creating this specification document to show project code and output of project.

## **Core concepts used in project**

File Handling, Sorting, Flow Control, Exception Handling, ArrayList, Collections, File Streams, Regex, Methods, Objects etc.

## 

## **Flow of the Application**



## **Demonstrating the project code and output of the code.**

To demonstrate the project code, below are the sub-sections configured to highlight output of the project:

1. [**Creating the project, package and class in Eclipse**](#Step_1)
2. **Writing a program in Java for Lockers Pvt Ltd app .**
3. [**Pushing the code to GitHub repository**](#Step_6)

## **Step 1: Creating a new project, package and class in Eclipse**

* Open Eclipse
* Go to File -> New -> Project -> Java Project -> Next.
* Type in **Simplilearn Phase-1** Project in project name and click on “Finish.”
* Select your project and go to File -> New -> Package.
* Enter **com.simplilearn.phase1.project** in Package name and click on “Finish.”
* Select your package and go to File -> New ->Class
* Enter **LockersPvtLtd\_Project** in Class name and click on “Finish.”

## **Step 2: Writing a program in Java for Lockers Pvt Ltd app .**

package com.simplilearn.phase1.project;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Scanner;

**Step 2.1 : Declaring the variables and the path in the class so that all the methods can access .**

public class LockersPvtLtd\_Project {

Scanner scanner = new Scanner(System.in);

String fileName;

String path = "D:\\Simplilearn LockersPvtLtd Project\\File\\";

**Step 2.2: Creating a boolean method to check if the main folder in the declared path is existed or not. If not existed create a folder with the path mentioned above.**

LockersPvtLtd\_Project(boolean b) throws IOException

{

//Checking Folder Exist or not. If not exists folder will be created

File file = new File (path);

if (file.mkdirs())

{

System.out.println ("Folder is created" +path);

}

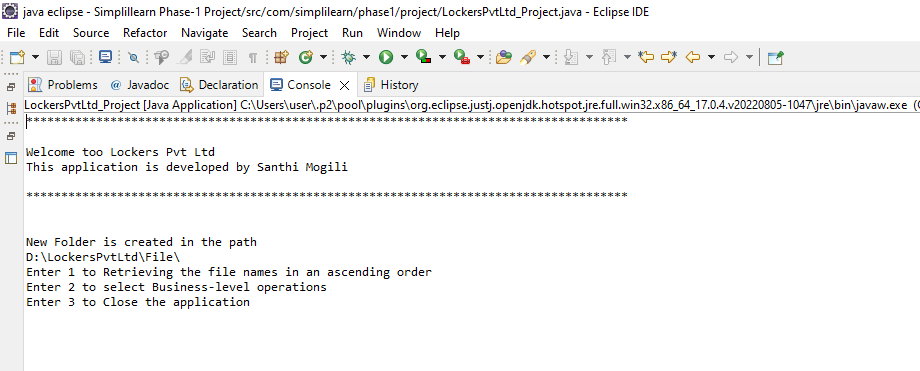
else {

System.out.println ("Folder already exists");

}

}

**Output:**

****

**Step 2.3: Creating the methods for features() to do the operations such as**

**2.3.1. Retrieve the filenames in ascending order method.**

**2.3.2. Business Level Operations**

**2.3.2.1. Add a file method.**

**2.3.2.2. Delete a file method. Business Level Operations Features**

**2.3.2.3 Search a file method.**

**2.3.3. Close the application.**

//2.3.1--To Retrieve all files in ascending order

public void displayFile() throws IOException

{

File file = new File(path);

File filenames[]=file.listFiles();

for(File ff:filenames) {

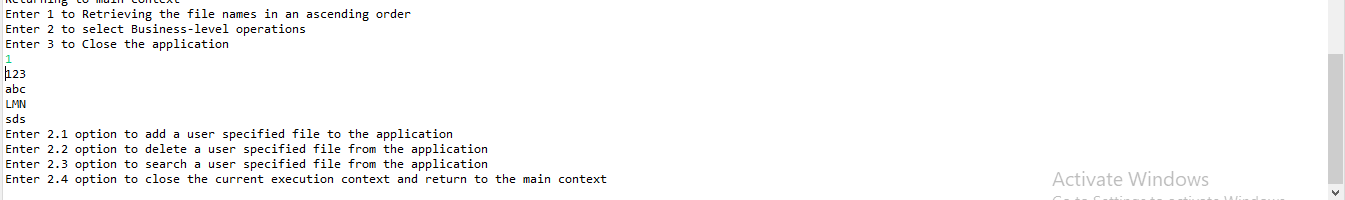
System.out.println(ff.getName());

}

selectBusinessOperation();

}

**Output:**

****

//2.3.2.1--To add a user specified file to the application

**public** **void** addFile() **throws** IOException

{

System.***out***.println("Please enter the File name to Create\n");

fileName = scanner.next();

File fi = **new** File(path+fileName);

**boolean** b=fi.createNewFile();

**if**(b!=**true**)

{

System.***out***.println("Filename Already Found \n");

}

**else**

{

System.***out***.println("File Added Successfully\n");

}

//To insert and append the content into the file created.

System.***out***.println("Would you like to add some content in the file or append some content in the already existing file?\nPress(Y/N)\n");

String choice=scanner.next().toLowerCase();

**if**(choice.equals("y")) {

FileWriter fos = **new** FileWriter(fi,**true**);

String s= scanner.next()+scanner.nextLine();

fos.write(s);

System.***out***.println("Content written to file\n");

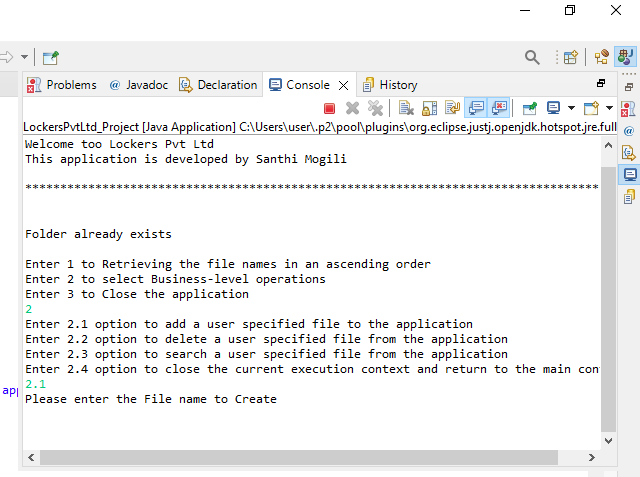
fos.close();

}

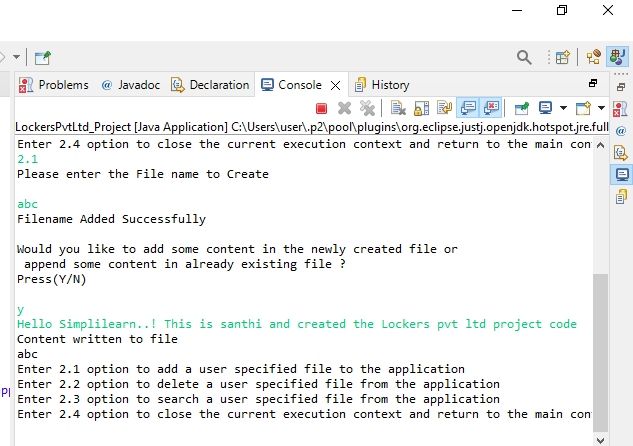
selectBusinessOperation();

}

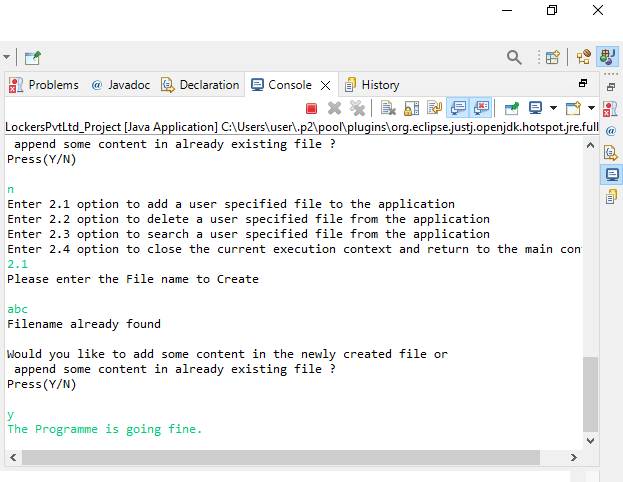
**Output 1: Add a file option output**

****

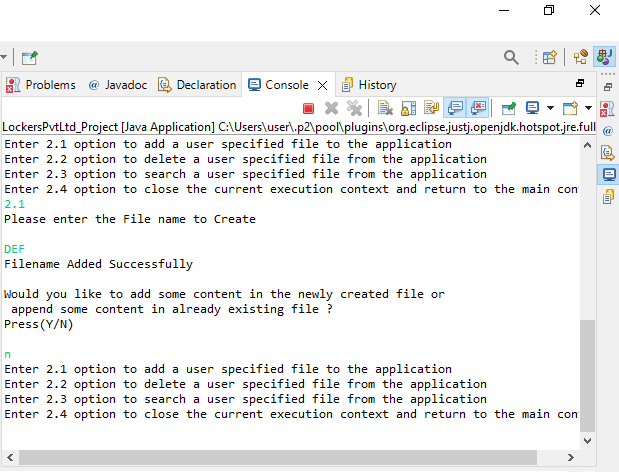
**Output 2: File added with content output**



**Output 3: File recalled again to append the content output**

****

**Output 4: File added without content output**



//2.3.2.2--To delete a user specified file from the application

public void deleteFile() throws IOException

{

System.out.println("Please enter the File name to Delete\n");

fileName = scanner.next();

File fi = new File(path+fileName);

boolean b=fi.delete();

if(b==true)

{

System.out.println("File Deleted Successfully");

}

else

{

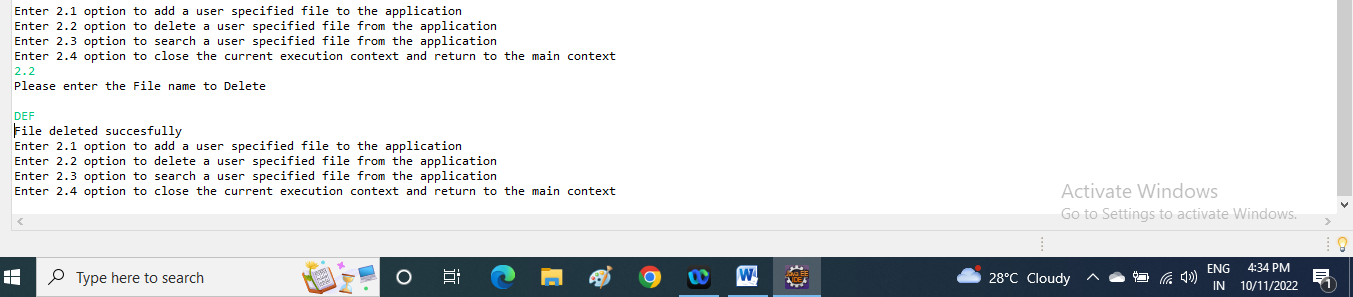
System.out.println("File Not Exsist to Delete");

}

selectBusinessOperation();

}

**Output : To delete a file output**

****

//2.3.3.3--To search a user specified file from the application

public void searchFile() throws IOException

{

System.out.println("Please enter the File name to Search\n");

//adding case sensitive to search a specified existed file

fileName = scanner.next();

File file = new File(path);

File filenames[]=file.listFiles();

int flag=0;

for(File ff : filenames)

{

if(ff.getName().equals(fileName))

{

flag=1;

break;

}

else {

flag=0;

}

}

if(flag==1)

{

System.out.println(fileName +" File Found!! in this path \n " +path);

}

else

{

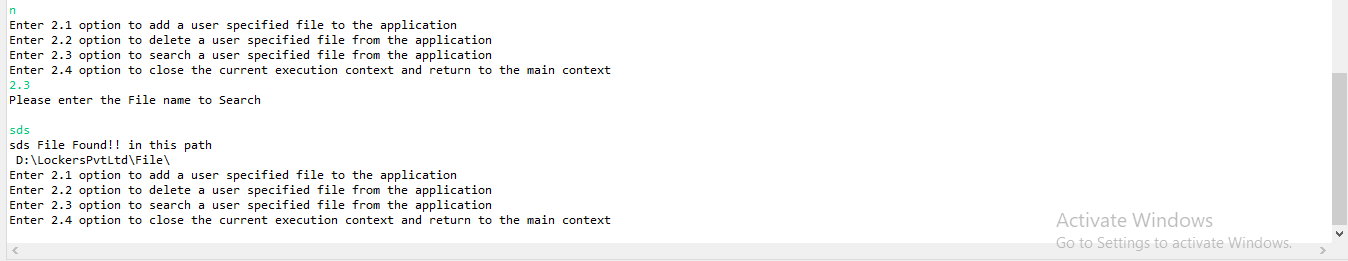
System.out.println("File Not Found");

}

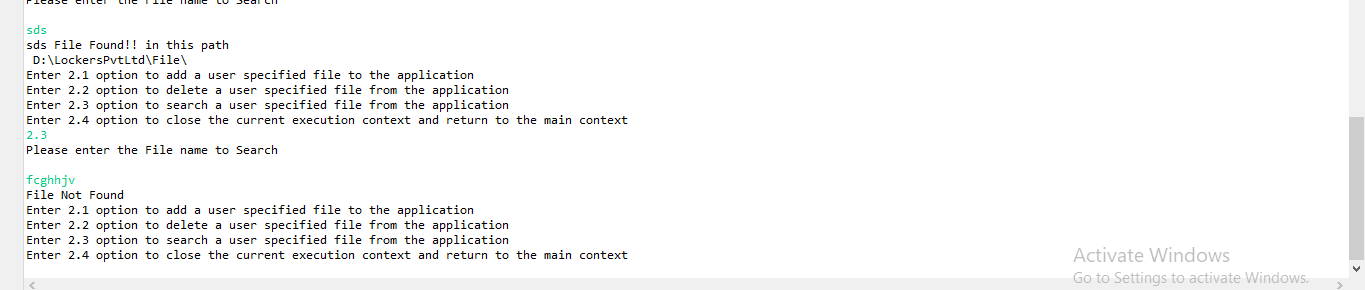
selectBusinessOperation();

}

**Output 1: To search a file output**

****

**Output 2: To search a non existent file output**

****

//2.3.3--To Close the Application

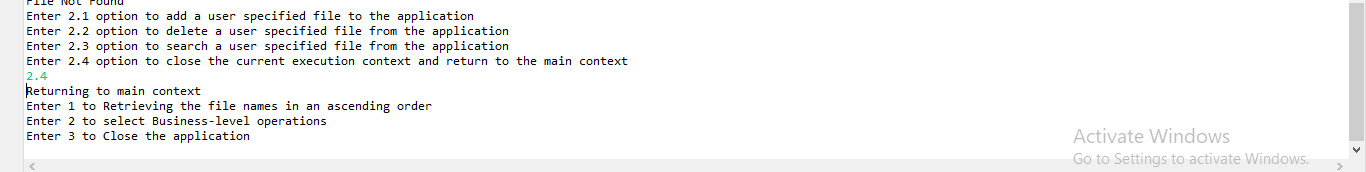
public void closeApplication() throws IOException

{

System.out.println("Closed the Application");

System.exit(0);

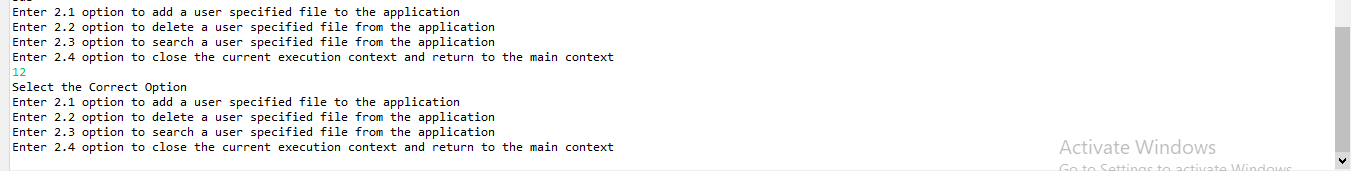
**Output 1: Return to the main context output**

****

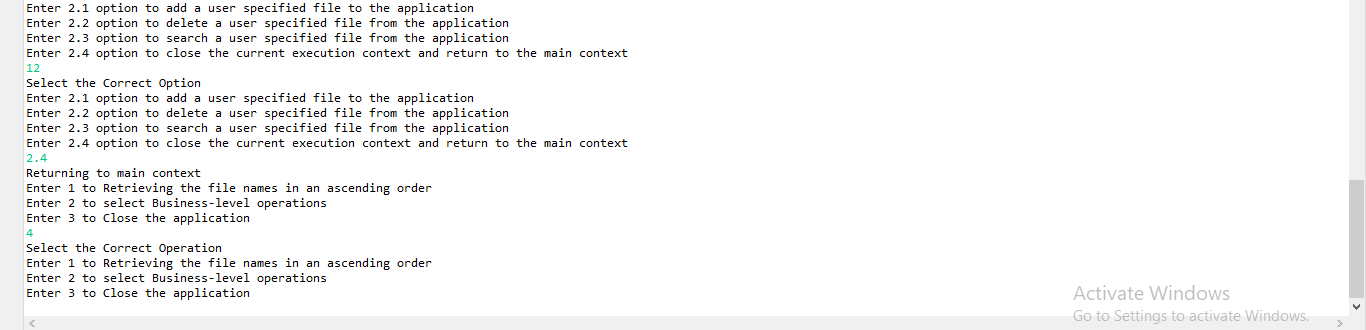
**Output 2: To close the application output**

****

**Invalid Output 1 :**

****

**Invalid Output 2 :**

****

}

**Step 1.3.2: Creating the Business Level Operations method for the user to select options to add, delete ,search files in the folder and return main path.**

//2.3.2.This is the Business Operation Method

public void selectBusinessOperation() throws IOException

{

System.out.println("Enter 2.1 option to add a user specified file to the application");

System.out.println("Enter 2.2 option to delete a user specified file from the application");

System.out.println("Enter 2.3 option to search a user specified file from the application");

System.out.println("Enter 2.4 option to close the current execution context and return to the main context");

String option = scanner.next();

switch (option) {

case "2.1":

addFile();

break;

case "2.2":

deleteFile();

break;

case "2.3":

searchFile();

break;

case "2.4":

features();

break;

default:

System.out.println("Select the Correct Option");

break;

}

selectBusinessOperation();

}

**Step 2.3.1: Creating the features method for the user to do the operations in the app.**

//2.3.1.This is Generic features and 3 operations ----> main method

public void features() throws IOException

{

System.out.println("Enter 1 to Retrieving the file names in an ascending order");

System.out.println("Enter 2 to select Business-level operations");

System.out.println("Enter 3 to Close the application");

String operation = scanner.next();

switch (operation) {

case "1":

displayFile();

break;

case "2":

selectBusinessOperation();

break;

case "3":

closeApplication();

break;

default:

System.out.println("Select the Correct Operation");

break;

}

features();

}

**Step 2.4 : To display application name and developer details and calling the features of applications.**

public static void main(String[] args) throws IOException {

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

System.out.println("\*\*Welcome too Lockers Pvt Ltd.com\*\*\n"+"This application is developed by Santhi Mogili\n");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

System.out.println();

//calling the class to get features of app

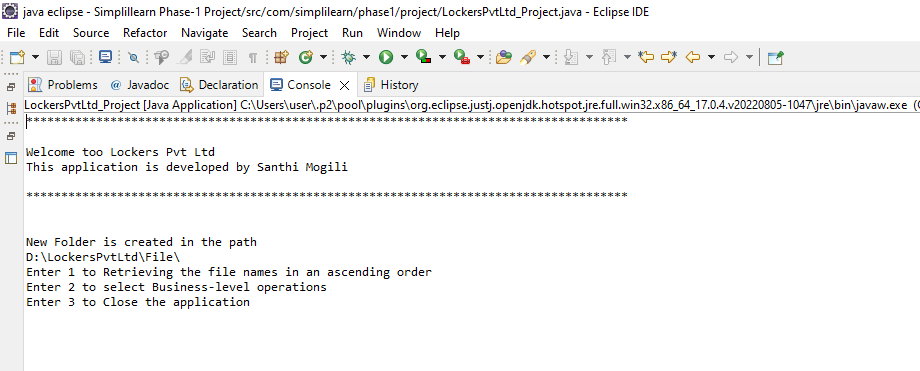
LockersPvtLtd\_Project lockersPvtLtd = new LockersPvtLtd\_Project(true);

lockersPvtLtd.features();

}

}

**Display Ouput:**

****

## **Pushing the code to GitHub repository**

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

**cd <** **D:\LockersPvtLtd\File>**

* Initialize repository using the following command:

**git init**

* Remote add the repository using the following command

**https://github.com/Mogilisanthi/Simplilearn-Phase\_1-Project.git**

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

**git add .**

* Commit the changes using the following command:

**git commit . -m “lockers pvt ltd project”**

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

**git push -u origin main**

## **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. User is also provided the option to write content if they want to insert content into the newly created file and can append the content in the existing file by calling with same filename in output in add file method.
3. The application also allows user to delete and search the files with respect to filename.
4. 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.
5. Even if one wants to update the path, they can change it through the source code. Application has been developed keeping in mind that there should be very less “hardcoding” of data.