Virtual Key for Repositories

This document contains sections for:

The code for this project is hosted at https://github.com/Prateekdu/Phase-1-Practice-Project.git .
The project is developed by Prateek Dubey.

Sprints planning and Task completion

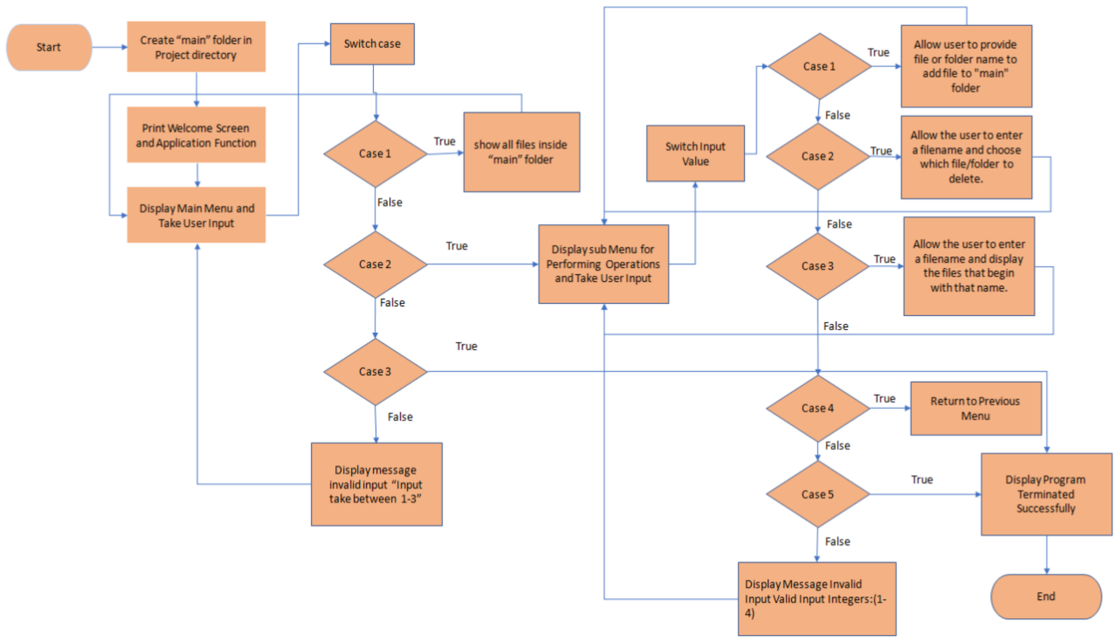
The project is planned to be completed in 1 sprint. Tasks assumed to be completed in the sprint are:

- 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, Recursion, Exception Handling, Streams API

Flow of the Application


Start
Create "main" folder in Project directory
Print Welcome Screen and Application Function
Display Main Menu and Take User Input
Switch case
Case 1
True
show all files inside "main" folder
False
Case 2
True
Display sub Menu for Performing Operations and Take User Input
False
Case 3
True
False
Display message invalid input "Input take between 1-3"
Switch Input Value
Case 1
True
Allow user to provide file or folder name to add file to "main" folder
False
Case 2
True
Allow the user to enter a filename and choose which file/folder to delete.
False
Case 3
True
Allow the user to enter a filename and display the files that begin with that name.
False
Case 4
True
Return to Previous Menu
False
Case 5
True
Display Program Terminated Successfully
False
Display Message Invalid Input Valid Input Integers:(1-4)
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 (**Main.java**)
3. Writing a program in Java to display Menu options available for the user (**VertualDemo.java**)
4. Writing a program in Java to handle Menu options selected by user (**VirtualDemo.java**)
5. Writing a program in Java to perform the File operations as specified by user (**Operations.java**)
6. Pushing the code to GitHub repository

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 **Main** 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 (**Main.java**)

```
package VirtualKeyFotYourRepositories;

public class Main {
public static final String path = "C:\\Users\\Prateek\\Desktop\\assessment1";

	public static void main(String[] args) {
		VirtualDemo vd = new VirtualDemo();
		vd.introScreen();
		vd.mainMenu();
	}

}
```

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

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

3.1. Displaying Welcome Screen

3.2. Displaying Main Menu

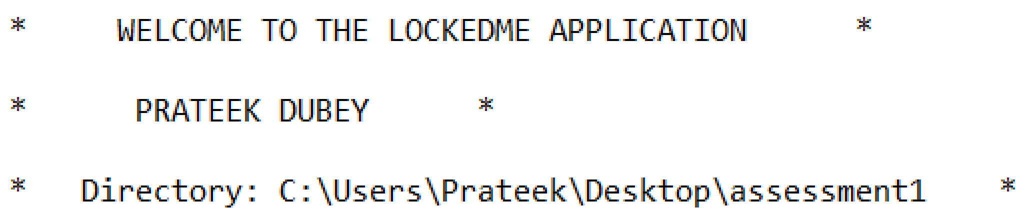
3.3. Displaying Sub Menu for File Operations

Step 3.1: Writing method to display Welcome Screen

```
public void introScreen()
	{
		System.out.println();
```

```
                System.out.println("*     WELCOME TO THE LOCKEDME APPLICATION      *");
                System.out.println();
                System.out.println("*      PRATEEK DUBEY      *");
                System.out.println();
                System.out.println("*    Directory: " + Main.path +"     *");
                System.out.println("\n\n");
        }
```

Output:

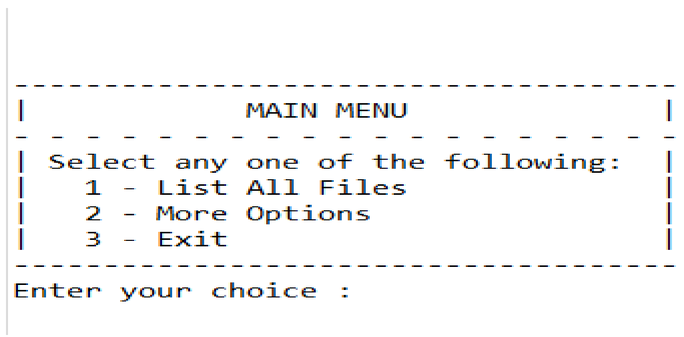


```
*      WELCOME TO THE LOCKEDME APPLICATION       *

*       PRATEEK DUBEY       *

*    Directory: C:\Users\Prateek\Desktop\assessment1     *
```

Step 3.2: Writing method to display main Menu

```
public void mainMenuOptions()
{
                System.out.println("-------------------------------------");
                System.out.println("|            MAIN MENU               |");
                System.out.println("- - - - - - - - - - - - - - - - - - -");
                System.out.println("| Select any one of the following:   |");
                System.out.println("|   1 - List All Files               |");
                System.out.println("|   2 - More Options                 |");
                System.out.println("|   3 - Exit                         |");
                System.out.println("-------------------------------------");
                System.out.println("Enter your choice : ");
}
```

Output:

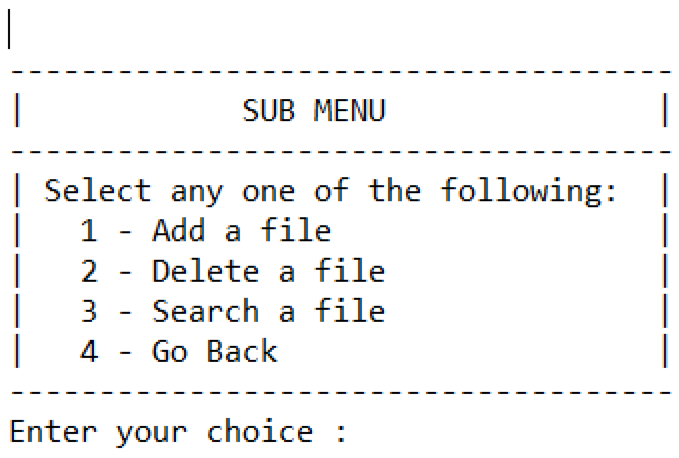


```
-------------------------------------
|            MAIN MENU               |
- - - - - - - - - - - - - - - - - - -
| Select any one of the following:   |
|   1 - List All Files               |
|   2 - More Options                 |
|   3 - Exit                         |
-------------------------------------
Enter your choice :
```

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

```
public void subMenuOptions() {
        System.out.println("-------------------------------------");
        System.out.println("|             SUB MENU              |");
        System.out.println("-------------------------------------");
        System.out.println("| Select any one of the following:  |");
        System.out.println("|   1 - Add a file                  |");
        System.out.println("|   2 - Delete a file               |");
        System.out.println("|   3 - Search a file               |");
        System.out.println("|   4 - Go Back                     |");
        System.out.println("-------------------------------------");
        System.out.println("Enter your choice : ");

    }
```

Output:



```
|
-------------------------------------
|             SUB MENU              |
-------------------------------------
| Select any one of the following:  |
|   1 - Add a file                  |
|   2 - Delete a file               |
|   3 - Search a file               |
|   4 - Go Back                     |
-------------------------------------
Enter your choice :
```

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

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

4.1. Handling input selected by user in main Menu

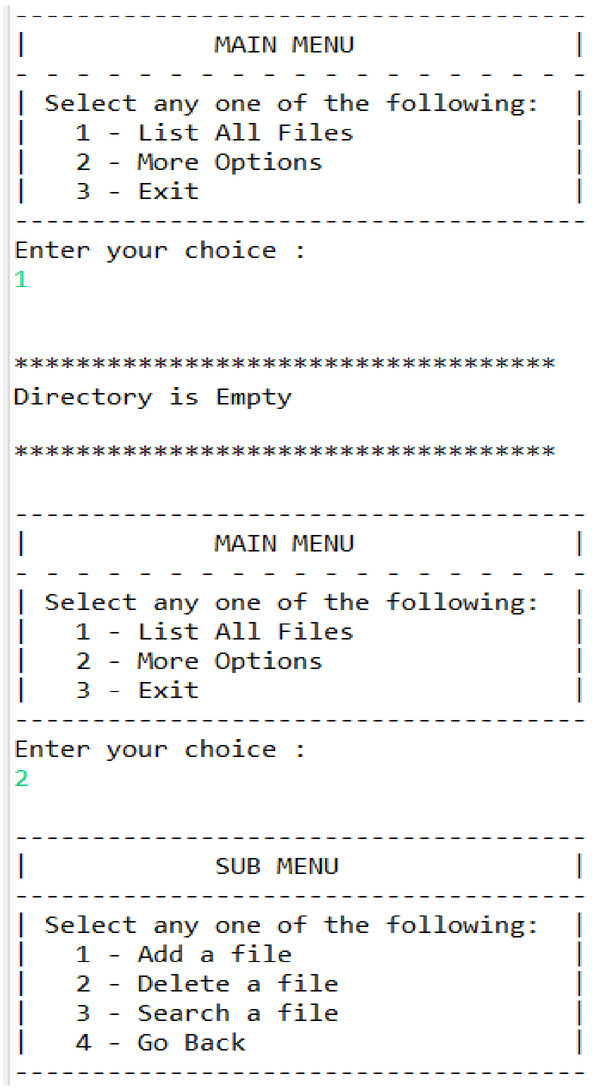
4.2. Handling input selected by user in sub Menu for File Operations

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

```
public void mainMenu()
 {
      int choice = 0;
      char decision = 0;
	do
	{
		mainMenuOptions();
		try
		{
			choice = Integer.parseInt(scan.nextLine());
		}
		catch (Exception e)
		{
			System.out.println(e);
			mainMenu();
		}
		switch (choice)
		{
			case 1:
			System.out.println();
			try
			{
				p.listAllFiles(Main.path);
			}
			catch(Exception e)
			{
				System.out.println(e.getMessage());
			}
			System.out.println("\n**********************************\n");
					break;
			case 2:
			System.out.println();
			subMenu();
			break;
			case 3:
			System.out.println("\n Are you sure you want to exit ? ");
			System.out.println("  (Y) ==> Yes    (N) ==> No         ");
			decision =  scan.nextLine().toUpperCase().charAt(0);
			if(decision == 'Y')
			{
				System.out.println("\n");
				exitScreen();
				System.exit(1);
			}
			else if(decision == 'N')
			{
				System.out.println("\n");
				mainMenu();
			}
			else
			{
				System.out.println("\nInvalid Input \nValid Inputs 
:(Y/N)\n");
				mainMenu();
			}
			default:
			System.out.println("\nInvalid Input \nValid Input Integers:(1-
3)\n");
			mainMenu();
```

```java
                }

        }
while(true);

}
```

Output:



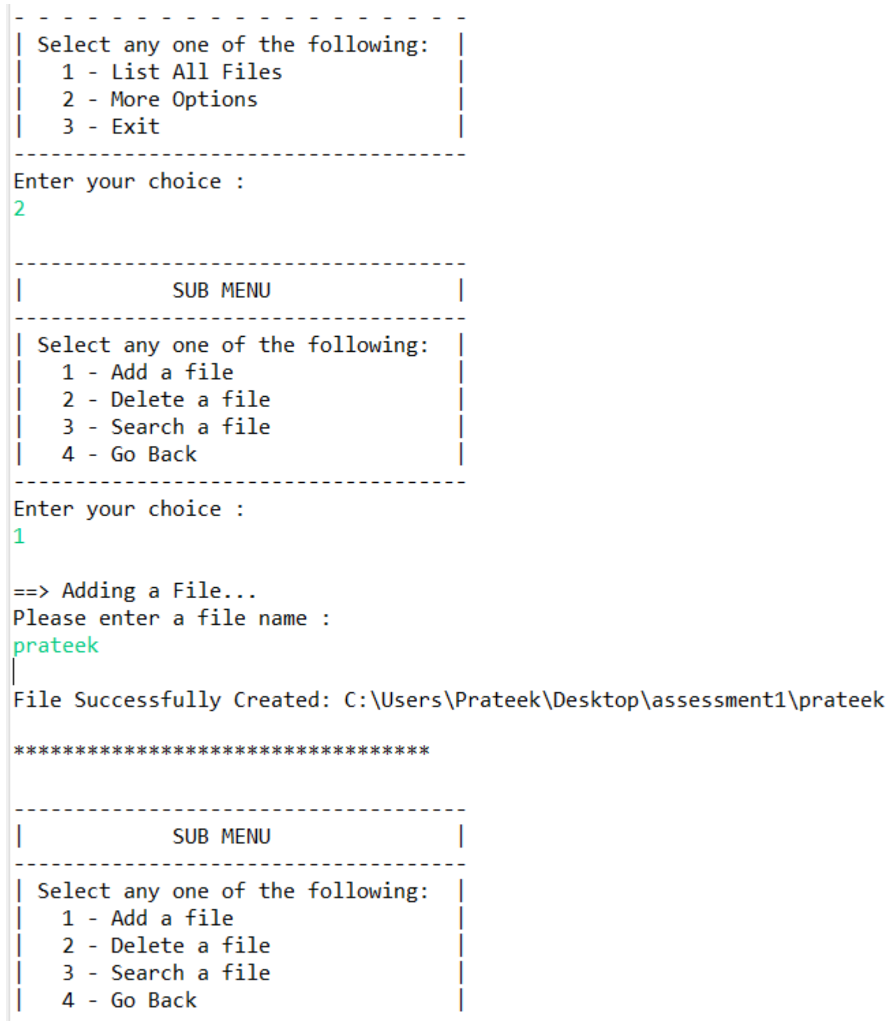
```
------------------------------------
|            MAIN MENU             |
- - - - - - - - - - - - - - - - - - -
| Select any one of the following:  |
|    1 - List All Files             |
|    2 - More Options               |
|    3 - Exit                       |
------------------------------------
Enter your choice :
1


***********************************
Directory is Empty

***********************************

------------------------------------
|            MAIN MENU             |
- - - - - - - - - - - - - - - - - - -
| Select any one of the following:  |
|    1 - List All Files             |
|    2 - More Options               |
|    3 - Exit                       |
------------------------------------
Enter your choice :
2

------------------------------------
|            SUB MENU              |
------------------------------------
| Select any one of the following:  |
|    1 - Add a file                 |
|    2 - Delete a file              |
|    3 - Search a file              |
|    4 - Go Back                    |
------------------------------------
```

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

```java
public void subMenu() {
                String file = null;
                String fileName = null;
                int choice = 0;

                do {
                    subMenuOptions();

                    try {
                           choice = Integer.parseInt(scan.nextLine());
                    } catch (NumberFormatException e) {
                           System.out.println("Invalid Input \nValid Input
Integers:(1-4)");
                           subMenu();
                    }


                    switch (choice) {
                    case 1:
                                  System.out.println("\n==> Adding a File...");
                                  System.out.println("Please enter a file name : ");
                                  file = scan.nextLine();
                                  fileName = file.trim();
                                  try {
                                         p.createNewFile(Main.path, fileName);
                                  }
                                  catch(Exception e) {
                                         System.out.println(e);

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

                    case 2:
                                  System.out.println("\n==> Deleting a File...");
                                  System.out.println("Please enter a file name to
Delete : ");

                                  file = scan.nextLine();
                                  fileName = file.trim();
                                  try {
                                         p.deleteFile(Main.path, fileName);
                                  }
                                  catch(Exception e)
                                  {
                                         System.out.println(e.getMessage());
                                  }

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

                    case 3:
                                  System.out.println("\n==> Searching a File...");
                                  System.out.println("Please enter a file name to
Search : ");

                                  file = scan.nextLine();
                                  fileName = file.trim();
                                  try {
                                         p.searchFile(Main.path, fileName);
                                  }
```

```
                                        catch(Exception e) {
                                                System.out.println(e.getMessage());
                                        }

        System.out.println("\n***********************************\n");
                                        break;
                        case 4: mainMenu();
                                        break;

                        default:
                                System.out.println("Invalid Input \nValid Input 
Integers:(1-4)");
                                subMenu();

                        }

                file = null;
                fileName = null;

                }while(true);

        }
```

Output:



```
- - - - - - - - - - - - - - - - - - - - -
| Select any one of the following:  |
|   1 - List All Files              |
|   2 - More Options                |
|   3 - Exit                        |
-------------------------------------
Enter your choice :
2

-------------------------------------
|            SUB MENU               |
-------------------------------------
| Select any one of the following:  |
|   1 - Add a file                  |
|   2 - Delete a file               |
|   3 - Search a file               |
|   4 - Go Back                     |
-------------------------------------
Enter your choice :
1

==> Adding a File...
Please enter a file name :
prateek

File Successfully Created: C:\Users\Prateek\Desktop\assessment1\prateek

**********************************

-------------------------------------
|            SUB MENU               |
-------------------------------------
| Select any one of the following:  |
|   1 - Add a file                  |
|   2 - Delete a file               |
|   3 - Search a file               |
|   4 - Go Back                     |
```

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

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

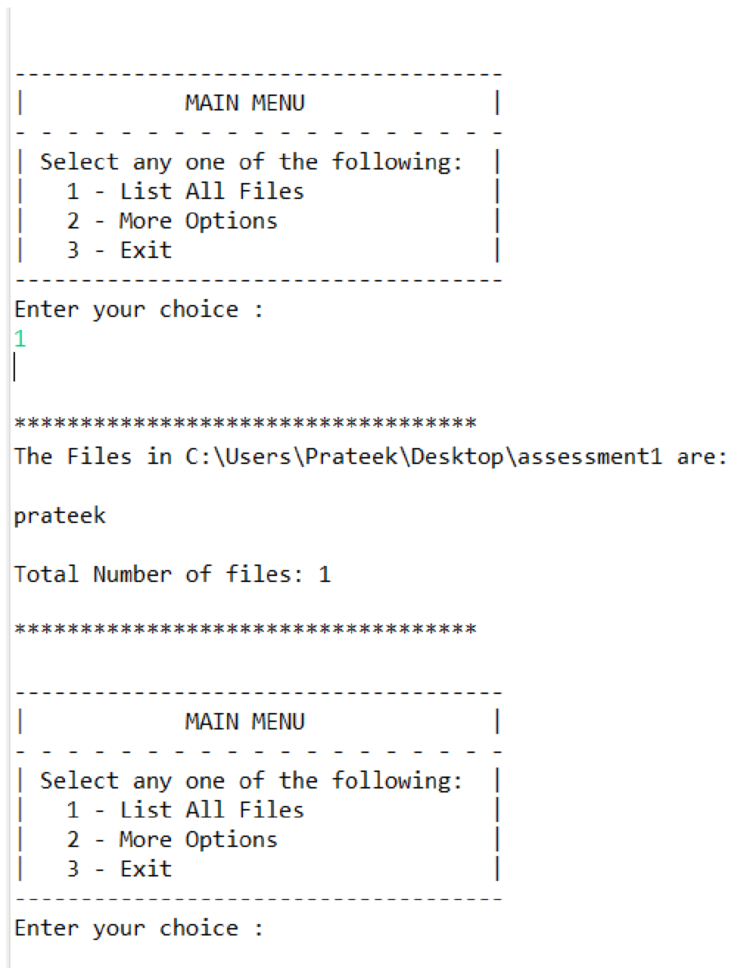
- **Operations** consists methods for -:

5.1. Displaying all files in "main" folder in ascending order .
5.2. Creating a file/folder as specified by user input.
5.3. Search files as specified by user input in "main" folder and it's subfolders.
5.4. Deleting a file/folder from "main" folder

Step 5.1: Writing method to display all files in "main" folder in ascending order

```
public void listAllFiles(String path) {
        if (path == null || path.isEmpty() || path.isBlank())
            throw new NullPointerException("Path cannot be Empty or null");


        File dir = new File(path);

        if(!dir.exists())
            throw new IllegalArgumentException("Path does not exist");

        if(dir.isFile())
            throw new IllegalArgumentException("The given path is a file. A 
directory is expected.");

        String [] files = dir.list();

        System.out.println("\n***********************************");
        if(files != null && files.length > 0) {

            Set<String>filesList = new TreeSet<String>(Arrays.asList(files));
            System.out.println("The Files in "+ dir.getAbsolutePath() + " 
are: \n");
            for(String file1:filesList) {

                System.out.println(file1);

            }

            System.out.println("\nTotal Number of files: "+ 
filesList.size());
        }else {

            System.out.println("Directory is Empty");
        }

    }
```

Output:



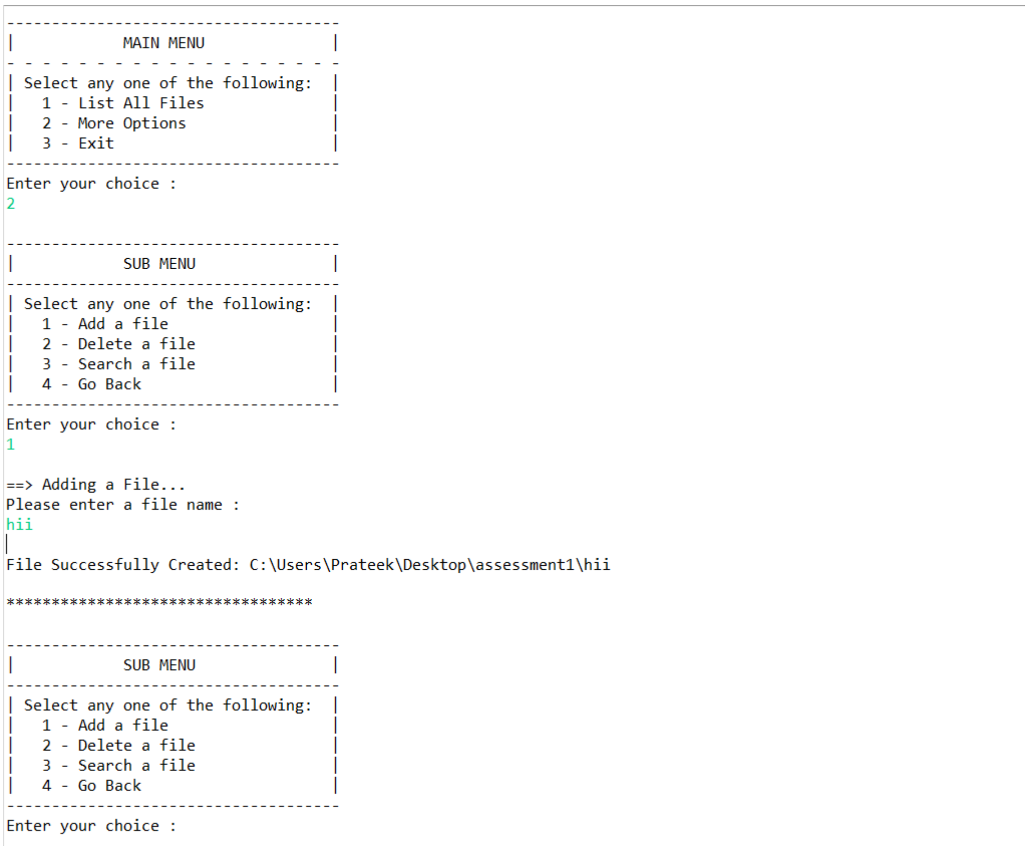
```
------------------------------------
|           MAIN MENU              |
- - - - - - - - - - - - - - - - - - -
| Select any one of the following:  |
|   1 - List All Files             |
|   2 - More Options               |
|   3 - Exit                       |
------------------------------------
Enter your choice :
1

***********************************
The Files in C:\Users\Prateek\Desktop\assessment1 are:

prateek

Total Number of files: 1

***********************************

------------------------------------
|           MAIN MENU              |
- - - - - - - - - - - - - - - - - - -
| Select any one of the following:  |
|   1 - List All Files             |
|   2 - More Options               |
|   3 - Exit                       |
------------------------------------
Enter your choice :
```

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

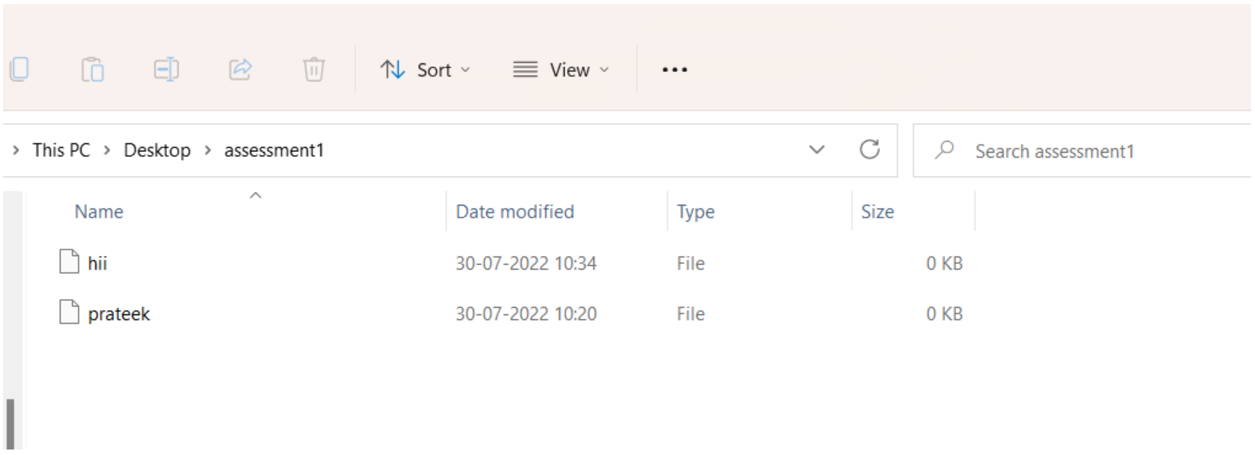
```
public void createNewFile(String path , String fileName) throws IOException {

		if (path == null || path.isEmpty() || path.isBlank())
			throw new NullPointerException("Path cannot be Empty or null");


		if (fileName == null || fileName.isEmpty() || fileName.isBlank())
			throw new NullPointerException("File Name cannot be Empty or
null");

		File newFile = new File(path + File.separator + fileName);

		boolean createFile = newFile.createNewFile();

		if (createFile) {

			System.out.println("\nFile Successfully Created: " +
newFile.getAbsolutePath());

		}else if(!createFile) {
```

```
                    System.out.println("\nFile Already Exist.. Please try again." );
                }
        }
```

Output:



```
------------------------------------
|            MAIN MENU             |
- - - - - - - - - - - - - - - - - -
| Select any one of the following:  |
|   1 - List All Files              |
|   2 - More Options                |
|   3 - Exit                        |
------------------------------------
Enter your choice :
2

------------------------------------
|            SUB MENU              |
------------------------------------
| Select any one of the following:  |
|   1 - Add a file                  |
|   2 - Delete a file               |
|   3 - Search a file               |
|   4 - Go Back                     |
------------------------------------
Enter your choice :
1

==> Adding a File...
Please enter a file name :
hii
|
File Successfully Created: C:\Users\Prateek\Desktop\assessment1\hii

**********************************

------------------------------------
|            SUB MENU              |
------------------------------------
| Select any one of the following:  |
|   1 - Add a file                  |
|   2 - Delete a file               |
|   3 - Search a file               |
|   4 - Go Back                     |
------------------------------------
Enter your choice :
```

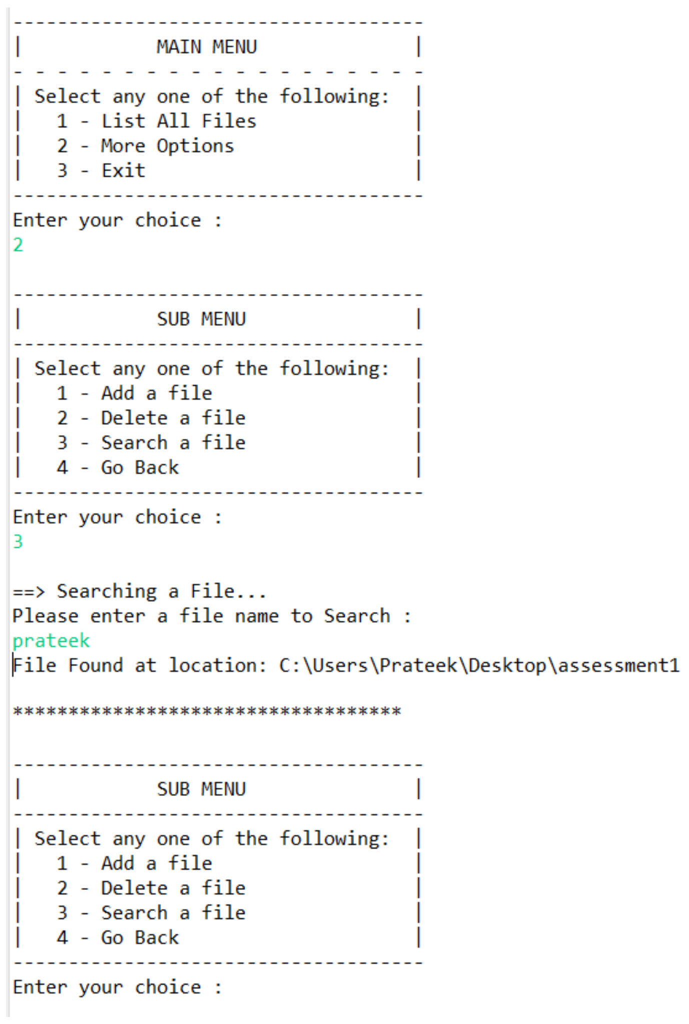


Name	Date modified	Type	Size
hii	30-07-2022 10:34	File	0 KB
prateek	30-07-2022 10:20	File	0 KB

Step 5.4: Writing method to search for all files as specified by user input in “main” folder and it’s subfolders.

```
public void searchFile(String path , String fileName){

            if (path == null || path.isEmpty() || path.isBlank())
                  throw new NullPointerException("Path cannot be Empty or null");


            if (fileName == null || fileName.isEmpty() || fileName.isBlank())
                  throw new NullPointerException("File Name cannot be Empty or
null");

            File dir = new File(path);

            if(!dir.exists())
                  throw new IllegalArgumentException("Path does not exist");

            if(dir.isFile())
                  throw new IllegalArgumentException("The given path is a file. A
directory is expected.");


            String [] fileList = dir.list();
            boolean flag = false;

            Pattern pat = Pattern.compile(fileName);

            if(fileList != null && fileList.length > 0) {
                  for(String file:fileList) {
                        Matcher mat = pat.matcher(file);
                        if(mat.matches()) {
                              System.out.println("File Found at location: " +
dir.getAbsolutePath());
                              flag = true;
                              break;
                        }
                  }
            }

            if(flag == false)
                  System.out.println("File Not Found.. Please try again.");


      }
```

Output:

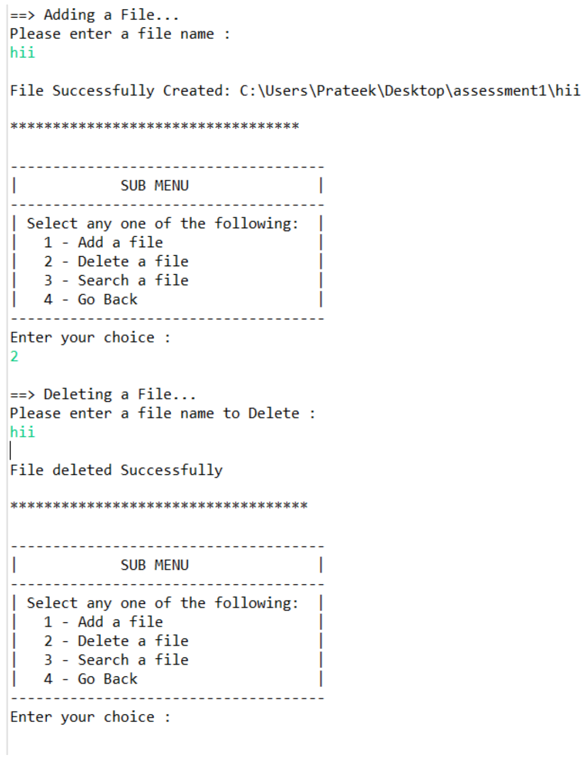


```
----------------------------------------
|             MAIN MENU                 |
- - - - - - - - - - - - - - - - - - - -
| Select any one of the following:      |
|    1 - List All Files                 |
|    2 - More Options                   |
|    3 - Exit                           |
----------------------------------------
Enter your choice :
2

----------------------------------------
|             SUB MENU                  |
----------------------------------------
| Select any one of the following:      |
|    1 - Add a file                     |
|    2 - Delete a file                  |
|    3 - Search a file                  |
|    4 - Go Back                        |
----------------------------------------
Enter your choice :
3

==> Searching a File...
Please enter a file name to Search :
prateek
File Found at location: C:\Users\Prateek\Desktop\assessment1

***********************************

----------------------------------------
|             SUB MENU                  |
----------------------------------------
| Select any one of the following:      |
|    1 - Add a file                     |
|    2 - Delete a file                  |
|    3 - Search a file                  |
|    4 - Go Back                        |
----------------------------------------
Enter your choice :
```

Step 5.5: Writing method to delete file/folder specified by user input in "main" folder

```
public void deleteFile(String path , String fileName) throws IOException {

            if (path == null || path.isEmpty() || path.isBlank())
                  throw new NullPointerException("Path cannot be Empty or null");


            if (fileName == null || fileName.isEmpty() || fileName.isBlank())
                  throw new NullPointerException("File Name cannot be Empty or
null");

            File newFile = new File(path + File.separator + fileName);

            boolean deleteFile = newFile.delete();

            if (deleteFile) {

                  System.out.println("\nFile deleted Successfully");
            }else {
                  System.out.println("\nFile Not Found.. Please try again." );
            }
      }
```

Output:



```
==> Adding a File...
Please enter a file name :
hii

File Successfully Created: C:\Users\Prateek\Desktop\assessment1\hii

**********************************

-------------------------------------
|             SUB MENU              |
-------------------------------------
| Select any one of the following:  |
|   1 - Add a file                  |
|   2 - Delete a file               |
|   3 - Search a file               |
|   4 - Go Back                     |
-------------------------------------
Enter your choice :
2

==> Deleting a File...
Please enter a file name to Delete :
hii
|
File deleted Successfully

***********************************

-------------------------------------
|             SUB MENU              |
-------------------------------------
| Select any one of the following:  |
|   1 - Add a file                  |
|   2 - Delete a file               |
|   3 - Search a file               |
|   4 - Go Back                     |
-------------------------------------
Enter your choice :
```

Step 6: 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 doesn't restrict user to specify the exact filename to search/delete file/folder.
3. The application also allows user to delete file which are not empty.
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. When the option to retrieve files in ascending order is selected, user is displayed with two options of viewing the files.

   5.1. Ascending order of all files and folders inside the "main" folder.

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.
- Allowing user to search, delete, add file in the main folder.