Write - up
Lockedme.com application

Developed by- Yoonus 11th August 2021

➔ Table of contents

Aim

Develop a prototype of the lockedme.com application . The prototype must contain the business operations advised by the stakeholders. The application must be developed within 15 working days

Modules in the application

- ➔ Retrieve the file name in the ascending order
- ➔ Option to add a user specified file into the directory
- ➔ Option to delete a user specified file from the directory
- ➔ Option to search a user specified file in the directory
- ➔ Navigation option to stop the current execution context and return to the main menu

Sprint

The project concluded in two sprints .
Modules implemented in the first sprint
- ➔ Retrieve the file name in the ascending order
- ➔ Option to add a user specified file into the directory

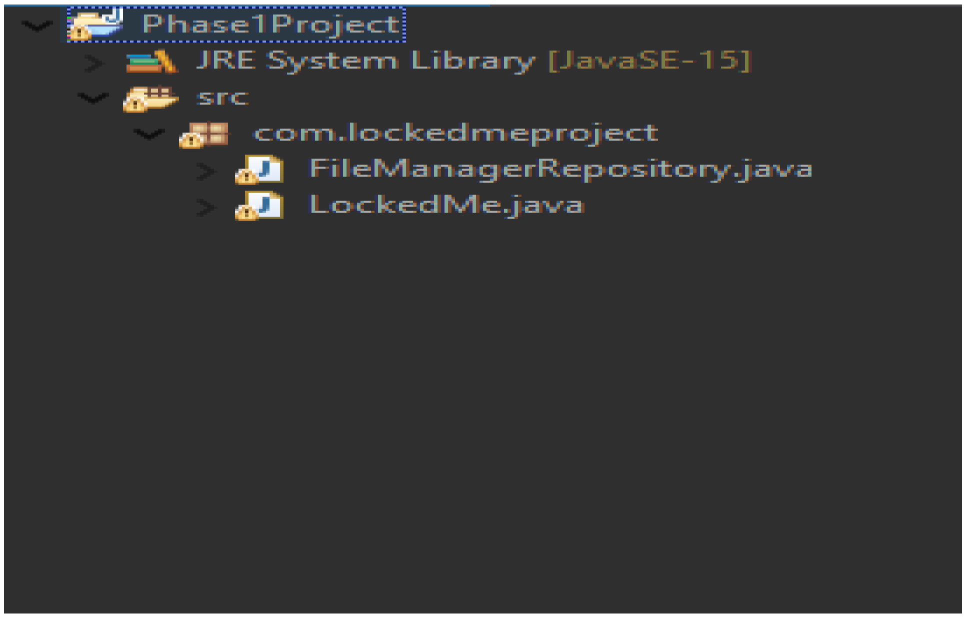
Modules implemented in the first sprint
- ➔ Option to delete a user specified file from the directory
- ➔ Option to search a user specified file in the directory
- ➔ Navigation option to stop the current execution context and return to the main menu

Github link of the source code

https://github.com/yoonus86/Lockedme.com-java-project.git

Source code of the application

A new java project file was created in eclipse IDE . The project is named as “Phase1Project”.


```
Phase1Project
  JRE System Library [JavaSE-15]
  src
    com.lockedmeproject
      FileManagerRepository.java
      LockedMe.java
```

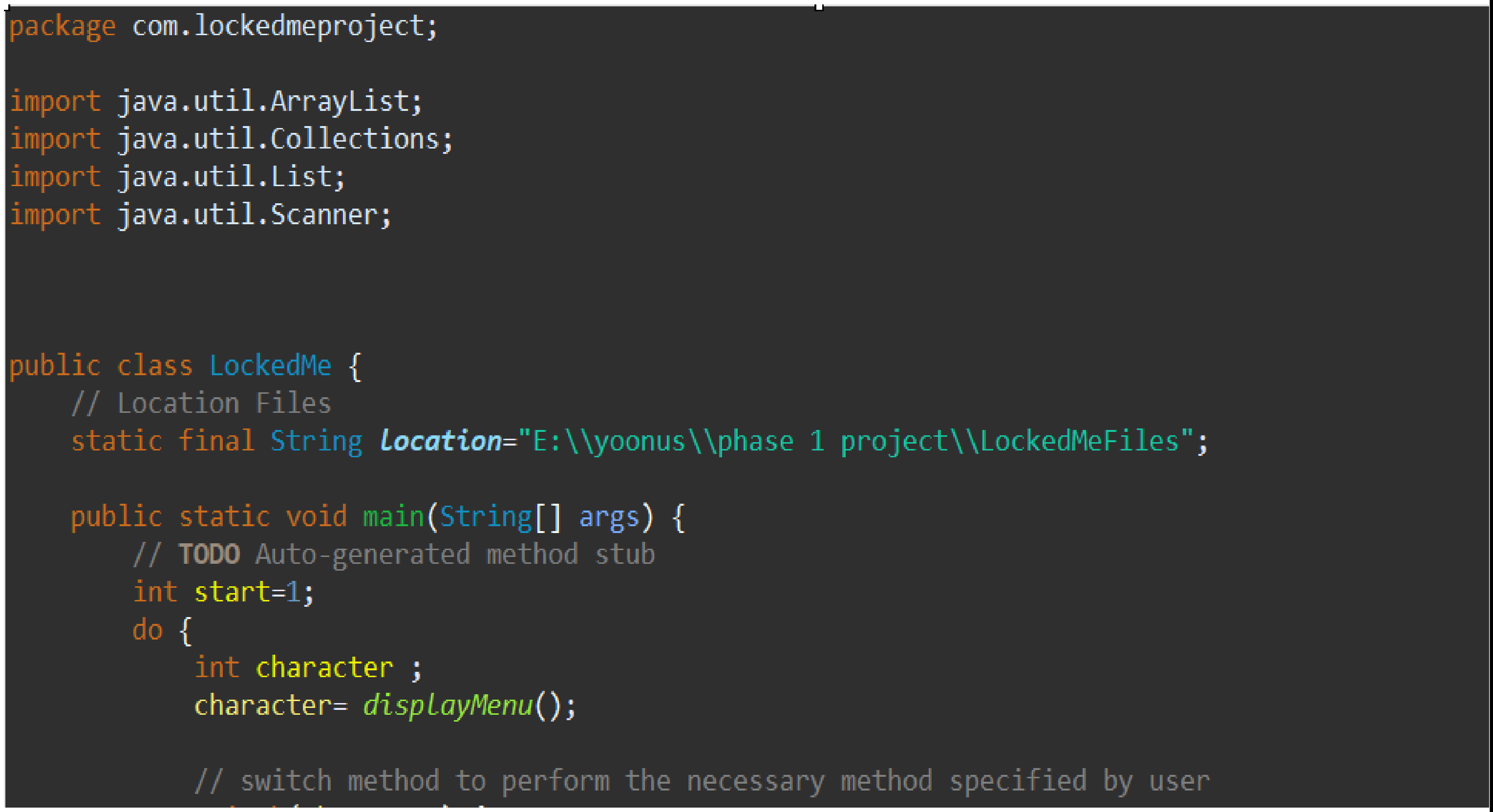
The above picture showcases the folder structure of our source code.

Display menu

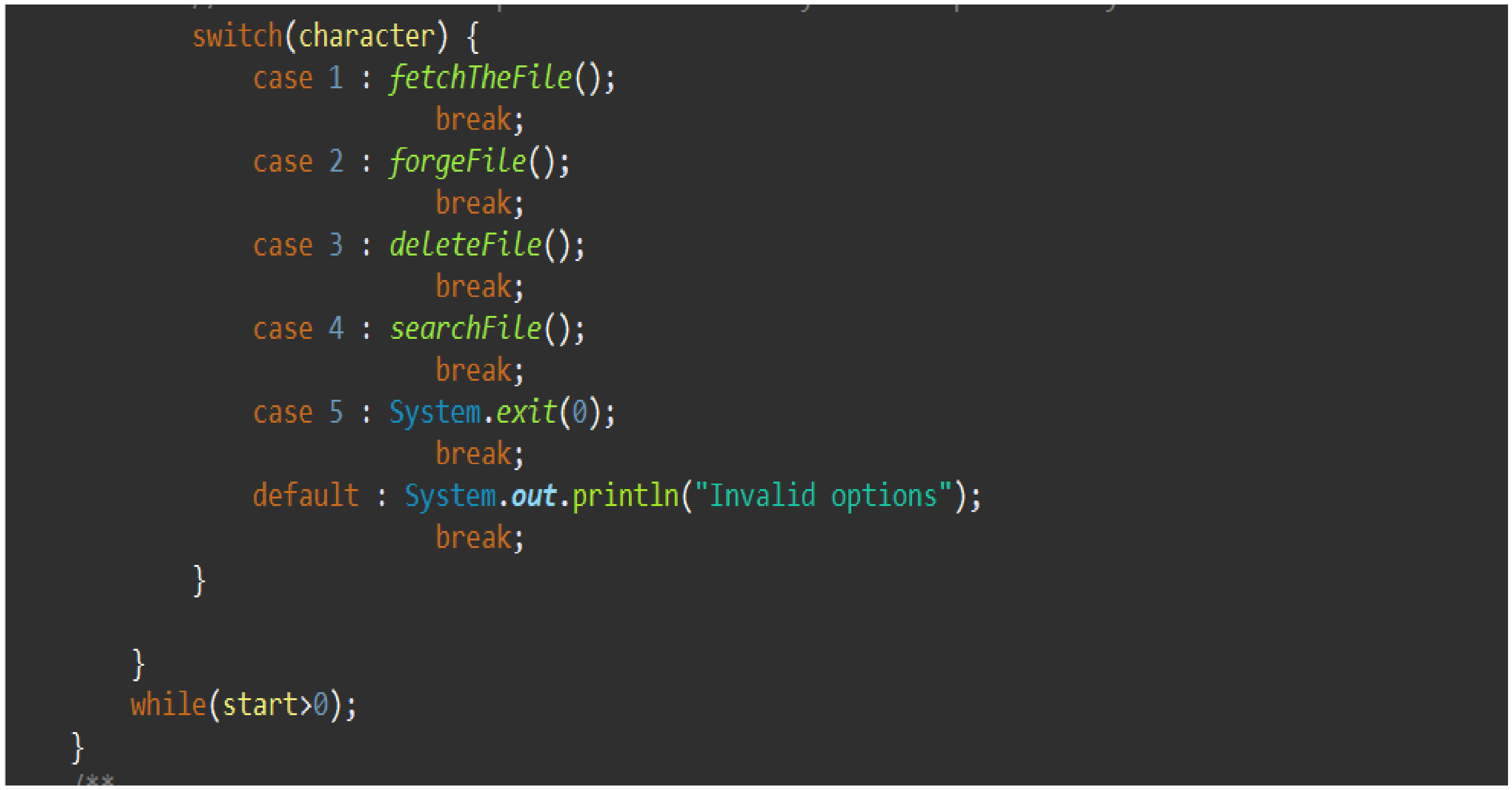
This display menu helps to interact with users and provides a window for the user to access the services provided by the application. The user can choose the options advised by the application to initiate the operations. The code consists of a do while loop which consists of a switch statement with 5 cases. The options are->

- ➔ “(1) - view all the files in a sorted order.
- ➔ “(2)- add new file to the directory
- ➔ “(3)- delete a file from the directory
- ➔ “(4)- search a file from the directory
- ➔ “(5)- exit the application

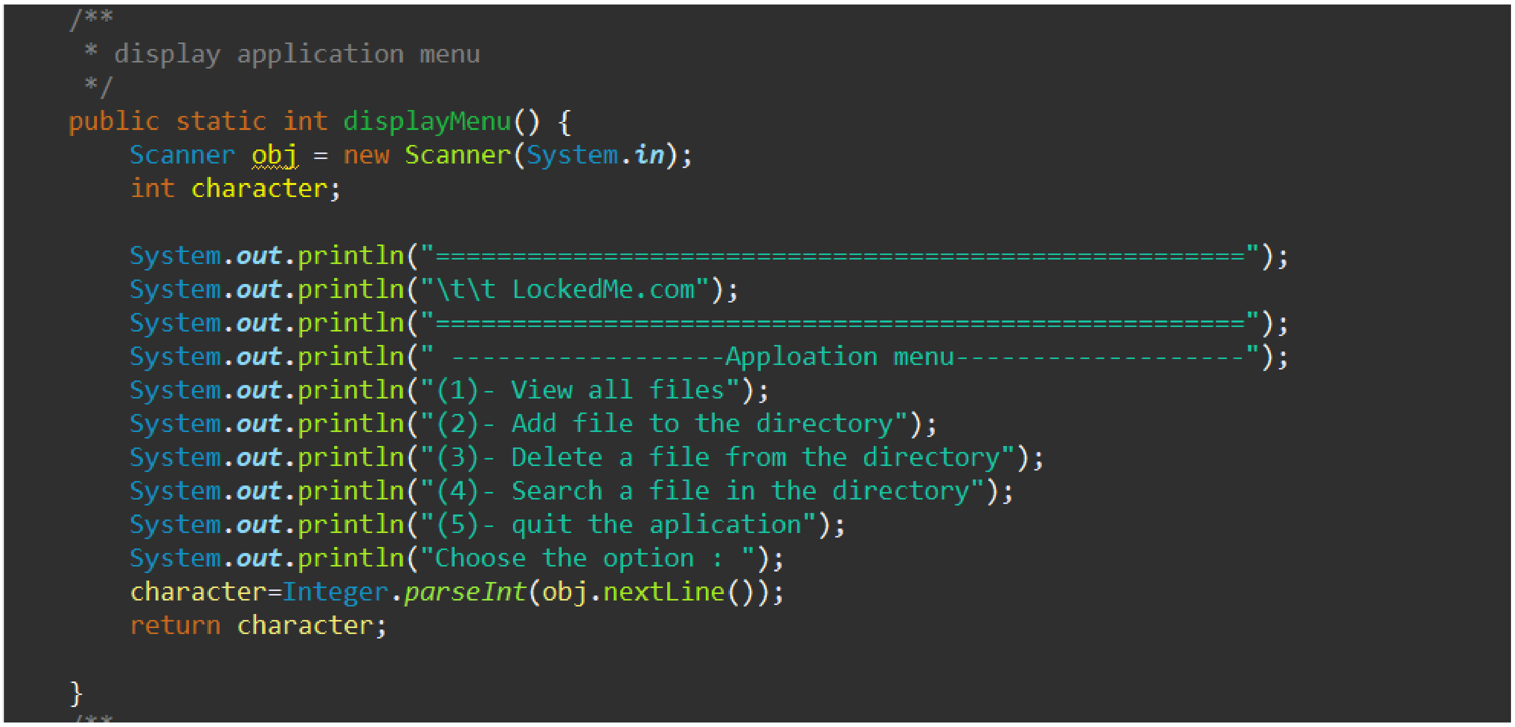
On choosing the desired option the switch statements initiate appropriate methods. After the completion of the operation the console will backtrack to the display menu again. Options “1”, “2” , “3” and “4” are business level operations while option “5” is to quit the application.



```java
package com.lockedmeproject;

import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Scanner;



public class LockedMe {
    // Location Files
    static final String Location="E:\\yoonus\\phase 1 project\\LockedMeFiles";

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int start=1;
        do {
            int character ;
            character= displayMenu();

            // switch method to perform the necessary method specified by user
```



```java
            switch(character) {
                case 1 : fetchTheFile();
                            break;
                case 2 : forgeFile();
                            break;
                case 3 : deleteFile();
                            break;
                case 4 : searchFile();
                            break;
                case 5 : System.exit(0);
                            break;
                default : System.out.println("Invalid options");
                            break;
            }

        }
        while(start>0);
    }
    /**
```

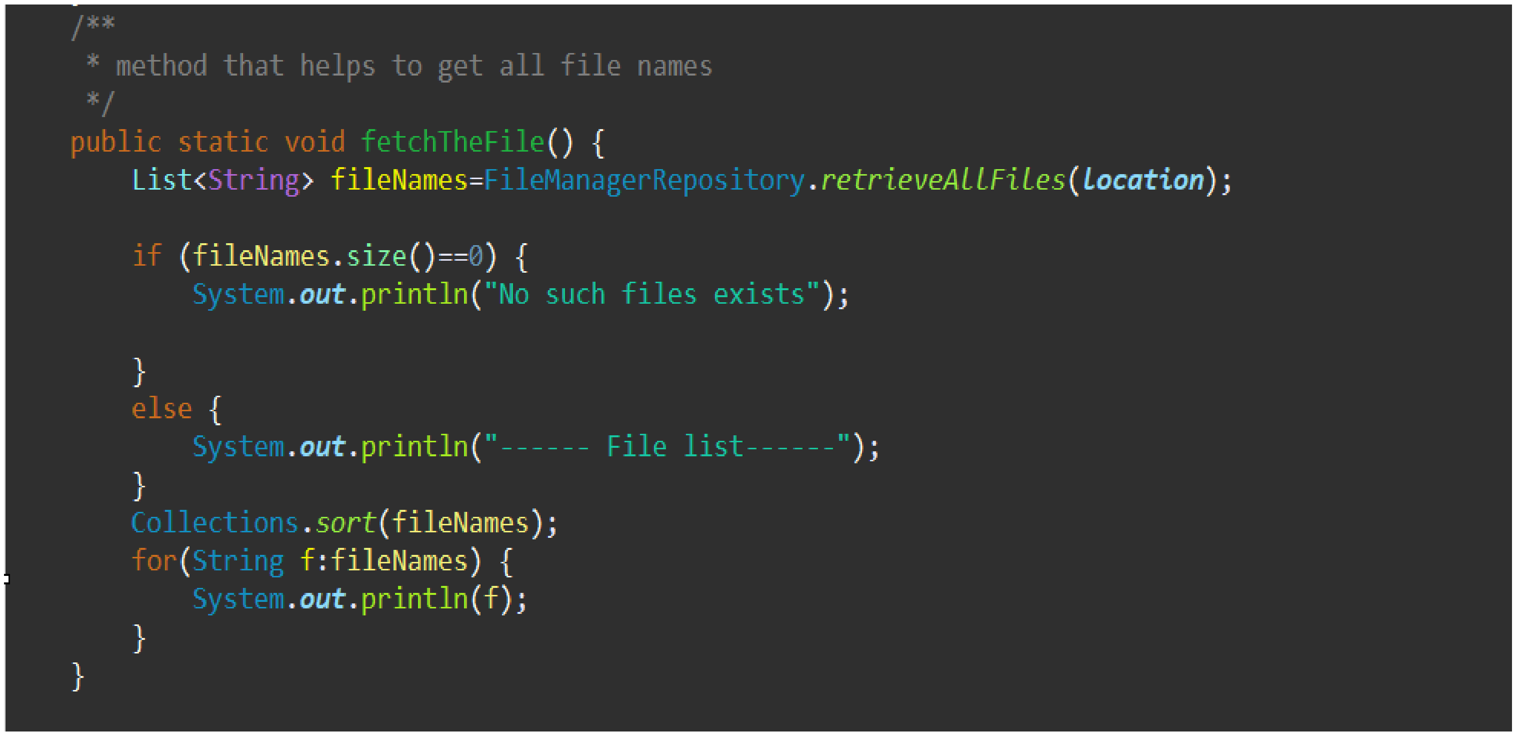


```
    /**
     * display application menu
     */
    public static int displayMenu() {
        Scanner obj = new Scanner(System.in);
        int character;

        System.out.println("=====================================================");
        System.out.println("\t\t LockedMe.com");
        System.out.println("=====================================================");
        System.out.println(" ------------------Apploation menu-------------------");
        System.out.println("(1)- View all files");
        System.out.println("(2)- Add file to the directory");
        System.out.println("(3)- Delete a file from the directory");
        System.out.println("(4)- Search a file in the directory");
        System.out.println("(5)- quit the aplication");
        System.out.println("Choose the option : ");
        character=Integer.parseInt(obj.nextLine());
        return character;

    }
```

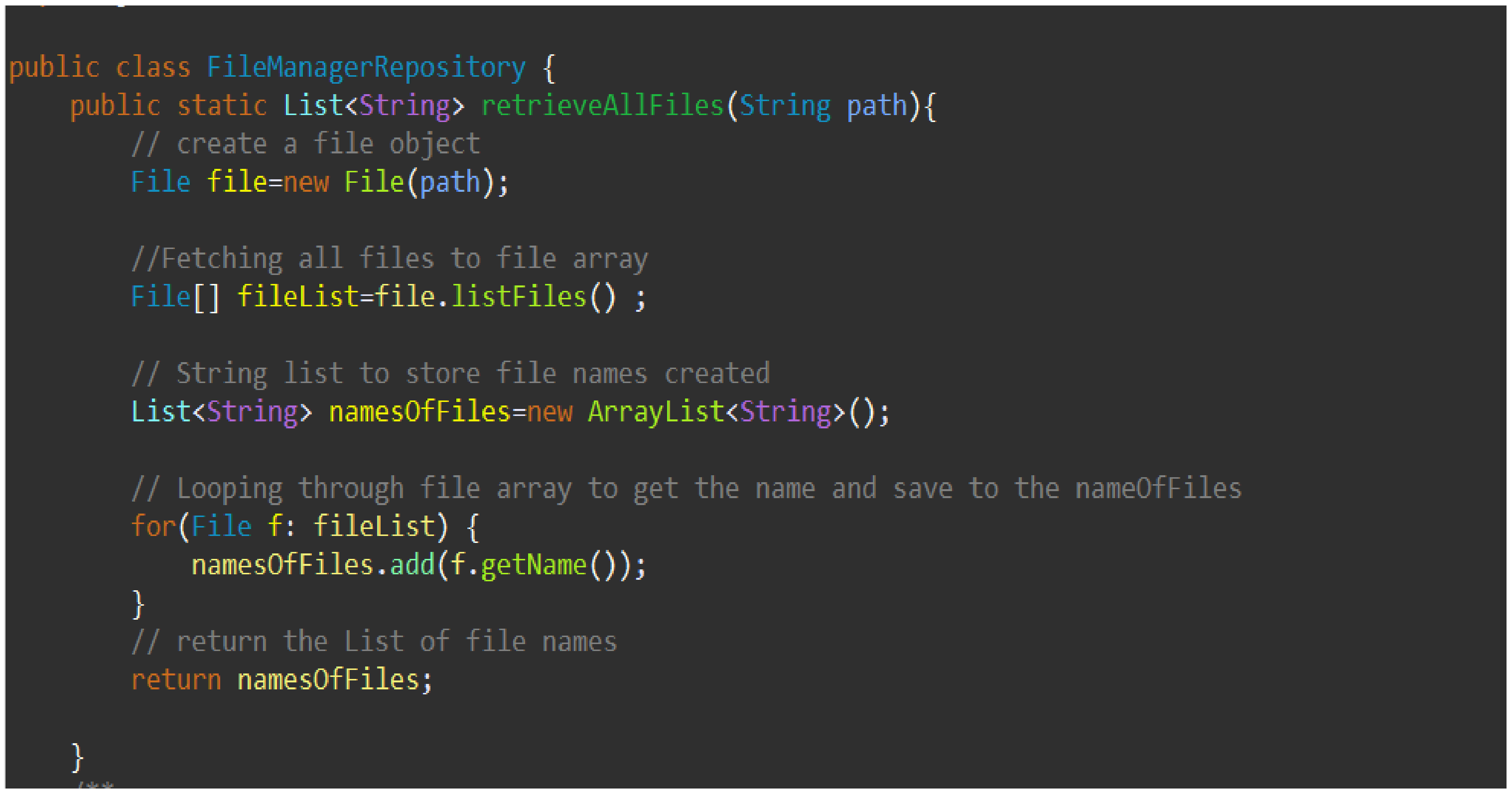
Modules

[1] View the Files in the directory

This method helps to view the files in the directory. The method name is fetchTheFile(). A string variable filename is initiated and is used to store the list of files .To fetch the list we will call the static retrieveAllFiles method from FileManagerRepositrory class by providing location of the directory. Collection.sort() will sort the filenames in the list .



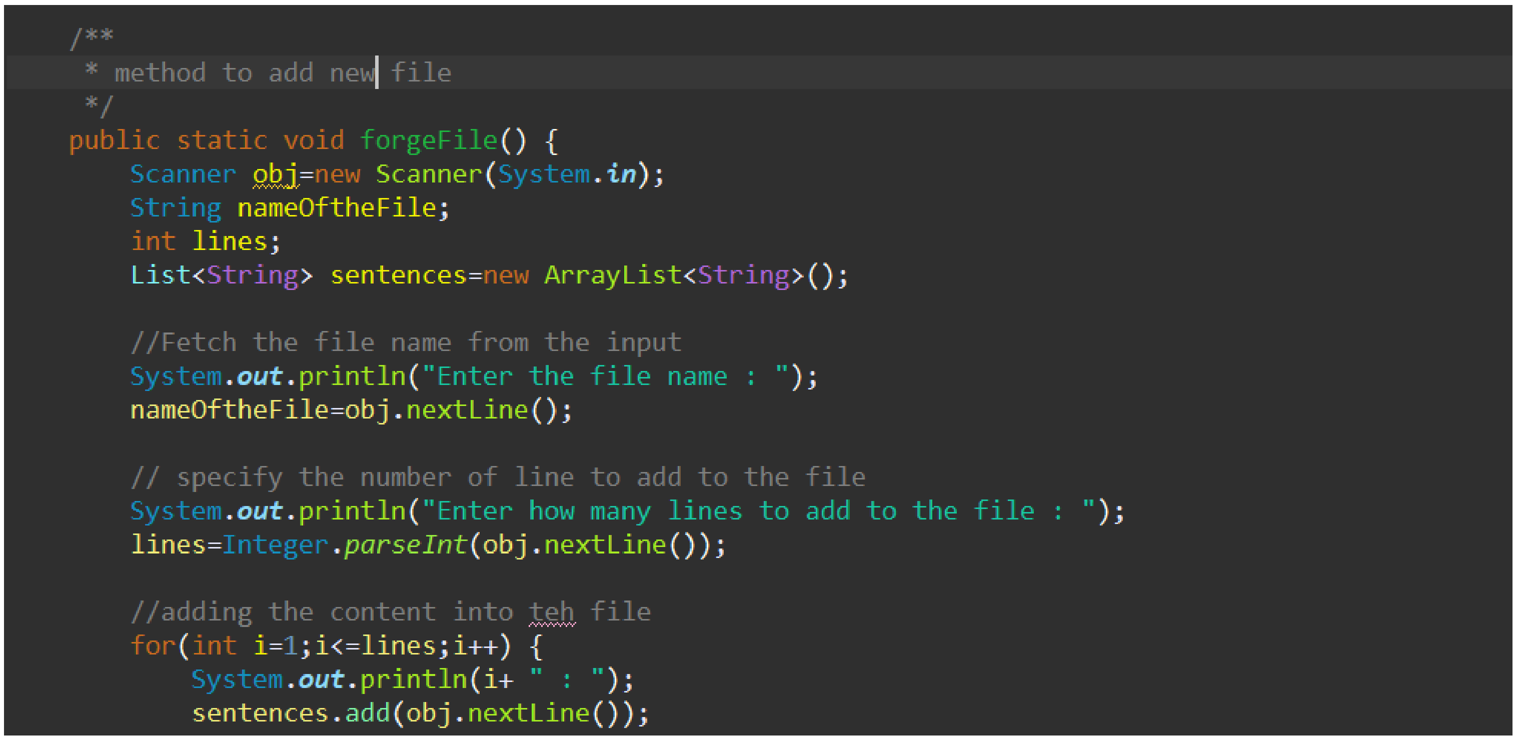
```
    /**
     * method that helps to get all file names
     */
    public static void fetchTheFile() {
        List<String> fileNames=FileManagerRepository.retrieveAllFiles(location);

        if (fileNames.size()==0) {
            System.out.println("No such files exists");

        }
        else {
            System.out.println("------ File list------");
        }
        Collections.sort(fileNames);
        for(String f:fileNames) {
            System.out.println(f);
        }
    }
```



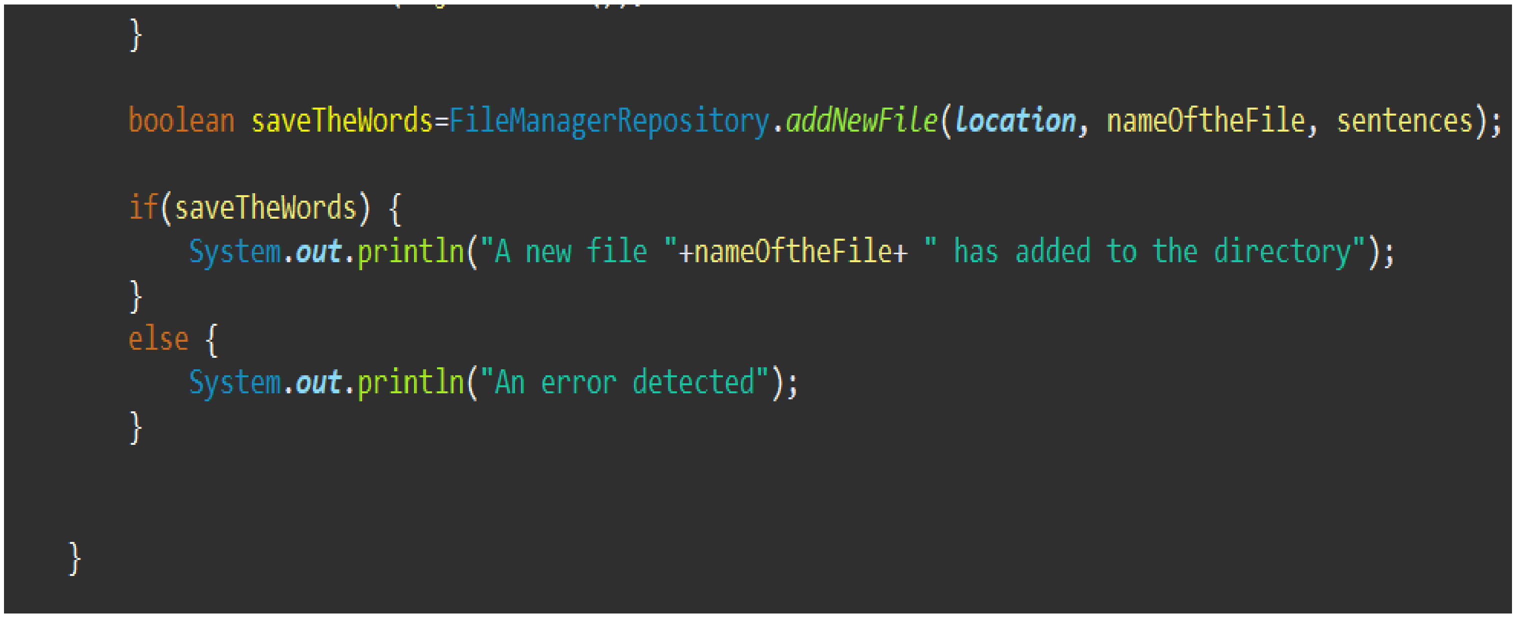
```java
public class FileManagerRepository {
    public static List<String> retrieveAllFiles(String path){
        // create a file object
        File file=new File(path);

        //Fetching all files to file array
        File[] fileList=file.listFiles() ;

        // String list to store file names created
        List<String> namesOfFiles=new ArrayList<String>();

        // Looping through file array to get the name and save to the nameOfFiles
        for(File f: fileList) {
            namesOfFiles.add(f.getName());
        }
        // return the List of file names
        return namesOfFiles;

    }
```

[2] Add new file to the directory

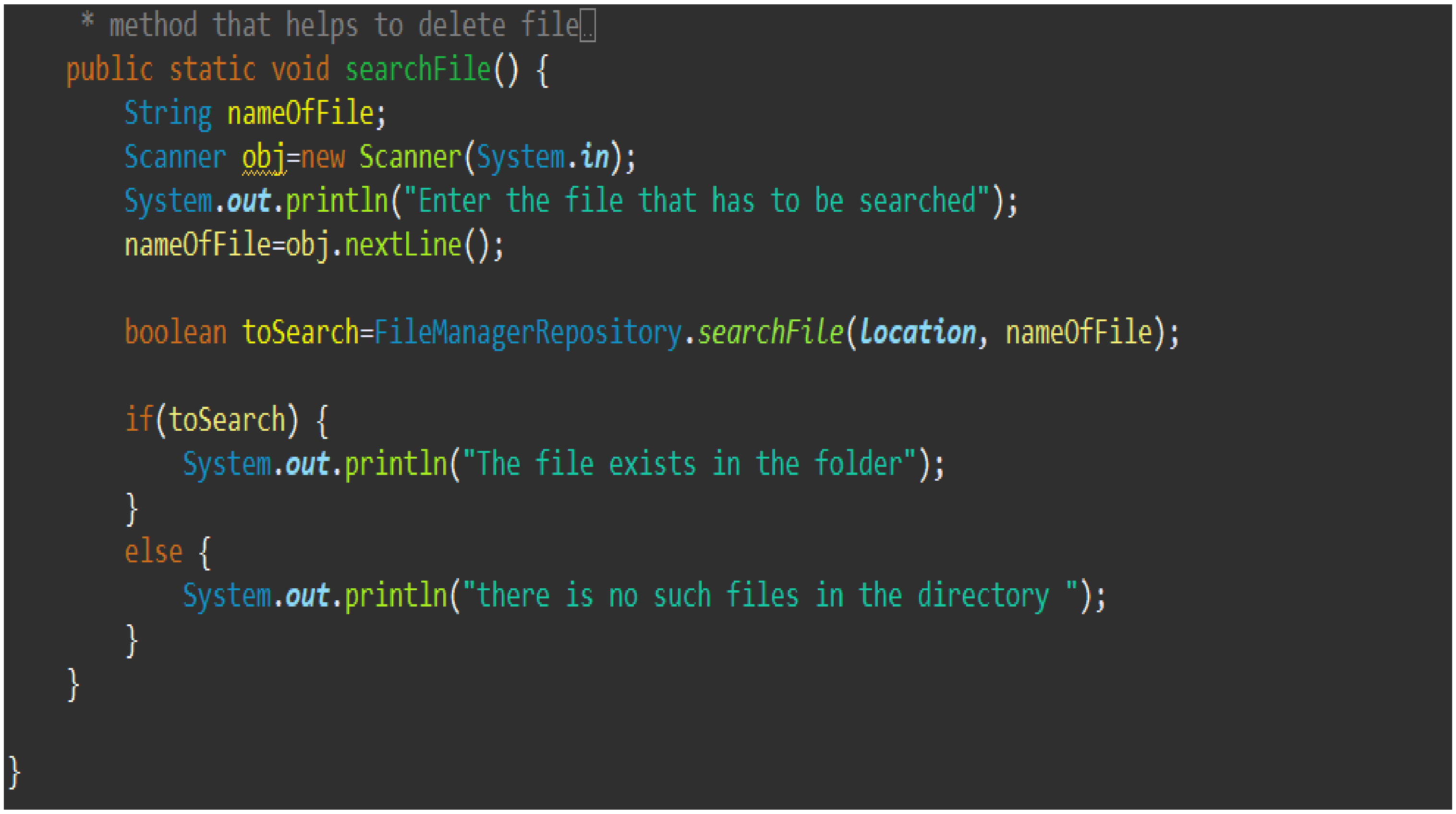
To add a new file into the directory the users can enter 2 in the console window. The method that helps to create a new file is forgeFile(). The string variables nameOftheFile and sentences stores the name of the new file and contents that is to be added to new file. With the help of static method addNewFile() from FileManagerRepository a new file is created. addNewFile() method return true if a new file is successfully created else it will return false .



```java
    /**
     * method to add new file
     */
    public static void forgeFile() {
        Scanner obj=new Scanner(System.in);
        String nameOftheFile;
        int lines;
        List<String> sentences=new ArrayList<String>();

        //Fetch the file name from the input
        System.out.println("Enter the file name : ");
        nameOftheFile=obj.nextLine();

        // specify the number of line to add to the file
        System.out.println("Enter how many lines to add to the file : ");
        lines=Integer.parseInt(obj.nextLine());

        //adding the content into teh file
        for(int i=1;i<=lines;i++) {
            System.out.println(i+ " : ");
            sentences.add(obj.nextLine());
```



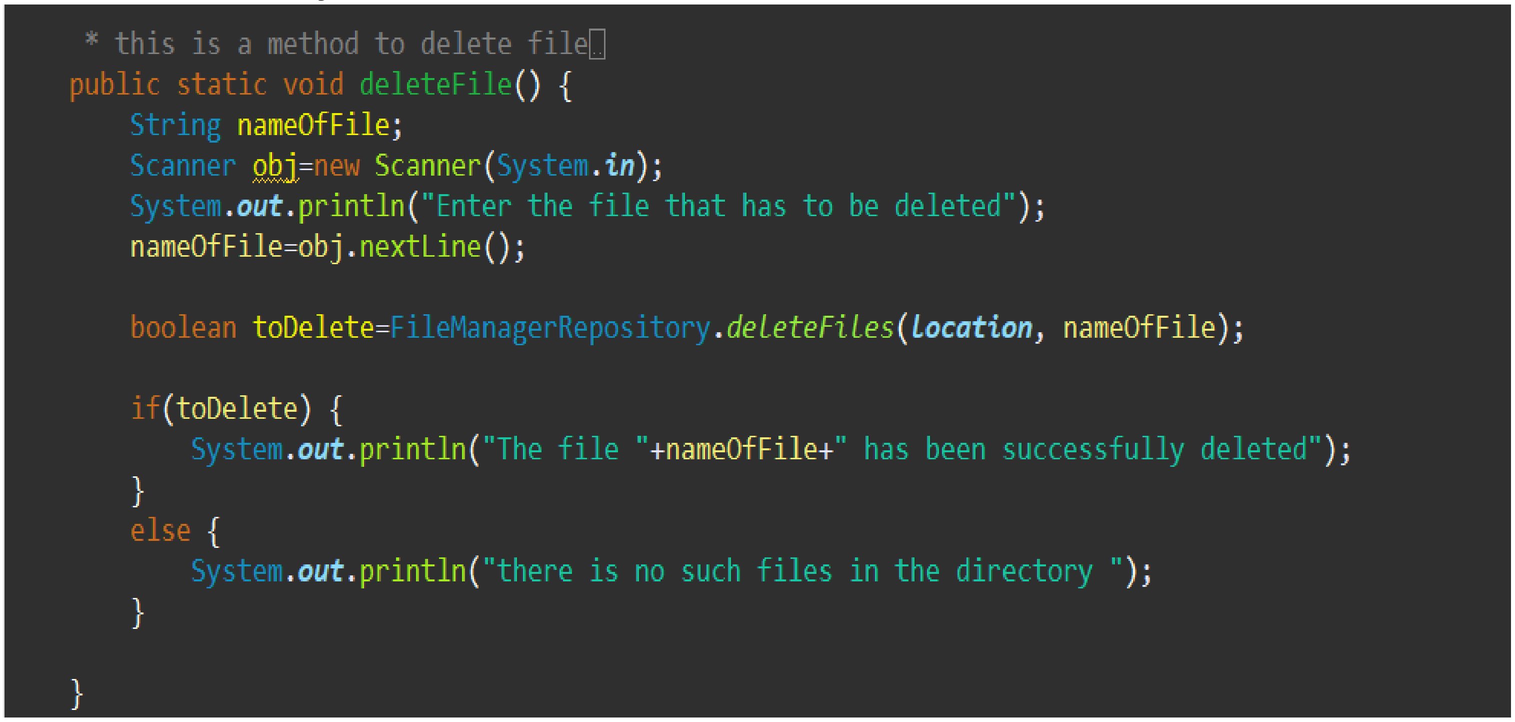
```
        }

        boolean saveTheWords=FileManagerRepository.addNewFile(location, nameOftheFile, sentences);

        if(saveTheWords) {
            System.out.println("A new file "+nameOftheFile+ " has added to the directory");
        }
        else {
            System.out.println("An error detected");
        }


    }
```



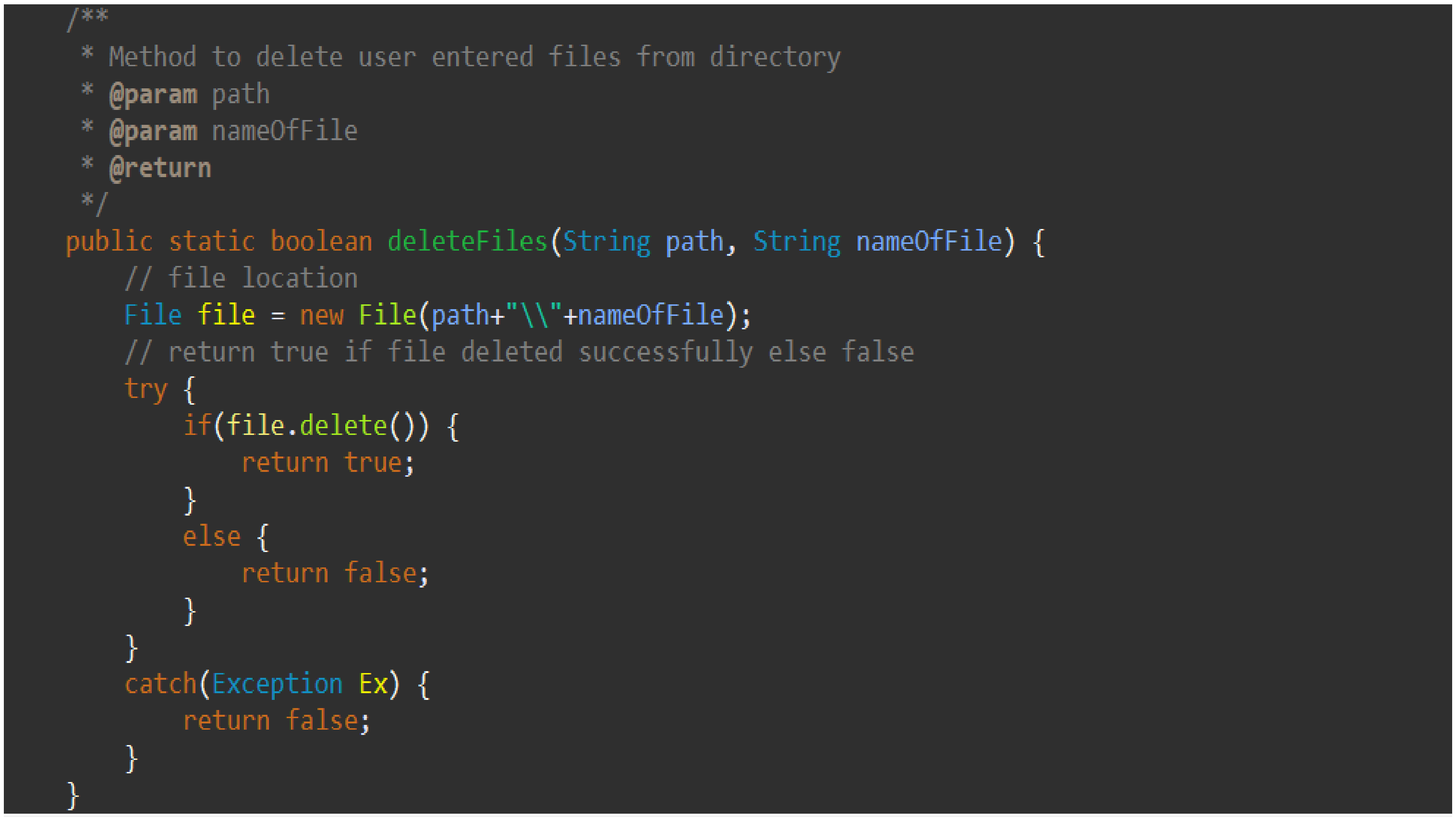
```
     * method that helps to delete file
    public static void searchFile() {
        String nameOfFile;
        Scanner obj=new Scanner(System.in);
        System.out.println("Enter the file that has to be searched");
        nameOfFile=obj.nextLine();

        boolean toSearch=FileManagerRepository.searchFile(location, nameOfFile);

        if(toSearch) {
            System.out.println("The file exists in the folder");
        }
        else {
            System.out.println("there is no such files in the directory ");
        }
    }

}
```

[iii] Delete a file

deleteFile() method helps to delete the file from the directory. String variable nameOfFile will store the element that is to be deleted. Static method deleteFiles() from FileManagerRepository class is called to delete the file. The deleteFiles() will take the two parameters: location of the folder and name of the file to be deleted. Upon deleting it will return true if the file has successfully been deleted or false if any error occurs.



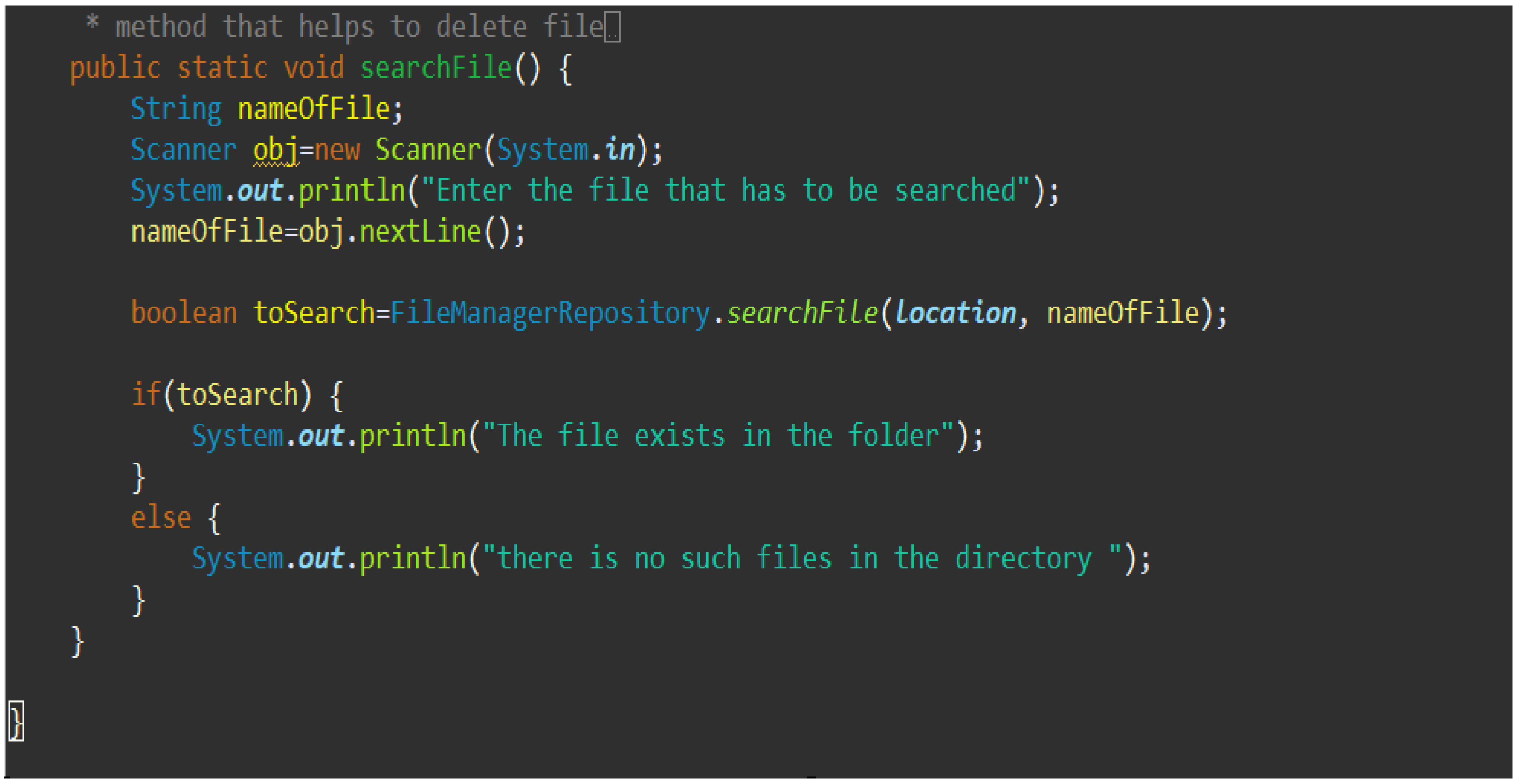
```java
     * this is a method to delete file
    public static void deleteFile() {
        String nameOfFile;
        Scanner obj=new Scanner(System.in);
        System.out.println("Enter the file that has to be deleted");
        nameOfFile=obj.nextLine();

        boolean toDelete=FileManagerRepository.deleteFiles(location, nameOfFile);

        if(toDelete) {
            System.out.println("The file "+nameOfFile+" has been successfully deleted");
        }
        else {
            System.out.println("there is no such files in the directory ");
        }

    }
```



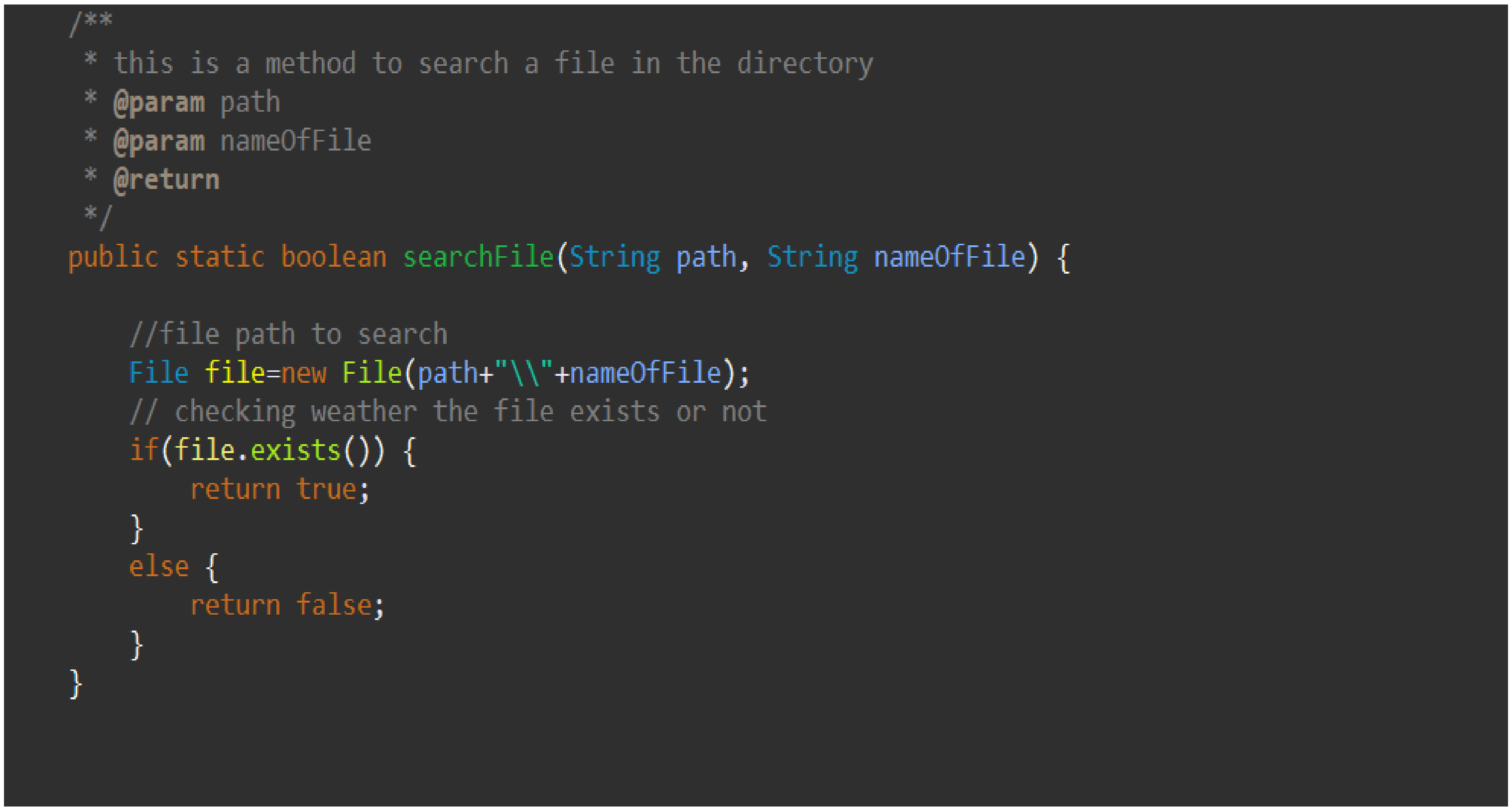
```java
    /**
     * Method to delete user entered files from directory
     * @param path
     * @param nameOfFile
     * @return
     */
    public static boolean deleteFiles(String path, String nameOfFile) {
        // file location
        File file = new File(path+"\\"+nameOfFile);
        // return true if file deleted successfully else false
        try {
            if(file.delete()) {
                return true;
            }
            else {
                return false;
            }
        }
        catch(Exception Ex) {
            return false;
        }
    }
```

[iv] Search a file in the directory

searchFile() method helps to search user specified files in the directory. String variable nameOfFile stores the name of the file that is to be searched.Static method searchFile() from FileManagerRepository class is called to search the file. The searchFile() will take the two parameters: location of the folder and name of the file to be searched. Upon searching it will return true if the file has successfully been searched or false if any error occurs.

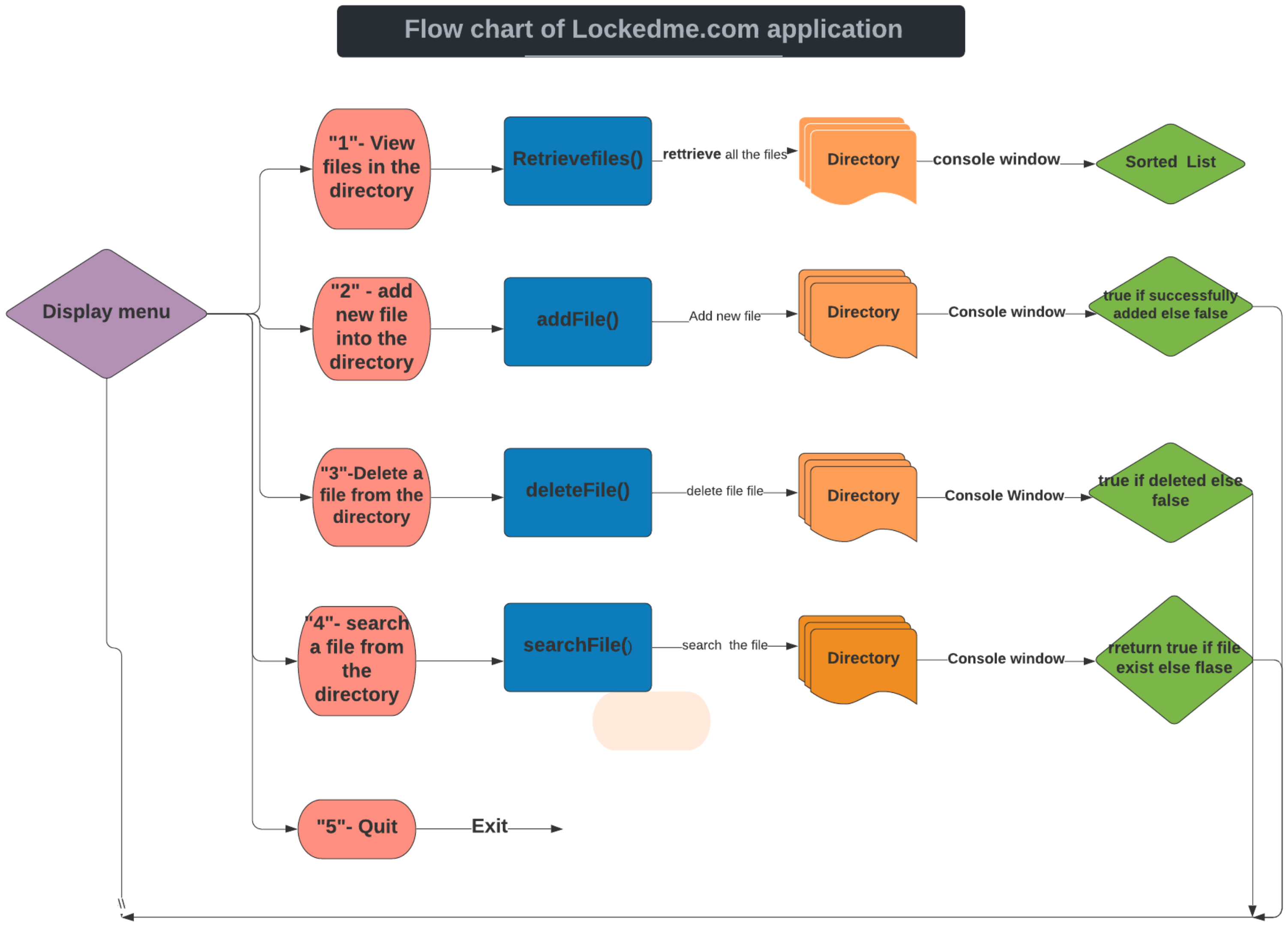


```java
     * method that helps to delete file
    public static void searchFile() {
        String nameOfFile;
        Scanner obj=new Scanner(System.in);
        System.out.println("Enter the file that has to be searched");
        nameOfFile=obj.nextLine();

        boolean toSearch=FileManagerRepository.searchFile(location, nameOfFile);

        if(toSearch) {
            System.out.println("The file exists in the folder");
        }
        else {
            System.out.println("there is no such files in the directory ");
        }
    }

}
```



```java
    /**
     * this is a method to search a file in the directory
     * @param path
     * @param nameOfFile
     * @return
     */
    public static boolean searchFile(String path, String nameOfFile) {

        //file path to search
        File file=new File(path+"\\"+nameOfFile);
        // checking weather the file exists or not
        if(file.exists()) {
            return true;
        }
        else {
            return false;
        }
    }
```

Flow chart



Flow chart of Lockedme.com application
Display menu
"1"- View files in the directory
Retrievefiles()
rettrieve all the files
Directory
console window
Sorted List
"2" - add new file into the directory
addFile()
Add new file
Directory
Console window
true if successfully added else false
"3"-Delete a file from the directory
deleteFile()
delete file file
Directory
Console Window
true if deleted else false
"4"- search a file from the directory
searchFile()
search the file
Directory
Console window
rreturn true if file exist else flase
"5"- Quit
Exit

<u>Concepts covered :</u>

- ➔ Eclipse IDE is used to develop the application
- ➔ Java concepts covered- class,methods, static methods, de while loop,
- ➔ Agile scrum has been followed for developing the tool
- ➔ Familiarised with Github
- ➔ Data structure covered - Sorting , Arrays

<u>Conclusion</u>

The prototype of the application has been successfully implemented. All necessary modules have been implemented in the application.
The application can be improved by including more features such as:

- ➔ **Allow the user to append data to the existing file.**
- ➔ **Retrieving files based on various criteria such as Last Modified, Type etc.**