**Locked Me**

**(Virtual Key for Repositories)**

Version 1.0

**Prepared By Rohit Kumar Jha**

**Date: 18-10-2022**

**Contents:**

This document Contains below details in different sections:

* Sprint Planning
* Core Concepts used in projects
* Flow Chart
* Demonstrating the code
* Points of the Application
* Conclusions

**The Code For This Project Is Hosted At:**

[**https://github.com/rohitkum549/Locked.in.git**](https://github.com/rohitkum549/Locked.in.git)

**Sprints planning and Task completion**

The project is planned to be completed in 3 sprints. Tasks assumed to be completed in the sprints are:

**Sprint 1:**

* Creating the flow of the application -flow chart
* Initializing git repository to track changes as development progresses.
* Writing the Java program for Welcome Screen with main menu
* Pushing code to GitHub.

**Sprint 2:**

* Writing the Java program to Enable User Input for main menu and the corresponding actions accordingly
* Testing the Java program with different kinds of User input
* Writing the Java program for secondary menu
* Pushing code to GitHub.

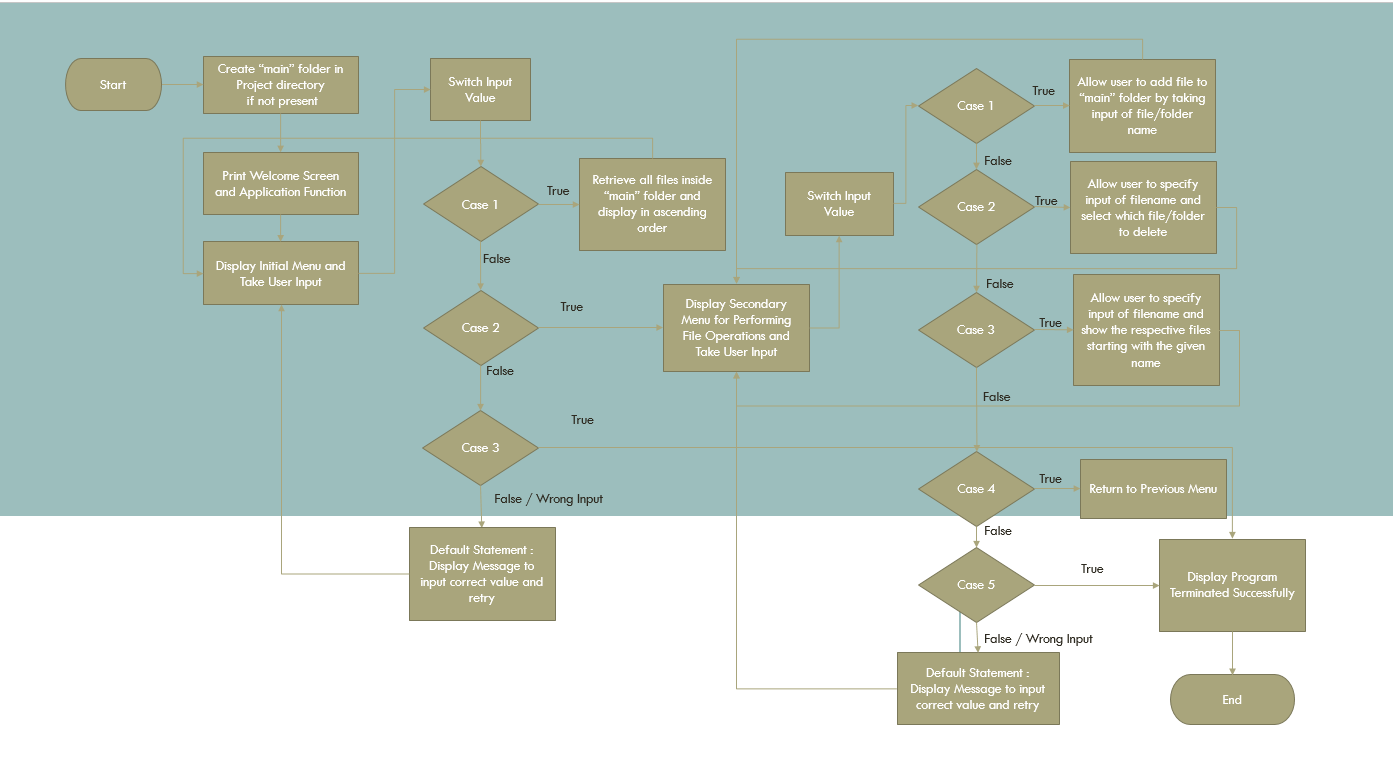
**Sprint 3:**

* Writing the Java program to Enable User Input for Secondary menu and the corresponding actions accordingly
* Testing the Java program with different kinds of User input for secondary menu
* Pushing code to GitHub.
* Creating this specification document highlighting application capabilities, Test Screen shots and user interactions.

## Core concepts used in project

* Collections framework
* File Handling
* Sorting
* Flow Control
* Recursion
* Exception Handling
* Streams API

**Flow Chart**

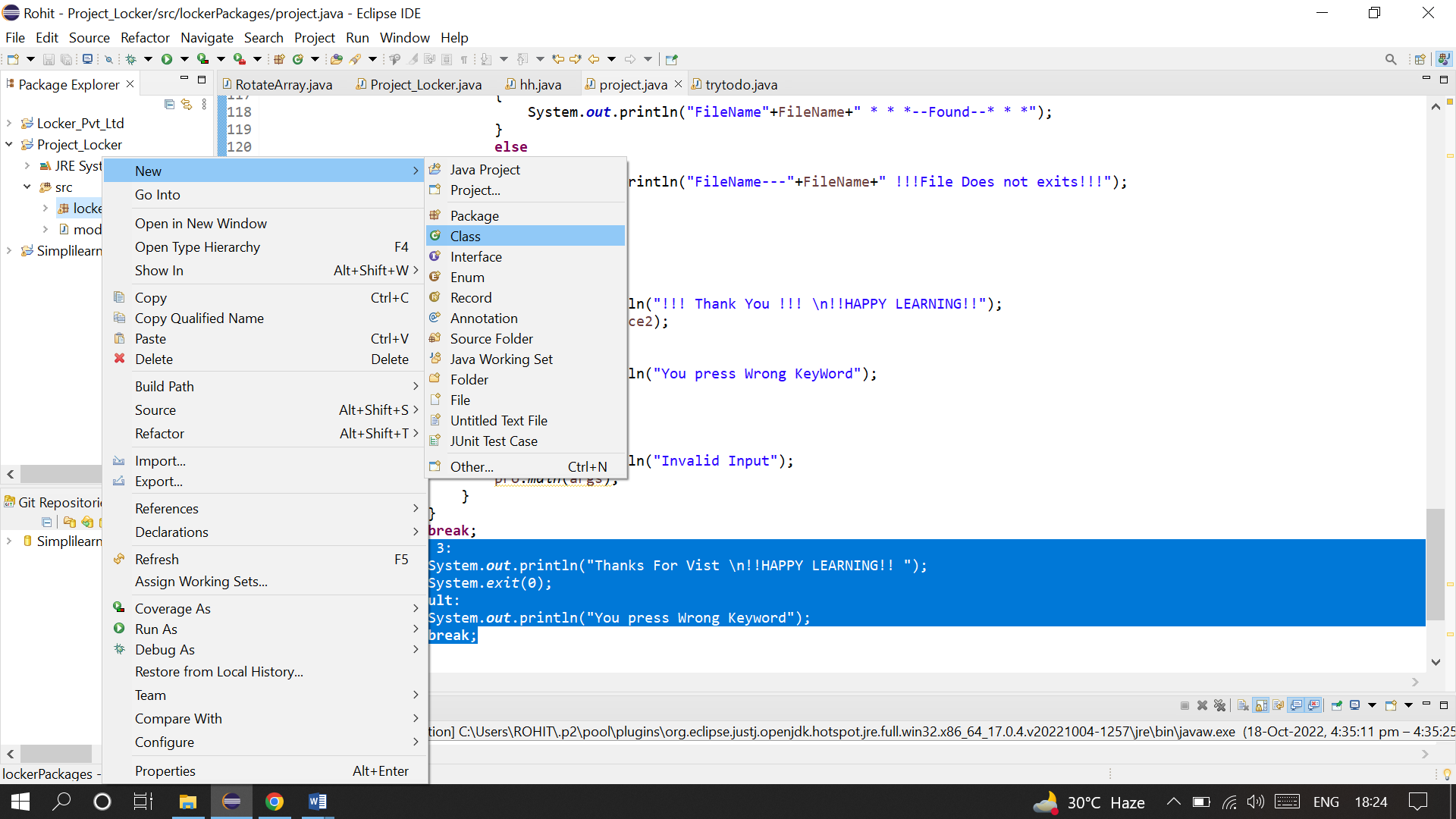


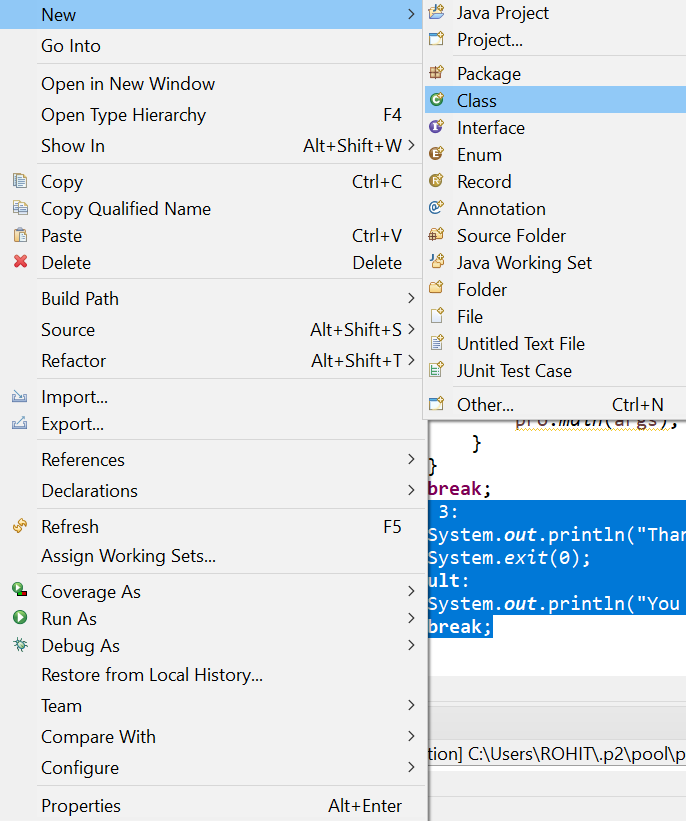
Product Description, Code walkthrough, Test Evidence and the procedure

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

## **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 **project** 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 (**project.java**)

**package** lockerPackages;

**public** **class** project

{

**public** **static** **void** main(String[] args)

{

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

{

project pro=**new** project();

Scanner X=**new** Scanner(System.***in***);

System.***out***.println("\tEnter the DirPath");

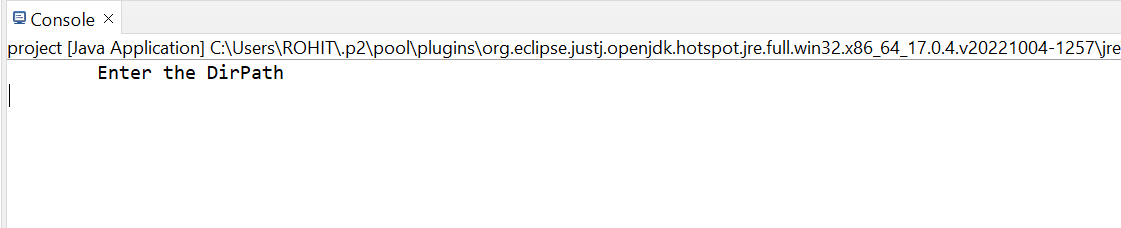
String DirPath=X.nextLine();

File folder=**new** File(DirPath);

}

}

}



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

project pro=**new** project();

Scanner X=**new** Scanner(System.***in***);

System.***out***.println("\tEnter the DirPath");

String DirPath=X.nextLine();

File folder=**new** File(DirPath);

System.***out***.println("\t\* \* \* \* \* \* \* \* Welcome Here \* \* \* \* ");

System.***out***.println("\t\* \* \* \* \* \* \* \* Locker.in \* \* \* \* \* \*");

System.***out***.println("\tApplication Developer: Rohit KumarJha");

System.***out***.println("\tFile menu Related menu Application");

System.***out***.println();

System.***out***.println("\t|^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^|");

System.***out***.println("\t| MAIN MENU |");

System.***out***.println("\t|^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^|");

System.***out***.println("\t| Select any one of the following: |");

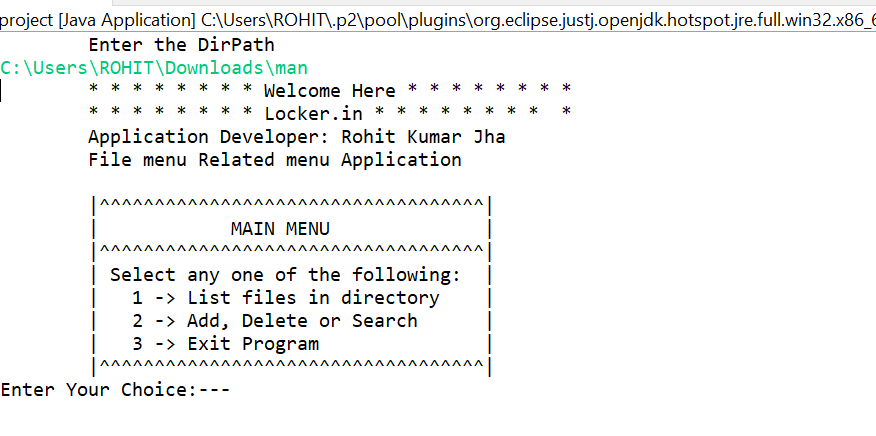
System.***out***.println("\t| 1 -> List files in directory |");

System.***out***.println("\t| 2 -> Add, Delete or Search |");

System.***out***.println("\t| 3 -> Exit Program |");

System.***out***.println("\t|^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^|");

System.***out***.print("Enter Your Choice:--- ");



* 1. List Files in Directory

**case** 1:

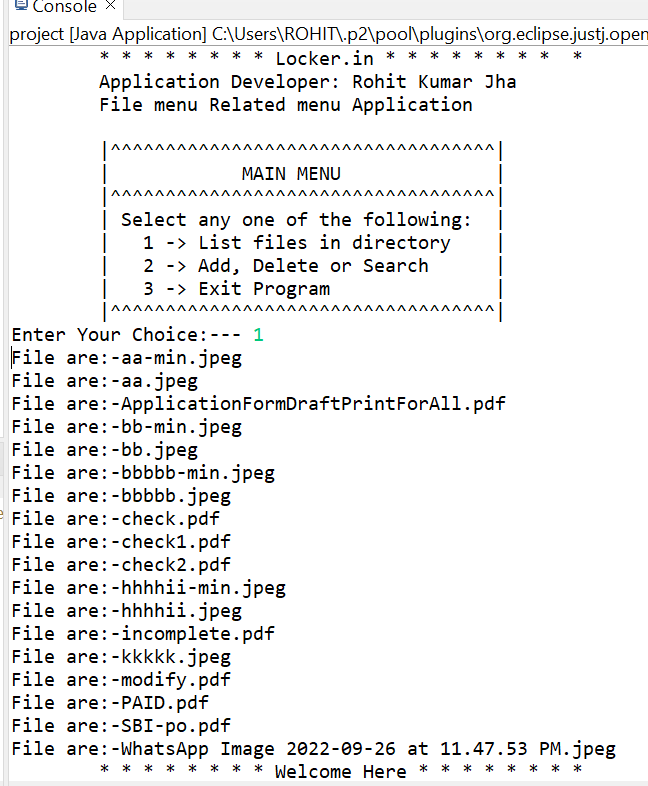
File[] FileList=folder.listFiles();

Arrays.*sort*(FileList);

**for**(**int** i=0;i<FileList.length;i++)

{

System.***out***.println("File are:-"+FileList[i].getName());

}**break**; 

* 1. Add

**case** 1: /\*ADD\*/

System.***out***.println("Enter The File Name:-");

String FileNameAdd=X.next();

System.***out***.println();

**if**(**new** File(folder,FileNameAdd).exists())

{

System.***out***.println("Already Exits Please Try Different Name---"+FileNameAdd);

}

**else**

{

File AddFile=**new** File(DirPath+"/"+FileNameAdd);

**if** (AddFile.createNewFile())

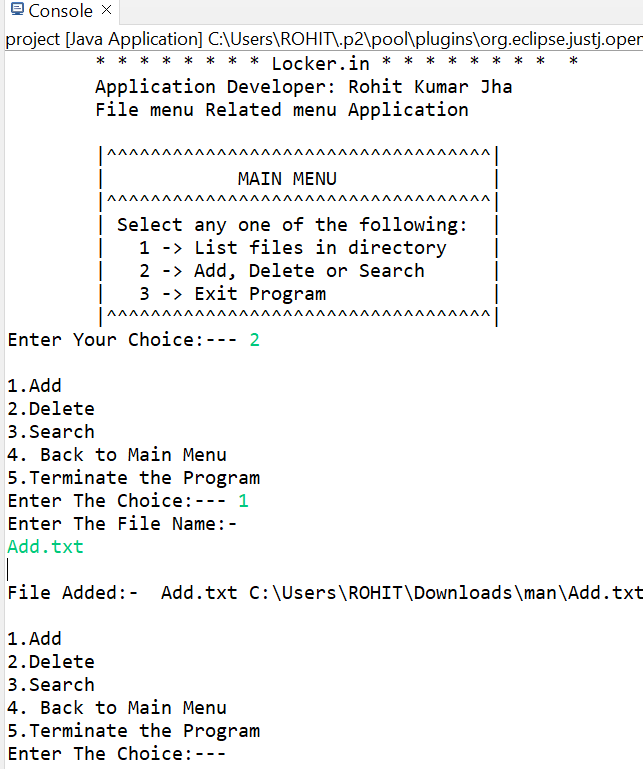
{

System.***out***.println("File Added:- "+FileNameAdd+" "+AddFile);

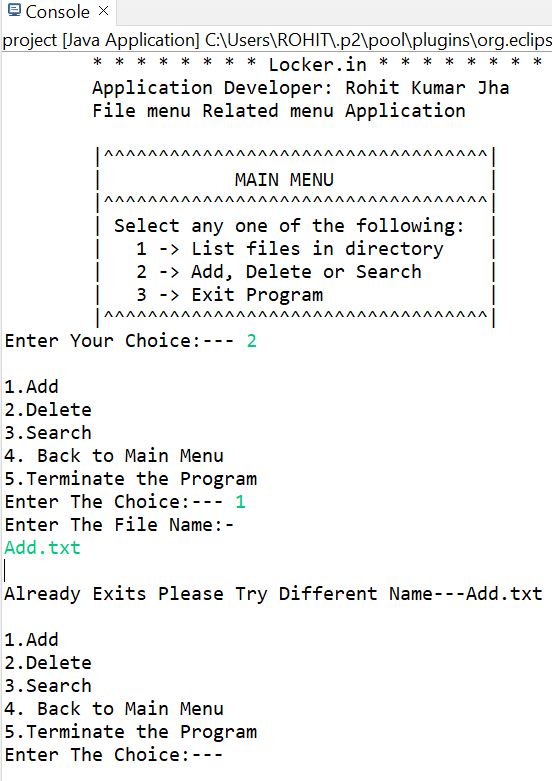
}

}

**break**;



IF file already Exits



* 1. Delete

**case** 2: /\*DELETE\*/

System.***out***.println("Enter the FileName To Delete:--");

String DeleteFileName=X.next();

File DeleteFile=**new** File(DirPath+"/"+DeleteFileName);

**if**(DeleteFile.delete())

{

System.***out***.println(DeleteFileName+" !!-Deleted Sucessfully!!");

}

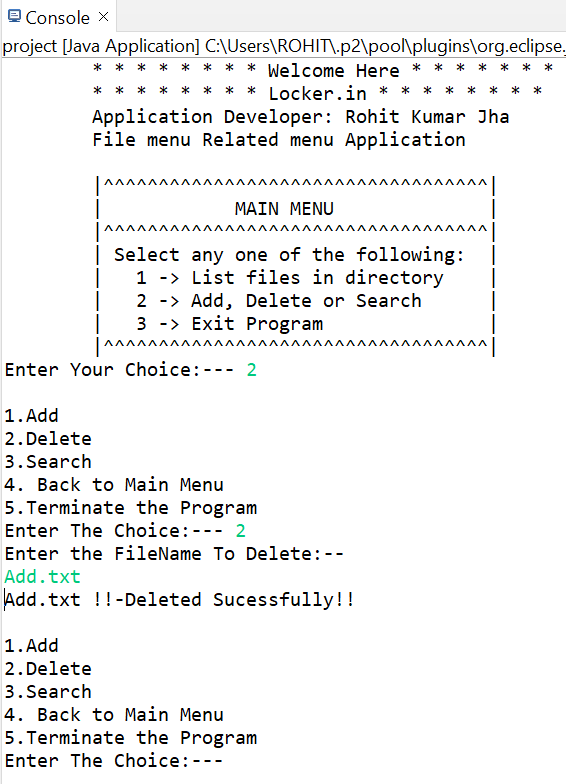
**else**

{

System.***out***.println("!!!-File not Found!!!");

}

**break**;



* 1. Search

**case** 3: /\*SEARCH\*/

System.***out***.println("Enter The File Name:-");

String FileName=X.next();

File SearchFileDir[]=folder.listFiles();

**boolean** found=**false**;

**for**(**int** i=0;i<SearchFileDir.length;i++)

{

**if**(SearchFileDir[i].getName().startsWith(FileName))

{

found=**true**;

**break**;

}

}

**if**(found)

{

System.***out***.println("FileName"+FileName+" \* \* \*--Found--\* \* \*");

}

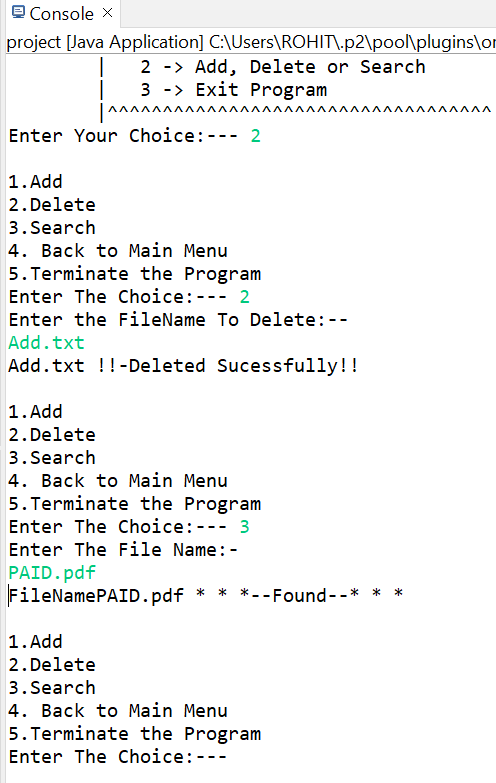
**else**

{

System.***out***.println("FileName---"+FileName+" !!!File Does not exits!!!");

}

**break**;



* 1. Terminate Program

**case** 3:

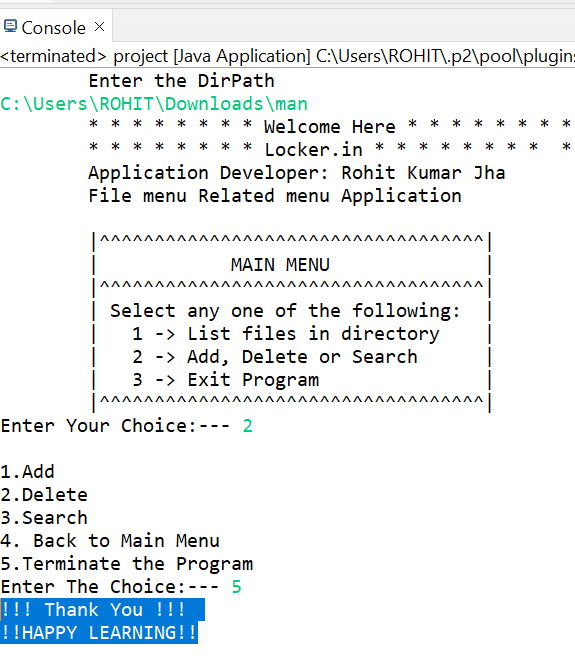
System.***out***.println("Thanks For Vist \n!!HAPPY LEARNING!! ");

System.*exit*(0);

**default**:

System.***out***.println("You press Wrong Keyword");

**break**;



## **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 any file/folder name as input. Even if the user wants to create nested folder structure, user can specify the relative path, and the application takes care of creating the required folder structure.
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/folder. They can specify the starting input, and the program searches all files/folder 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 application also allows user to delete folders which are not empty.
6. 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.
7. When the option to retrieve files in ascending order is selected, user is displayed with two options of viewing the files.
   1. Ascending order of folders first which have files sorted in them,
   2. Ascending order of all files and folders inside the “main” folder.
8. The application is designed with modularity in mind. 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.

## 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.
* Asking user to verify if they really want to delete the selected directory if it’s not empty.
* Retrieving files/folders by different criteria like Last Modified, Type, etc.
* Allowing user to append data to the file.